

Title:

eFluidChart: MVP for pilot in VIRTUE QI project

Author:

Edward (Ned) Lloyd

Supervisor:

Dr Graham Roberts

Year of Submission:

2018

Degree Program:

MSc Computer Science

Disclaimer:

*"This report is submitted as part requirement for the MSc Computer Science degree at UCL. It is substantially the result of my own work except where explicitly indicated in the text."*

## Abstract

Fluid prescription in the NHS is a serious problem. This is true particularly for junior clinicians who have had little hospital experience. eFluidChart is a fluid prescription simulator that will help provide them with this necessary experience. I took over the project in the second phase of development. At this point it was already nearly functional. My role on the project was two-fold. First, it was to bring the project to a level where it was entirely functional and could be user-tested in a hospital. The other objective was to do with the longer term aims of the project. The plan is to eventually roll eFluidChart out in hospitals throughout Northern Ireland. Different hospitals have different environments so new features need to be added to ensure a level of flexibility for the application.

The project was created using the ASP.NET Core MVC framework. My biggest challenge was to understand both this technology and the way that the previous developer had used it to create the project structure. To make good reliable additions to the code, I needed to spend time carefully studying these aspects of the application. Another part of achieving this goal was to create a good testing strategy for my work.

The goals of the project were all achieved. I did bring the application to a level of functionality where it could be tested in a hospital. I also implemented a number of other features that would increase the ease with which the application could be used in other environments. My only qualification is that I would have liked to have created a more comprehensive automated testing strategy. However, I consider this a small qualification. In the time that I had, this was probably an unrealistic goal. In addition, I was able to supplement my automated testing with manual testing, and in doing so I did create an effective testing strategy.

## Table of Contents

|   |           |
|---|-----------|
| ABSTRACT .....  | 2         |
| TABLE OF CONTENTS .....                               | 3         |
| MAIN REPORT .....                                     | 6         |
| <b>Chapter 1: Introduction .....</b>                  | <b>6</b>  |
| 1.1 Problem and Background.....                       | 6         |
| 1.2 Goals.....  | 6         |
| 1.3 Aims.....   | 7         |
| 1.4 Project Approach .....                            | 8         |
| 1.5 Chapter Structure.....                            | 8         |
| <b>Chapter 2: Context and Design Strategy .....</b>   | <b>10</b> |
| 2.1 The Current Solution .....                        | 10        |
| 2.2 Our Solution .....                                | 11        |
| 2.3 Context: Approaches to the Problem .....          | 12        |
| 2.3.a Inflexibility of Technology and Structure ..... | 13        |
| 2.3.b Mid-July Deadline .....                         | 13        |
| 2.3.c Continued Development .....                     | 14        |
| 2.3.d Iterative Design .....                          | 14        |
| 2.3.e Test-Driven Approach .....                      | 15        |
| 2.3.f Version Control Using GitHub.....               | 15        |
| 2.3.g Chosen Technologies – Virtual Machine.....      | 15        |
| <b>Chapter 3: Requirements and Analysis .....</b>     | <b>18</b> |
| 3.1 The Problem: My Perspective .....                 | 18        |
| 3.2 Project Modules .....                             | 19        |
| 3.3 Requirements Gathering.....                       | 19        |
| 3.4 Importance of Use Cases .....                     | 22        |
| <b>Chapter 4: Design and Implementation .....</b>     | <b>25</b> |
| 4.1 ASP.NET Core MVC.....                             | 25        |
| 4.1.a Views .....                                     | 25        |
| 4.1.b Models and View Models .....                    | 26        |
| 4.1.c Controllers .....                               | 26        |
| 4.1.d Separation of Concerns.....                     | 26        |
| 4.2 Inherited Design.....                             | 27        |
| 4.2.a Database .....                                  | 27        |
| 4.2.b NHS.Fluid.Model .....                           | 28        |
| 4.2.c NHS.Fluid.Contracts .....                       | 29        |
| 4.2.d NHS.Fluid.AutoMapper .....                      | 29        |
| 4.2.e NHS.Fluid.Persistence .....                     | 30        |

|  |           |
|--|-----------|
| 4.2.f NHS.Fluid.Services .....   | 30        |
| 4.2.g NHS.Fluid.PersitenceSupport .....  | 32        |
| 4.2.h NHS.Fluid.Core .....   | 32        |
| 4.2.i NHS.Fluid.WebApp .....   | 32        |
| 4.3 My Changes .....   | 32        |
| 4.3.a Inflexibility of Technology and Deadline .....                               | 33        |
| 4.3.b Implementation of Prescription Validation Logic and Iterative Approach ..... | 35        |
| 4.3.c Creation of Tier 3 Feedback and User Feedback Section .....                  | 37        |
| 4.3.e Security .....   | 38        |
| <b>Chapter 5: Testing .....</b>  | <b>39</b> |
| 5.1 Testing Aims .....   | 39        |
| 5.2 Constraints on Testing Strategy .....  | 39        |
| 5.3 Automated Testing .....  | 40        |
| 5.4 Manual Testing .....   | 42        |
| <b>Chapter 6: Conclusions .....</b>  | <b>43</b> |
| 6.1 Goals and Aims .....   | 43        |
| 6.1.a Ensuring the Project was Completely Functional .....                         | 43        |
| 6.1.b Improving the Look and Feel of the Application .....                         | 44        |
| 6.1.c Ensuring the Project can be Scaled up for use in other Hospitals. ....       | 44        |
| 6.1.d Creating set of Automated Tests for the Project .....                        | 45        |
| 6.2 Mid July Deadline .....  | 45        |
| 6.3 Critical Evaluation .....  | 46        |
| <b>BIBLIOGRAPHY .....</b>  | <b>47</b> |
| <b>APPENDIX .....</b>  | <b>49</b> |
| <b>Appendix A: System Manual .....</b>   | <b>49</b> |
| A.1 Database Firewall .....  | 49        |
| A.2 Virtual Machine Setup .....  | 49        |
| A.2.a Create Virtual Machine .....   | 49        |
| A.2.b Configure Virtual Machine .....  | 50        |
| A.3 Setup from Windows Environment. ....   | 51        |
| A.4 Run Selenium IDE tests .....   | 52        |
| <b>Appendix B: User Manual .....</b>   | <b>54</b> |
| B.1 URL .....  | 54        |
| B.2 Application Roles .....  | 54        |
| B.3 Toolbar: .....   | 54        |
| B.4 Login .....  | 54        |
| B.5 Dashboard .....  | 54        |
| B.6 MyCase .....   | 55        |
| B.6.a MyCase index .....   | 55        |
| B.6.b MyCase View .....  | 55        |

|   |           |
|---|-----------|
| B.7 Case, User and Fluid Setup .....                          | 57        |
| B.7.a Index Page .....  | 57        |
| B.7.b Edit Page .....   | 58        |
| B.8 Global Setting Setup .....                                | 59        |
| <b>Appendix C: Supporting Documentation and Diagrams.....</b> | <b>59</b> |
| C.1 Project Outline Provided by Dr Alexander Davey.....       | 59        |
| C.1.a General Project background .....                        | 59        |
| C.1.b Feedback Background .....                               | 60        |
| C.2 NICE National Fluid Prescription Guidelines .....         | 63        |
| C.2.a General Guidelines.....                                 | 63        |
| C.2.b Fluid Prescription Type Guidelines .....                | 64        |
| C.2.c Feedback Explainer .....                                | 64        |
| C.3 Use Cases .....   | 65        |
| C.3.a Use Case Diagram .....                                  | 65        |
| C.3.b Use Cases .....   | 65        |
| C.4 Database Schema.....                                      | 72        |
| C.5 My Changes in Full .....                                  | 72        |
| C.6 GetInclude() method.....                                  | 74        |
| C.7 Validation Error .....                                    | 74        |
| C.8 Adherence to Design Conventions .....                     | 74        |
| C.9 Screen Size Comparison.....                               | 76        |
| <b>Appendix D: Testing.....</b>                               | <b>78</b> |
| D.1 Requirements With Outline of How They Were Tested .....   | 78        |
| D.2 Final Testing Errors .....                                | 79        |
| D.3 Test Example.....   | 80        |
| <b>Appendix E: Code Listing.....</b>                          | <b>82</b> |
| E.1 NHS.Fluid.Model .....                                     | 82        |
| E.2 NHS.Fluid.Contracts .....                                 | 85        |
| E.3 NHS.Fluid.AutoMapper .....                                | 90        |
| E.4 NHS.Fluid.Persistence .....                               | 90        |
| E.5 NHS.Fluid.Services.....                                   | 91        |
| E.6 NHS.Fluid.PersitenceSupport .....                         | 93        |
| E.7 NHS.Fluid.WebApp: Controller .....                        | 93        |
| E.8 NHS.Fluid.WebApp: Models (View Models) .....              | 100       |
| E.9 NHS.Fluid.WebApp: Views .....                             | 105       |
| E.10 NHS.Fluid.WebApp: Scripts .....                          | 115       |

## Main Report

### Chapter 1: Introduction

#### 1.1 Problem and Background

Intravenous fluid prescription is a task that many clinicians do on a very regular basis. It is important that patients are given the correct electrolytes in the correct amounts at the correct rate. Failure to do this can be extremely serious and even result in patient death [1]. Despite its importance fluid prescribing is still regularly done badly. There is a theory practice gap between the national fluid guidelines and their application in the hospital (see Appendix C.1 for more detail). This is particularly true for junior clinicians who have not had the necessary real-world experience. The problem is compounded by the fact that both junior and senior clinicians are working on very busy wards. Senior clinicians do not have time to advise their less experienced peers and junior clinicians are too busy to be able to give each prescription the time needed to learn from it. The problem that this application aims to solve is to bridge this theory practice gap, helping clinicians gain the experience needed to safely prescribe intravenous fluids.

The application aims to bridge this gap by providing clinicians with a simulator for fluid prescription. Users can respond to cases set by experienced clinicians and receive meaningful feedback based upon their prescriptions. The project is the brainchild of Dr Alexander Davey and it was he whom I had most contact with during my work on it. Before I started, the project had already gone through one round of development. This was done by 'Create Software', a Belfast based Software company. I was in contact with Simon Smith, the developer from 'Create Software' intermittently throughout the project. When I was finished with the project I would also hand it over to another programmer who would continue its development.

#### 1.2 Goals

The overall aim of the project was to bring the fluid prescription training simulator to a point where it could be used in the Belfast hospital where Dr Davey is based. The project would

then be user tested before the next round of development. Therefore, my most important goal was to make the application functional.

For this next round of development, the project would need funding. To obtain this it would need to be attractive for use in other hospitals. This was another important goal which could be broken down into two aspects. It would need a professional look and feel, and it would need to have features that would allow it to work in different clinical environments. My final goal was not something requested by Dr Davey but something that I thought would add a lot to the project. I wanted to create a set of automated tests for the application. This was for use for the next developer and would ensure that the code I handed over was reliable. The goals can be broken down in more detail as follows:

1. Ensure project was completely functional.
  - a. Develop and add to case feedback that the prescription user receives.
  - b. Add option for user to give their feedback on the case on completion of prescription.
  - c. Include potassium in dose summary of electrolytes prescribed.
  - d. Add NEWS score to case summary.
2. Improve look and feel of the project.
  - a. Display case images as thumbnails, not as full images when presenting the case in order to increase performance when loading page.
3. Ensure project can be scaled up for use in other hospitals.
  - a. Add fluid library management system allowing the administrator to add fluids used in their hospital.
  - b. Add Additive library management system allowing the administrator to add additives used in their hospital.
  - c. Ensure project looks professional on different screen sizes.
4. Create a set of automated tests for the project.

### 1.3 Aims

In order to achieve these goals, I set upon the following aims.

1. Develop a project approach that takes account of the position in the design lifecycle where I inherited the project, and the specific aims of the client.
2. Learn how to use the technologies that the project has been built with. Most importantly ASP.NET Core MVC and C#.
3. Learn how the project has been structured in order to add to the project in a way that builds sensibly upon, and is consistent with, the previous developer: Simon Smith. This would be important when passing the project onto the next developer.
4. Maintain good communication with Dr Davey throughout the process. It would be important to frequently check to ensure I had correctly understood the requirements that dealt with medical information. Dr Davey was also an important source of information regarding the needs of the end users. He could evaluate the project from the perspective of Junior and Senior Clinicians who would eventually be using it.
5. Maintain some communication with Simon Smith. He would be a great resource on understanding how the project structure works.
6. Test the project thoroughly to ensure that code delivered was reliable.

#### 1.4 Project Approach

I wanted my approach to the problem to take into account the specific circumstances in which I inherited the problem. These were, most notably, the fact I was taking over the project at a late stage and would eventually be handing it to someone else. Therefore, I had no choice in the technologies used, and there was already a very definite structure to the project that I would need to follow. The computer science principles I was particularly keen to bring to the project were iteration, test driven design and version control using GitHub.

#### 1.5 Chapter Structure

**Chapter 2 – Context and Design Strategy:** This chapter will outline the problem of fluid prescription and will state how the application aims to respond to this. It will also outline, in great detail, my design plan. This will include an explanation of why I used the technologies that I did.



**Chapter 3 – Requirements and Analysis:** This chapter aims to outline what my role was on the project. It states how I went about interpreting the requirements and details the important role that use cases played in this.

**Chapter 4 – Design and Implementation:** This chapter will outline the structure of the project in its entirety. It will then describe how I implemented my design strategy. Both parts will be illustrated with examples from my work on the project.

**Chapter 5 – Testing:** This chapter will set out the aims of my testing strategy. I will then consider the constraints and how the tests were implemented.

**Chapter 6 – Conclusions:** This chapter will evaluate how well I achieved the goals set out in chapter 1. It will then consider what I would have changed about the project and offer a final evaluation on all the work I have done.

## Chapter 2: Context and Design Strategy

### 2.1 The Current Solution

The current approach is for junior doctors to prescribe fluids to patients using their knowledge, learnt in medical school, of the national fluid prescription guidelines. They will also have senior clinicians to help them apply these rules in the real world. The approach combines practical experience with imparted experience from more senior clinicians. In theory, it is successful. Doctors do generally develop the skills necessary to safely prescribe fluids. The problem is that it comes at a cost of too much harm done to patients.

Hospitals are busy places. This is a key reason why the current approach is not satisfactory. Junior doctors do not have enough time to carefully consider each case. Without doing this the amount they will learn from each case will be limited. In addition, it is not the case that a single doctor will be attached, for the duration of a stay, to a single patient. In most cases, if a mistake is made with fluid prescription the consequences will not be dire. This means another doctor could correct the problem without the first ever knowing. The lack of feedback will limit the didactic potential of the prescription. Senior clinicians are also part of this hospital environment. They, also, do not have enough time to be constantly on hand to have their experience tapped into. The current solution does not stand within the context of very busy hospitals, but any future solution must.

Another point to note is that fluids constantly need to be prescribed, and it would be unfeasible to stop junior clinicians taking part in this until they can be proved to have a high level of expertise. There simply would not be enough staff. In addition, hands-on experience with real patients is a necessary part of the learning process that cannot be substituted. For these two reasons it is impossible to replace the current solution entirely. Any solution must compliment the approach rather than replace it.

There are aspects that cannot be changed when developing a solution to the problem. These are that:

A1. Junior Clinicians must continue to prescribe fluids to the patients.

A2. The hospital environment will always be busy.

The current approach fails in this environment because:

B1. Clinicians don't have enough time to carefully consider each case.

B2. Clinicians may not get good feedback from their prescription.

B3. Senior clinicians are too busy to be on hand to offer advice to their less experienced peers.

The aspects of the current approach that work well are:

C1. Using the experience of senior clinicians

C2. Giving clinicians real world experience

## 2.2 Our Solution

This project is to create a fluid prescription simulator. This will be used alongside the education that junior clinicians will get from practical experience within the wards. Clinicians will work through scenarios uploaded by their more experienced peers and receive constructive feedback on their prescriptions. The scenarios will be composed of various details about the patient such as weight, NEWS Score (a number indicating the risk of patient's condition deteriorating) [2] and a narrative describing the patient in more textual terms. Junior clinicians will respond to these scenarios. They will enter data such as amount of fluid, fluid type and fluid rate as well as when the patient next must be reassessed and any further comments they may have. Clinicians responding to these scenarios will start to see how guidelines are to be applied in 'messier' real world situations.

Clinicians will be able to log into the system, using a username and password provided to them, on their own laptops or phones. Avoiding drawback B1 doctors will be able to run through the cases wherever and whenever they like. They will not need to fit them in around other tasks on the ward. In addition, as an improvement on B2, the system will provide detailed feedback for the cases. This feedback will be from the National Fluid Prescription Guidelines (Appendix C.1.b, C.2.a and C.2.b provide a detailed overview of the feedback provided. For a simple overview see Section C.2.c). Junior clinicians will have time go through scenarios and consider the detailed feedback they will receive.

The application also tackles B3. Senior clinicians do not have time to successfully impart their experience. Using this application, they will be able to upload cases that contain this experience. This is designed to accentuate strength C1. They will only have to impart this experience once. It must be noted, this is not meant as a replacement for the normal help that senior clinicians will still offer to their junior peers on individual cases in the ward. It is meant to supplement this.

The application will stand in the context of a busy hospital (A1-A2). It will improve in the areas that the current solution fails (B1-B3). It also preserves and accentuates the good parts of the current solution (C1-C2). Finally, it is important to note that the simulator will never completely close the theory practice gap. Clinicians will not be able to arrive on their first day on the ward completely competent in fluid prescription. The aim is to reduce the amount of time that junior clinicians are prescribing without being fully competent.

### 2.3 Context: Approaches to the Problem

Having created project aims (Section 1.3) I wanted to cement these into a detailed design strategy. I wanted to approach the unique context of the project with good computer science principles. I decided that the context of the project could be summed up by the following points:

- a) Inflexibility of Technology and Structure
- b) Mid-July Deadline
- c) Continuing Development

The computer science principles I wanted to make particular use of were:

- d) Iteration
- e) Test-driven Design
- f) Version control using GitHub

Finally, I still needed to decide how I would run the code on my machine. Simon Smith had developed the project on a Windows environment and I would need to decide how I would run the technologies that were part of my project on my macOS operating system.

- g) Chosen Technologies: Virtual machine

### *2.3.a Inflexibility of Technology and Structure*

In order to write reliable code, I would have to understand what I had written, and I would have to make it consistent with what had come before me. Neither of these things would be possible without a deep understanding of the technologies and project structure. I had no choice in software and only a little flexibility on the structure. The project structure was also fairly complex and contained concepts that, at the start of the project, I had no experience with. I had never, for example, been part of a project that used a persistence or service layer.

Particularly in a project without tests, I deemed it dangerous to start trying to add things to the project until I had a full understanding of how it worked. I would be very likely to make mistakes and might not be able to spot any errors I did make. In addition, it was important that all my changes cohered with the project structure. In a project that I was starting from scratch I might have felt freer to 'learn on the job' and pick up the frameworks by playing around with them. Here, I decided to spend two to three weeks studying the ASP.NET Core MVC framework and the project structure before I made any additions of my own. For this I made great use of 'Lynda.com' [3]. Another important resource was Simon Smith (from 'Create Software') who was happy to explain certain concepts to me. Alongside these, I made great use of the Microsoft ASP.NET Core MVC documentation [4].

### *2.3.b Mid-July Deadline*

When I had finished developing the project it would be used to obtain funding for the next layer of development. Dr Davey initially requested that the coding in the project be completed by mid-July. It soon became clear that it was unrealistic. If I started at the beginning of June, planned the project for my first week and then spent two and a half weeks studying the project, I would only have three to four weeks to implement all the requirements. I informed Dr Davey of this to which he told me that it was an aim rather than a hard deadline. Even so I aimed to provide him with as much of an improvement as I

could in the application by this date. This was an added time constraint on the project and was a particular influence on my testing-strategy (see Chapter 5 for more details).

### *2.3.c Continued Development*

I had to consider the future of the project after I had finished developing it. In order to ensure someone else would easily be able to take over the project I wanted to write clear and reliable code. I have already stated the importance of carefully studying the project structure in pursuit of this goal. The other key part to this was creating, and adhering to, a good testing strategy (my testing strategy is explained in detail in Chapter 5).

### *2.3.d Iterative Design*

Taking over the project mid-way through a long design lifecycle meant that aspects of the project would not be iteratively developed. Much of the technology was already in place by the time I took the project over. The same was true for the requirements. However, this lack of flexibility meant that other aspects of the project had to be pursued in a more iterative manner.

Dr Davey handed the requirements to me. Because I had not created them myself, my difficulty lay in their interpretation. None of the requirements required a degree in medicine to understand but some of them are extensive and complex. This was particularly true for those associated with the National Fluid Prescription Guidelines (these were all the requirements associated with case feedback – see Appendix C.2.c). The extensiveness of these meant my work on the project would not capture all the guidelines. We decided to focus on the most important ones and, if there was time, to build on these. The other important factor was that Dr Davey had no coding experience of his own. This knowledge gap on both our parts required constant communication and iteration on what was developed. Dr Davey and I left the feedback requirements more flexible for this reason.

Any changes I made I would run past Dr Davey. He would make sure what I had done was correct from a medical point of view, and I would talk to him about what could be

realistically implemented in the time frame from a programming perspective. Each time I made a significant addition to the project the changes I made would be pushed to the Azure web application. Dr Davey would then view and test the changes from Belfast. Simon Smith had deployed the code as an Azure web app before I began work on it. I did not have access to the Azure account and would get Simon to deploy the changes I had pushed to GitHub.

Ideally, we would have had junior clinicians testing each deployment. However, this would happen at the next stage of development (after my work on the project was done). This user testing would be time consuming for Dr Davey, and also for the junior clinicians involved. It would not be realistic to do it on a regular basis. My development of the project would not be informed by user-testing with junior clinicians. Instead, Dr Davey would evaluate each iteration himself. In addition, for more minor changes, we would speak on the phone and discuss them.

#### *2.3.e Test-Driven Approach*

I used a combination of automated tests using Selenium IDE and manual testing. I decided that because of the time constraints I would not do automated unit-testing. For more detail see chapter 5.

#### *2.3.f Version Control Using GitHub*

GitHub [5] with SmartGit [6] were both part of the technologies used in the project that I inherited. I used these technologies so that I could cleanly undo changes I had made. GitHub also meant that, if I did make a mistake but wasn't able to spot it until later, I would be able go back to an older version of the code and view any changes that had been made. I would use GitHub to, as much as possible, replace the added confidence in my code that would have come from having automated unit-tests. Both ensured that if mistakes were made they could be more easily spotted and rectified.

#### *2.3.g Chosen Technologies – Virtual Machine*

There was one big software decision that I did need to make. This was how I would run the ASP.NET Core MVC [4] framework on my laptop. I have a MacBook and ASP.NET is a primarily Windows framework. One option would have been to use a Visual Studio Community [7] for Mac. However, I understood that Visual Studio for Mac was not completely the same as the regular Visual Studio [8]. In addition, 'Create Software' had used programs such as Internet Information Services (IIS) 10.0 Express [9] and SQL server management studio [10]. Neither of these would work on a MacOS environment.

It would have been possible to replace both of these programs by other software. Instead, I used a virtual desktop. I decided that it would be advantageous to, as much as possible, use the same technologies as 'Create Software' had. I hoped this would be conducive to writing code consistent with the original project. The reason for this was that Simon Smith was an important resource during the project. At the beginning, Simon helped me set up the project on my laptop. He would not have been able to help me had I used programs he had not worked on. Simon's help did not end at this point. We had intermittent meetings throughout the project. Having the same technologies meant that the project looked the same on our respective laptops and this made his advice clearer to me. Technologies such as SmartGit were also fairly complex and he was able to answer questions I had on it.

For these reasons I decided to use a Virtual Machine. The choice itself proved to be fairly easy. It needed to be free and, ideally, it needed not to have too steep a learning curve because of the time constraints detailed. I considered the Apple Boot Camp [11] (a 'dual-boot' system rather than a virtual machine), Virtual Box [12] and Wine (a compatibility layer rather than a virtual machine) [13]. I decided to avoid Apple Boot Camp because I didn't want to forgo use of my Mac applications whilst working on my project. In addition, I did not necessarily need the high performance that this approach would provide. Wine, I also ruled out because of the difficulty of setting it up.

This left Virtual Box as the simplest option particularly as it was the only one that was a true virtual machine. I was also attracted to it because, despite not having technical support (which would be rare on an open source application), I learnt it did have its own busy user forum where questions could be asked [14]. In the end, Virtual Box was a fairly good option.



In particular, it was easy to set up using a YouTube tutorial [15]. The only down side was that it was slow, but this was not the end of the world.

## Chapter 3: Requirements and Analysis

### 3.1 The Problem: My Perspective

The problem the project aims to respond to is the theory practice gap in prescribing fluid. This, and the way this application aims to respond to it, is detailed in Sections 2.1 and 2.2. Here, I will give an outline of the issues that I faced during my work on the project. When I took over the project it had already gone through the first round of development. The only part needed for the project to be functional was the addition of Tier 3 feedback and improvement of Tier 2 feedback (see Appendix C.2.c for feedback explainer). Creating a working application was the bare minimum required for the project, so this was the most important challenge facing me.

Creating a functioning project was not the only goal. The project was started by Dr Alexander Davey and is originally intended for the Belfast hospital where he is based. However, in the longer-term it is hoped that it will be rolled out to other hospitals in Northern Ireland. When other hospitals consider whether to use the application they will consider it along a number of lines.

- P1. How well application functions in the original hospital
- P2. The look and feel of the application
- P3. How well application will function at another hospital

P1 directly relates to the challenge, stated above, of creating a functioning application. P2 is also fairly clear. Software that appears well designed will give the user more confidence in it. Much of the front end had been completed by the time I took over. I needed to make sure that the new features I added would adhere to the project design constraints and follow good usability principles. Another part of front end design was ensuring that the project would work well on different screen sizes. This also related to P3 as being able to be viewed on different screen sizes would mean the application could be better used on

different machines. This added flexibility would hopefully make it easier to use in different hospital environments.

This was a 'Could Have' requirement (see Section 3.3) so I did not feel comfortable making sweeping, time-consuming changes to the application. Instead, I tried to preserve the feel of the desktop site as much as possible. The application is fairly mobile friendly as it is. The navigation, for example, already adhered to most of the guidelines provided by the 'Interaction Design Foundation' [16]: It contains only two layers and the toolbar on the left used to navigate to the various modules has clear and concise labelling. I endeavoured to ensure these features translated well to smaller screens.

P3 required a deeper knowledge of clinical practices themselves. The fluids used in the Belfast hospital where the application would be initially used were already in the database when I took over the application. However, other hospitals use different fluids. In order to make the application scalable there needed to be a way to add new fluids to the database. This was also the case for additives. However, there is a lot less variation in the additives used in Northern Ireland and this is why the requirement was just a 'Could Have'.

### 3.2 Project Modules

The project was split into 7 different modules. The functionality of each is spelled out in depth in the User Manual (Appendix G). In my work on the project I did not work on the Login, Dashboard and User modules. For the Case, Fluid and MyCase modules I edited and added to work already present in the project. I created the 'Global Setting' module entirely myself.

### 3.3 Requirements Gathering

The requirements themselves were handed to me from Dr Davey. They came from research he had done with stakeholders. This would have been with junior clinicians (he himself was one up until very recently) as well as nurses (who will often administer the fluids) and the patients themselves. Dr Davey had also done extensive research on the problem of fluid

prescription from a medical point of view. It was not, therefore, my place to change the requirements from this perspective.

My role was to focus on implementing, not adjusting, the requirements (or only adjusting them if they could not be implemented in the way they had been set out). I have explained in Section 2.3.d, the iterative approach I took to understanding the requirements that had been passed down to me. However, this could only be done once I had implemented them in some form. I needed to plan how I would do this.

The requirements I started with were fairly general. They corresponded, more or less, to the goals outlined in Section 1.2. I broke these down into sub-requirements and split along the lines of the three prongs of the problem I faced (P1-P3 – above). I tried to split them into small self-contained tasks that I could approach individually. For example, I split the Generic Feedback section (tier 3) into two requirements. One for the admin to be able to upload feedback to the database and one for the user to view the feedback.

I also split them into Functional and Non-Functional requirements [17]. I understood it was not just the features of the project that were important but also the constraints. The most obvious of these was the importance that the feedback correspond to the National Fluid Prescription Guidelines. They were also ordered according to the MoSCoW framework. According to Hatton, the technique offers a “quick source of information regarding prioritisation” and gives valuable insights on what to focus on during software engineering [18]. Particularly given the time constraints on this project, I found this to be very useful.

Table 1 *F = Functional; NF = Non-Functional*

| Id  | Requirement  | Type | Module         | Category      | Priority  |
|---|--|------|----------------|---------------|-----------|
| <b>P1: How well application functions in original hospital.</b> |  |      |                |               |           |
| R1  | System will have validation logic for first and second tier feedback   | F    | MyCase         | Case Feedback | Must Have |
| R2  | System will allow System Admin to create and edit generic feedback. This will be set in a Global Setting module that will need to be created | F    | Global Setting | Case Feedback | Must Have |
| R3  | System will allow Case Administrator to add Case Specific feedback   | F    | Case           | Case Feedback | Must Have |
| R4  | System will display Generic and Case Specific feedback on completion of prescription   | F    | MyCase         | Case Feedback | Must Have |

|     |  |    |                       |                |             |
|-----|--|----|-----------------------|----------------|-------------|
| R5  | System will allow Case Administrator to add image to feedback  | F  | Case                  | Case Feedback  | Could Have  |
| R6  | System will display feedback including images added by Case Administrator to user on completion of prescription                | F  | MyCase                | Case Feedback  | Should Have |
| R7  | System will ensure that validations contain constructive feedback. These must adhere to National Fluid Prescription Guidelines | NF | MyCase                | Case Feedback  | Must Have   |
| R8  | System will give user option to provide feedback on Case when prescription has been finished                                   | F  | MyCase                | User Feedback  | Should Have |
| R9  | System will have module for Case Administrators to see feedback on their cases.  | F  | Case Feedback Module* | User Feedback  | Might Have  |
| R10 | System will allow Case Administrator to add NEWS score to the database   | F  | Case                  | NEWS score     | Must Have   |
| R11 | System will display NEWS score to users as part of case information  | F  | MyCase                | NEWS score     | Must Have   |
| R12 | System will display total potassium dose of all fluids prescribed including any potassium in prescribed additives              | F  | MyCase                | Potassium Dose | Could Have  |

### P2: The look and feel of the application.

|     |  |    |             |             |             |
|-----|--|----|-------------|-------------|-------------|
| R13 | System will generate a thumbnail of a new image added in Case Setup. This thumbnail will be added along with the original image to the Azure container | F  | Case        | Case Images | Must Have   |
| R14 | System will display thumbnail image on MyCase Index page   | F  | MyCase      | Case Images | Must Have   |
| R15 | System will allow user to view original full quality image by clicking on thumbnail image  | F  | MyCase      | Case Images | Must Have   |
| R16 | System will display Case Index page in reasonable time   | NF | MyCase      | Case Images | Must Have   |
| R17 | Front end of system will have professional feel  | NF | All Modules | Front End   | Should Have |

### P3: How well application will function at another hospital

|     |  |    |                 |                  |             |
|-----|--|----|-----------------|------------------|-------------|
| R18 | System will allow System Admins to add a new fluid to the library            | F  | Fluid Setup     | Fluid Library    | Must Have   |
| R19 | System will allow System Admins to make a fluid inactive                     | F  | Fluid Setup     | Fluid Library    | Must Have   |
| R20 | System will allow System Admin to edit composition of fluid                  | F  | Fluid Setup     | Fluid Library    | Must Have   |
| R21 | System will allow System Admin to view fluid library                         | F  | Fluid Setup     | Fluid Library    | Must Have   |
| R22 | System will allow user to select fluid from library when making prescription | F  | MyCase          | Fluid Library    | Must Have   |
| R23 | System will display only fluid name to user (not abbreviation)               | NF | MyCase          | Fluid Library    | Could Have  |
| R24 | System will allow System Admins to add a new Additive to the library         | F  | Additive Setup* | Additive Library | Could Have  |
| R25 | System will allow System Admins to make an Additive inactive                 | F  | Additive Setup* | Additive Library | Could Have  |
| R26 | System will allow System Admin to edit composition of Additive               | F  | Additive Setup* | Additive Library | Could Have  |
| R27 | System will allow System Admin to view Additive library                      | F  | Additive Setup* | Additive Library | Could Have  |
| R28 | System will display on most screen sizes                                     | NF | All Modules     | Front End        | Should Have |

\* Requirement was never implemented. Therefore, module was not created and is not part of the project.

### 3.4 Importance of Use Cases

I was focusing on how to implement the requirements rather than working out the requirements themselves. This meant use cases, with their particular focus on user-system interaction [19], appealed to me. I was also influenced by a blog post by Alistair Cockburn entitled 'Why I still use Use-Cases' [20]. He cited their importance in providing "everyone involved with an agreement of what the system will basically do". I have already discussed how important the communication between myself and Dr Davey has been in this project. Use cases allowed me to reach an understanding of the requirements and their possible implementation where I could confidently talk to him about them. Cockburn also referred to how useful they could be in offering a "look-ahead ... [to] spot issues that are likely to take a long time to get answers for". In a project with such time constraints I thought it very important to plan carefully in this way.

I looked to Jim Arlow and Ila Neustadt's book 'UML 2 and the Unified Process' [19] in informing my use case modelling. As they suggested, I started by creating a list of actors, and from here created a Use Case Diagram (this can be seen in Appendix C.3.a). This was useful in seeing how different tasks related to each other. There are three different roles a user can be in the application. There is the 'Prescription User' who can respond to cases with prescriptions. Then there is the 'Case Administrator' who can also create cases. Finally, there is the 'System Administrator' who can use all the functionality of the application including adding new fluids and changing the generic feedback.

I then started to specify the detail for each use case. There is no UML standard for use cases, so I decided to adapt the one provided by Arlow and Neustadt. They suggested that "it is best to keep use case modelling as simple as possible" [19] so I further streamlined their specification in line with my project. I had tried to split the recruitments up into fairly discrete tasks and there were no use cases with Secondary actors. Due to the way the application was split into fairly distinct modules there were also no alternative flows. I got rid of these two sections.

As Arlow and Nuestadt write “Use Case modelling is iterative” [19] and I approached the task this way. I developed the use cases as the project went on and the requirements became clearer through my discussion with Dr Davey. The detailed use cases can be seen in Appendix C.3.b

*Table 2*

| Use Case ID | Use Case                                      |
|-------------|---|
| <b>U1</b>   | AddNewFluid                                   |
| <b>U2</b>   | EditCompositionOfFluid                        |
| <b>U3</b>   | MakeFluidInactive                             |
| <b>U4</b>   | ViewFullImageInCase                           |
| <b>U5</b>   | UserReceivesTier1And2Feedback                 |
| <b>U6</b>   | UserReceivesTier3Feedback                     |
| <b>U7</b>   | AdminAddsGenericFeedback                      |
| <b>U8</b>   | SeniorClinicianAddsCaseSpecificFeedback       |
| <b>U9</b>   | UserGivesFeedbackOnCase                       |
| <b>U10</b>  | SeniorClinicianReceivesFeedbackOnUploadedCase |
| <b>U11</b>  | UserViewsTotalPrescribedPotassium             |
| <b>U12</b>  | AddNewAdditive                                |
| <b>U13</b>  | EditCompositionOfAdditives                    |
| <b>U14</b>  | MakeAdditiveInactive                          |
| <b>U15</b>  | AddImageToCaseFeedback                        |
| <b>U16</b>  | UserViewImageInFeedback                       |
| <b>U17</b>  | CaseAdminAddsNewsScore                        |
| <b>U18</b>  | PrescriptionUserViewsNewsScore                |

Using the use cases as look-aheads I got a better idea of what would be needed to implement each feature. By doing this, I was able to make another useful distinction. This was between requirements that would be heavily based on features already part of the project, and those with less of a precedent. Wanting to give the project a professional look and feel (P2), I make usability a priority in the implementation of both of these requirement sets.

Use cases U1-U3 (requirements R18-R23) demonstrated that actions needed for completion of the Fluid Setup module would be similar to those for the Case and User Setup modules. Informed by Don Norman’s design heuristics “consistency and standard” [21] I wanted to make sure their designs would cohere. The different tables in the project all perform very similar functions (for more information see the User Manual in Appendix B). It would be

strange if they looked or behaved in very different ways. I made sure I studied the existing features carefully to ensure I managed this continuity.

For use cases U6, U8 (requirements R3-R8) it became clear that new Partial Views would need to be created. These did not match any other part of the project. I needed to design a view from scratch. I would need to come up a new design consistent with good usability principles. How I did this is explained in greater depth in Section 4.3.c. All the requirements for the MyCase apart from R1 were of this type.

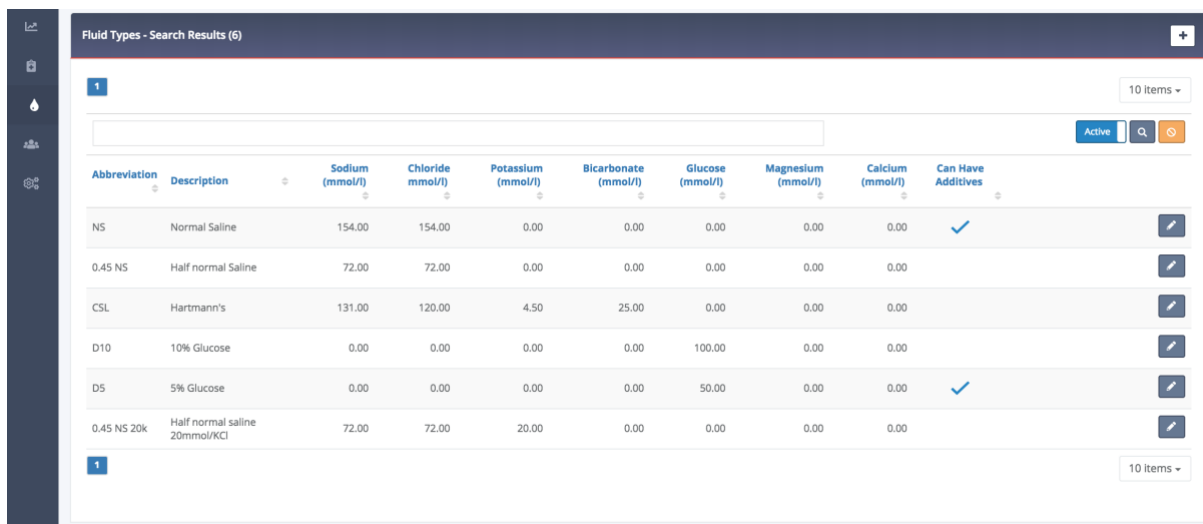
The more detailed I could be in my analysis of my requirements, the better I would understand what was necessary to implement them. The better I could understand this, the fewer errors I would make in this next stage of development. The earlier an error is caught the less time will have been wasted on it and the easier it will be to fix. The analysis spelled out above was a vitally important part of my work on the project.



## Chapter 4: Design and Implementation

## 4.1 ASP.NET Core MVC

ASP.Net Core MVC was the framework used for the project I inherited. I will give a brief explanation of this architecture using the example of the fluid table (*Figure 1*) as a simple example. The structure is explained clearly in the Microsoft Documentation [4] and I will quote heavily from this resource throughout the section. The framework has a Model-View-Controller (MVC) architecture. I will briefly explain how the Model, View and Controller is used in an ASP.NET Core MVC application.



| Abbreviation | Description                   | Sodium (mmol/l) | Chloride (mmol/l) | Potassium (mmol/l) | Bicarbonate (mmol/l) | Glucose (mmol/l) | Magnesium (mmol/l) | Calcium (mmol/l) | Can Have Additives |
|--------------|-------------------------------|-----------------|-------------------|--------------------|----------------------|------------------|--------------------|------------------|--------------------|
| NS           | Normal Saline                 | 154.00          | 154.00            | 0.00               | 0.00                 | 0.00             | 0.00               | 0.00             | ✓                  |
| 0.45 NS      | Half normal Saline            | 72.00           | 72.00             | 0.00               | 0.00                 | 0.00             | 0.00               | 0.00             |                    |
| CSL          | Hartmann's                    | 131.00          | 120.00            | 4.50               | 25.00                | 0.00             | 0.00               | 0.00             |                    |
| D10          | 10% Glucose                   | 0.00            | 0.00              | 0.00               | 0.00                 | 100.00           | 0.00               | 0.00             |                    |
| D5           | 5% Glucose                    | 0.00            | 0.00              | 0.00               | 0.00                 | 50.00            | 0.00               | 0.00             | ✓                  |
| 0.45 NS 20k  | Half normal saline 20mmol/KCl | 72.00           | 72.00             | 20.00              | 0.00                 | 0.00             | 0.00               | 0.00             |                    |

Figure 1 – fluid table

## 4.1.a Views

“Views are responsible for presenting content through the user interface. There should be minimal logic within views, and any logic in them should relate to presenting content” [4]. In figure 1 Views are responsible for the layout and display issues such as where the navigation bar is and how the form is set out. In ASP.NET Core MVC there are also Partial Views. These are Views that are rendered in other Views [22]. The fluid table in *figure 1* is made up of a series of Partial Views. A detailed description of this is provided in Section 4.4.a.

#### 4.1.b Models and View Models

The Model “represents the state of the application and any business logic or operations that should be performed by it” [4]. Here, as is often the case, there is a Model corresponding to each table in the database. In simpler applications only one kind of Model may be used. In this application View Models were also used. These handle only the data that a View needs for its display. It is not good practice for them to display business logic. ‘Create Software’ had interpreted this to allow the View Models to contain simple validations. As is explained in section 4.3.b I adhered to this understanding. The role of the Model will become clearer as I discuss the rest of the structure of the application.

#### 4.1.c Controllers

Controllers are the “initial entry point” [4] of the application and “handle user interaction, work with the model, and ultimately select a view to render” [4]. The controller connects the Model and the View together. They handle browser requests by retrieving the correct model for the View. The models used in the Views need not always be Models (as they are termed in this application). They can be any object such as a POJO (see Section 4.3.c) or even a String. For the view in *figure 1* a generic ‘BaseVM’ View Model is loaded in with a POJO of type ‘FluidTypeItem’ by the controller.

#### 4.1.d Separation of Concerns

The application is structured according to this MVC framework to achieve a separation of concerns between the three components. This means “[avoiding] co-locating different concerns within the design or code” [23]. The project, therefore, becomes “easier to code, debug, and test” [4]. If the ‘Edit Case’ page (*figure 1*) had the navigation bar in the middle of the page, it would be clear you needed to look for the error in the ‘Edit Case’ View. In addition, when you fix this error you will be much less likely to interfere with another part of the code.

## 4.2 Inherited Design

There is flexibility in how to use ASP.NET Core MVC and I had to understand how the project I inherited had used the technology. The project is split into 8 main sections, each with its own function. In the proceeding section I will go through the database and then each of these sections, detailing their role in the application. I will use examples from my work on the project to help illustrate this. In order to be confident in the changes I made, I needed to understand how all the sections worked. However, some sections were more important than others to the work that I was doing. I will sketch these out in more detail.

### *4.2.a Database*

The Database was a MySQL database hosted on Azure that I inherited from 'Create Software'. The database was created with an ORM (Object-relational mapping) model designer called Entity Developer [6]. An ORM is a tool for interacting with the database. To do this, code is written in C# using the LINQ [24] syntax. The code is converted into the necessary SQL statements by Entity Developer. If the call to the database is to return a table row, data is obtained in a form that can be stored as an Object-Orientated class. These calls to the database take place in the Persistence Layer (Section 4.2.e).

Entity Developer gives a visual representation of the database as well as tools to add and edit tables (*figure 2 - annotations added*). When changes are made to the database using

the Entity Developer interface, it is easy to right click and select the option of updating the database from these changes. It is then possible to select 'Update Storage and Mapping'. This will regenerate all the Model class and POCO

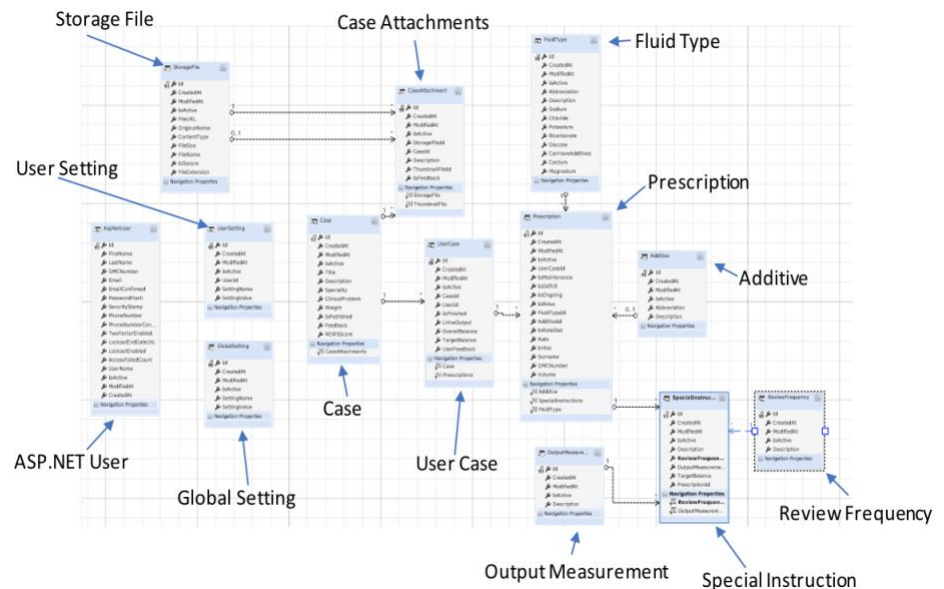


Figure 2 – Database Schema (a larger version of this can be found in the appendix A.4

class (see Section

4.2.c) files in the project to reflect the changes. These are created with corresponding 'generated' files. These are classes containing all the fields of the database table as class variables. An example of a Model and POCO 'generated' file can be seen in Appendix E.1 and E.2 respectively. When the files are regenerated in Entity Developer any changes to the .generated files will be lost. Changes made by the developer to the regular POCO or Model file will remain.

#### 4.2.b NHS.Fluid.Model

It is important to note that when I refer to Models I am not referring to View Models. These will be referred to as View Models. When I created the 'Global Setting' table using Entity Developer the Model associated with it was automatically generated. Also generated was a file entitled 'GlobalSetting.Generated'. This, as can be seen in Appendix E.1, was a class containing all the fields in the database table such as 'SettingName' and 'Setting Value'. Models can contain all the information pertaining to a row in a database table.

#### 4.2.c NHS.Fluid.Contracts

Also generated by Entity Developer when adding a new table in the database are plain old CLR objects (POCOs) found in the Contracts folder. These, like *figure 3*, also come with a .generated file containing the table database fields and can also hold data corresponding to a row in the database. Models contain data as it is in the database. POCO's are concerned by how the data will appear to the user and do not contain logic pertaining to how the data is obtained. This is the property of persistence ignorance. The POCO 'GlobalSettingItem' would not contain a GlobalSettingItem.GetByID() method. They could, however, contain data, validations and business logic. Many of the POCO's in the project contain no methods and are only used to store data.

However, some do have some logic in them such as the 'UserCaselItem' POCO. Here, the dose of different prescriptions needs to be worked out from other factors such as electrolyte content in the fluid as well as patient weight and rate prescribed. This is worked out in the class. When implementing requirement R12 to display the potassium dose adjusted for weight, I created a method that returns this value. It adds total potassium in the additives to the total potassium in the fluids prescribed (derived from variables including prescription rate and volume of electrolyte in fluid). This figure is then divided by patient weight. When this value is required in a View, it only requires calling the 'PotassiumByWeight' variable from 'UserCaselItem'. This code can be seen in appendix E.2 - UserCaselItem.

#### 4.2.d NHS.Fluid.AutoMapper

For each Model there is a corresponding POCO item. The Models are mapped onto POCO's using the AutoMapper library. This was set up before I began my work on the project. The only change I made to this section was, when creating the global setting module, adding two extra lines in order to map the 'GlobalSetting' Model onto the 'GlobalSettingItem' POCO. This can be seen in Appendix E.3 - DataTierProfile.

#### 4.2.e *NHS.Fluid.Persistence*

This section is where the logic pertaining to querying the database is found. It was created in order to compartmentalise the database. It contains methods which interact with the database using the LINQ syntax. If it was decided to change the database, the only thing that would need to be changed would be this layer (the code to interact with the database would be different).

The layer had been created pretty much in its entirety before I started the project. I did not do a lot of work on this layer. In adding the 'GlobalSettingPersistenceManager', the class simply inherited from the base class of 'Generic Persistence Manager'. This base class contains most of the methods with which to interact with the database. The only other changes I made were to the 'FluidTypePersistenceManager' and these will be detailed in Section 4.3.a of this chapter.

#### 4.2.f *NHS.Fluid.Services*

The Services layer is on top of the Persistence layer. It uses methods from the persistence layer to interact with the database and transforms the data into a form more suitable for the user. A good example of this process can be found in my implementation of requirements R13-R16 (Case Image). On this application images are saved in a container on Azure. An identifier for these images is stored in the database and they are displayed in Views by calling this identifier. The display of this image can be seen in Appendix H.8 MyCase/View (lines 96-119)

The method in the service layer takes a 'CaseAttachmentItem' (i.e. a POCO class in this case named 'item' in *figure 3* below) and checks whether the file is an image. If it is, it converts it to a smaller thumbnail image.

```
45. var storedFile = fileService.CreateSecure(file, "Cases", fileName, extension,
    contentType);
46. if (storedFile != null)
47. {
48. item.StorageFileId = storedFile.Id;
49.
```

```
50. if (newFile != null && newFile.Length > 0)
51. {
52.     var thumbnailStoredFile = fileService.CreateSecure(newFile, "Cases", fileName,
        extension, contentType);
53.     item.ThumbnailFileId = thumbnailStoredFile.Id;
54. }
55.
56. Save(item);
```

*Figure 3 – Case Attachment Service, StoreLineAttachment() Method*

In *figure 3*, on line 45, the original image is stored in the Azure container using the `CreateSecure()` method (a method created before my work on the project). The Storage File ID for 'item' is then set as the ID for that latterly stored file (line 48). There is then an 'if' statement (line 50) checking that there is a thumbnail file (here named 'newFile'). If there is, the procedure just described is repeated for the thumbnail (there is a separate field for the thumbnail file Id; line 53). Finally, the save method (from the 'BaseServices' file which was also present before my work on the project) is called to save the 'CaseAttachmentItem' ('item') to the database (line 56).

```
persistedItem = Mapper.Map<TEntityItem, TEntity>(item);
return Mapper.Map<TEntity, TEntityItem>(persistenceManager.Create(persistedItem));
```

*Figure 4 – Base Services Save() Method*

*Figure 4* shows the code from the save method tasked with saving the item to the database. The Mapper detailed in Section 4.2.d is used to map a 'CaseAttachmentItem' POCO onto a 'CaseAttachment' Model. This is then set as 'persistedItem'. The `Create()` method is then called from the Persistence layer and a new row is created in the database corresponding to 'persistedItem'.

Here, we can see how the Services Layer takes the data from the user then changes it into a format that can be saved in the database. Then, a method from the Persistence Layer is called to save this object in the database. We can also see the different roles that POCOs and Models play. POCOs are what are passed into and returned by Service Layer methods and are normally used in the Controller. They are normally populated with data inputted in

the Views (this is certainly the case above). It is the content of Models that are eventually added to the database by the Persistence Layer.

#### *4.2.g NHS.Fluid.PersitenceSupport*

The project uses the dependency injector Ninject. Ninject is a tool that reduces hard-coded dependencies between classes. Hard-coded Dependencies are when a class instantiates a concrete implementation of another class. In dependency injection classes will instantiate an interface of the other class. The concrete implementation of this interface will be passed in using the constructor. This means that code using the original class can decide which concrete implementation of the interface to pass in. Ninject manages which concrete implementation to pass into the constructor. For this to happen, classes need to bound to their interfaces and this takes place in the PersistenceSupport section of the project. This area of the application was not part of my work on the project. However, I did need to understand how it worked in order to be able to create classes that were consistent with Ninject.

#### *4.2.h NHS.Fluid.Core*

This is a section where various helper methods related to the project are kept. This includes files such as 'RadndomPassword.cs', used to generate a random password, and 'Converter.cs', in which the date is converted from a timestamp. It was not a section that I added to and I only very rarely used methods from it.

#### *4.2.i NHS.Fluid.WebApp*

This folder is where bulk of the project is situated and includes the Controllers, Views, View Models, Scripts and CSS files etc.

### *4.3 My Changes*

I started to make my changes to the project according the design plan outlined in Section 2.3. It is not always easy to determine what code I added and what code was there before



me. There is not enough space in this section to describe, in detail, all the changes I made. I have therefore included a table in the Appendix (C.5) in which I detail all the code that I added. It should be possible to get a fairly clear idea of all the changes I made to the project from here. I have also added a representative selection of my code in Appendix E of this report. In this section, I will outline how I applied my design plan. I have tried to include examples of the code I added that are informative of key stages of my work.

#### *4.3.a Inflexibility of Technology and Deadline*

Detailing the structure of the project in the above section was to provide context so as to better understand the changes that I made in the project. It was also to emphasise how important Section 2.3.a was in my design plan. The project was complex and encompassed concepts I, previously, had no understating of. The time I spent carefully studying the technologies and project structure was incredibly beneficial. After 2 weeks I had made a lot of progress in my understanding of the system. However, I felt that to really grasp it I would need to make changes to and interact with the code. I decided to start, carefully, implementing requirements. I made use of GitHub, so I could safely undo any mistakes I made.

I chose requirements R16 to R19 (the fluid prescription module) to work on first; it made sense to start with 'Must Have' requirements. The requirements were based heavily on already existing parts of the project (more detail in Section 3.4). Because of the time I spent studying the User Setup and Case Setup modules (to which the Fluid Setup module was very similar), I was able to ensure that my additions cohered with the design of these other modules. The advantage of starting with these requirements was that I had a template to work from as I worked on this area. This was particularly useful as I worked on the more complex layers of the project.

First, I needed to create a table to display the fluids. It was necessary to understand the role of View Models and POJO items. The table was created in the '`_FluidTypeTable`' Partial View (all Partial Views in this project begin with an underscore) which was itself rendered in the 'Index' file for Fluid Type.

```
30. <tbody>
31. @foreach (var item in Model.Items)
32. {
33. Model.RowVM = new BaseRowVM<FluidTypeItem>();
34. Model.RowVM.Item = item;
35. Html.RenderPartial("_FluidTypeRow", Model.RowVM);
36. }
37. </tbody>
```

*Figure 5 – Fluid Type Table Partial View*

Each table in the project used the 'BaseRowVM' and 'BaseTableVM' files. They are generic classes, so it is necessary to specify the data type they are using. They are both files I inherited from the previous developer. *Figure 5* shows how the fluid table is created by a loop (line 31). The loop is over a list of Model.Items from the 'BaseTableVM' View Model (the model passed into the '\_FluidTypeTable' Partial View). 'Items' has previously been set as a list of 'FluidTypeItem' POCOs corresponding to each fluid in the database.

On each iteration of the loop the 'RowVM' is set to a new instance of 'BaseRowVM' which is itself set to type 'FluidTypeItem' (a POCO; line 33). The variable of 'Item' for 'RowVM' is then set as the current iteration of 'item' (line 34) this is a 'FluidTypeItem' POCO (from the list Items corresponding to the fluids in the database). The Partial View '\_FluidTypeRow' is then rendered with the View Model 'RowVM' passed in as the model. '\_FluidTypeRow' is composed of a single row of a table (for code see appendix E.8 \_FluidTypeRow). On each column it displays an electrolyte content. This is obtained from the 'RowVM' variable 'Item' (set on line 34).

When finishing the requirements, I manual tested the features before I uploaded to the code to GitHub. I noticed the table on the Fluid Index View could not be searched. In this project, the search function works by querying the database for any fluids that correspond to that search. It is more efficient to return only the fluids needed for the search rather than return all the fluids in the database, and then only display the ones needed.

Understanding this, I knew that it would be the Persistence Layer that I would need to

adapt. I added the search method to this layer. The method was composed by calling a number of other methods written before my work on the project. I was able to work from very similar methods present in the 'User Setup' and 'Case Setup' modules. This method can be seen in Appendix E.3 - FluidTypePersistenceManager.

These are both good examples of having to understand and use classes I had not created myself. It was important to have studied the project to be able to do this. However, I also found that starting to add features to the project slightly earlier than planned in my design strategy was a good alteration on my initial plan. It gave me a level of understanding I would not have had without some interaction with the code. Also, successfully implementing this feature gave me confidence in my understanding of the project.

#### *4.3.b Implementation of Prescription Validation Logic and Iterative Approach*

The implementation of the Case and User Feedback parts of this project had no template. Requirement R1 mandated adding validations to user prescription inputs. Ensuring I adhered to View Model conventions as well as the structure of the rest of the project, I only put simple validations in the 'PrescriptionViewVM'. The more complex validations were put in the 'MyCase' Controller. The 'PostPrescriptionPartial' method on the 'MyCase' controller is responsible for taking the input in the '\_PrescriptionEditor' Partial View as a 'PrescriptionViewVM' View Model and saving it to the database using a Service Layer method.

```
262. var myCase = userCaseService.GetInclude(model.UserCaseId);
263. var currentCase = caseService.GetInclude(myCase.CaseId);
264. if (currentCase.NEWScore > 4 && !(model.Volume <= model.Rate / 4))
265. {
266.     ModelState.AddModelError("CustomError", "If Bolus is selected and NEWS score is 5
        or above, rate must be such that the patient receives the total volume in 15 minutes or
        less.");
267. }
```

Figure 6 - 'MyCase' Controller, 'PostPrescriptionPartial' method

*Figure 6* shows a validation from this method that requires the NEWS score of the case being worked on. The score must be obtained from the 'Case' table in the database. In the Controller, interfaces for the 'UserCase' ('userService') and 'Case' ('caseService') services are instantiated by the constructor (this can be seen in Appendix E.6 – MyCaseController (lines20-43)). On line 262, GetInclude() (a method from the 'userService' that can be seen in full in Appendix C.6) is used to create a 'UserCaseItem' POCO, named 'myCase', corresponding to the user's current prescription (found in the 'UserCase' table in the database). This is done by passing in the 'UserCaseId' from the 'PrescriptionViewModel' which contains the inputs from the View.

On line 263 'CaseId' (the ID corresponding to the Case that the prescription is for) is obtained from 'myCase' and then used to create a 'CaseItem' POCO (this time using the 'caseService') named 'currentCase'. The 'currentCase' will correspond to the case, set by a case admin, that the user is creating a prescription in response to. The 'PrescriptionViewModel' does not have a variable for the Case ID, which is why the 'currentCase' object needs to be created. From here, on line 266, the NEWS score (from the 'currentCase' object is checked in the 'if' statement. If the NEWS score is higher than 4 and a quarter of the model rate is not higher than or equal to the prescription volume (line 265), the validation will be displayed.

This is why the project conventions were adhered to by splitting the validations between the Controller and View Models. For the validations in the Controller, the service layer interfaces are already instantiated using the constructor. To do this in the View Model would have been to use code concerned with obtaining data. View Models must only contain code concerned with how the information is to be displayed. The validations in the View Models are confined to just those that iterate over inputs from the View i.e. there is no need to search tables from the database.

Section 2.3.d explains the iterative approach I took to this part of the project. An example of this was when, towards the end of the project, Dr Davey spotted that the contents of the validations were correct, but they were displayed in response to the wrong inputs. More

details concerning this problem can be found in Appendix C.7 but these aren't necessary for the example.

Fixing this required rewriting all the validations. To do this I decided to use GitHub. After having finished rewriting the validations, I used the versioning feature in SmartGit to view the validations as they had been before the changes I made. I went through each one ensuring that their content had been preserved. On completion, the project was again deployed to Azure and evaluated by Dr Davey. As an important and complicated part of the project, the guidelines needed to be treated carefully. This use of GitHub combined with frequent iterative evaluation by Dr Davey ensured this was the case.

#### *4.3.c Creation of Tier 3 Feedback and User Feedback Section*

In Section 3.4 I stated how the Prescription Feedback section did not have another feature on which it was based. This meant I had more freedom in designing it. Wanting to maintain a high level of usability for the project, I turned to Don Norman's 10 usability heuristics [21]. It is important to note that despite the added freedom I had, there were still existing design conventions in the project that I had to adhere to. Norman's heuristic of "Consistency and Standards" was still important. Throughout the project, the sections were split into their own subtle boxes. I did the same for the sections in the feedback. The feedback images were also displayed according to project convention (*figure 7*).

Placing the feedback on the same page as the rest of the prescription also ensured that the user would always be aware of where there were "emergency exits" [21] to leave the page they were on (the "User Control and Freedom" heuristic [21]). The same navigation bar would always be on the left. Norman's heuristics the "Visibility of System Status" [21] was also adhered to. The prescription page followed a clear and definite path. Each time part of the prescription was finished the next part would be displayed below.

Finally, the "Aesthetic and Minimalist Design" [21] heuristic was relevant. The information displayed comprises only the necessary information to understand what is being displayed. All these principles can seem obvious or intuitive, but it is possible to be overly familiar with

a project and make design mistakes. In addition, these sections are an important part of the project. Without good feedback, the application would not teach junior clinicians. Also, if the user was frustrated with the case, it was very important they would be able effectively vocalise this. Any mistakes here could have big impact on the user's satisfaction with the application. I kept the heuristics at the forefront of my mind to avoid falling into this trap.

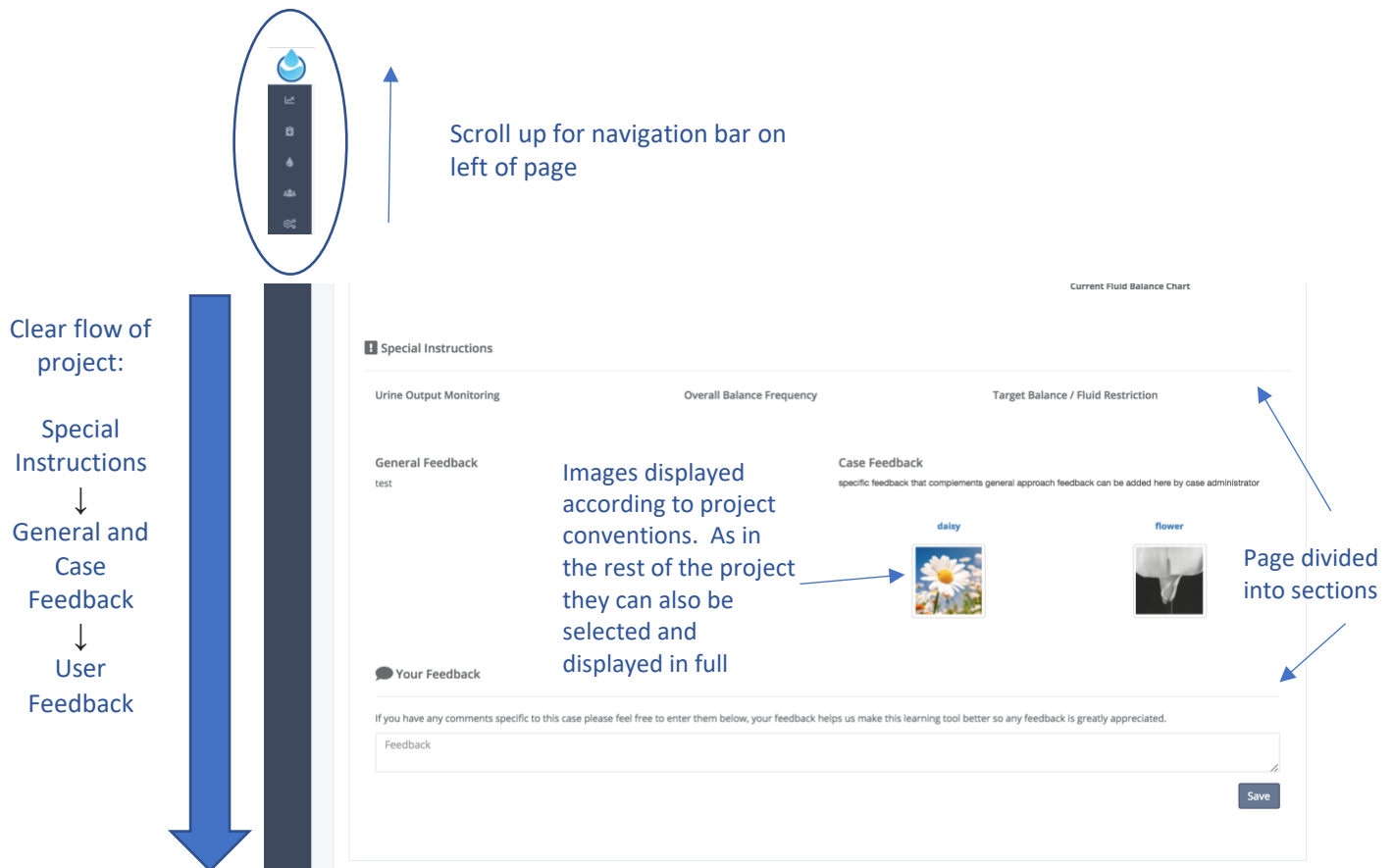


Figure 7 – Front-End Tier 3 Feedback and user Feedback

#### 4.3.e Security

Whilst security on the application was important, improving it was not in the requirements I was given. This meant that for my work on the project I only needed to understand how the existing security functions worked so that I could follow the standards set by the previous developer. The application used the ORM 'Entity Developer' and the ASP.NET Anti-Forgery-Token.

## Chapter 5: Testing

### 5.1 Testing Aims

The most important aim in my testing strategy was to ensure that the requirements that I had implemented were completed in their entirety using reliable code. If there were still any flaws with the requirements, I wanted them to be documented for the next developer. The other aim was to hand the project over to the next developer with a series of automated tests. When I started the project, there were no automated-tests. I think that, had there been tests, I would have felt more confident in experimenting with the code that I added. I would have been more confident that I would spot any errors I made.

### 5.2 Constraints on Testing Strategy

The most important constraint on my testing strategy was time. This stemmed partly from my need to spend a lot of time studying the ASP.NET Core MVC framework before my work began. The Mid-July deadline outlined in Section 2.3.b also played a role here. This meant I decided to leave testing until after a working version of the application had been delivered. By the time I had delivered all the 'Must Have' requirements, I did not have a huge amount of time left and I decided to forgo a unit-test-approach. This would have been my ideal strategy because it is so conducive to writing clearly defined code and would have given me greater confidence in changes I made. However, it was also another technology with a potentially steep learning curve that I had no experience with.

The other constraint came from the complexity of the project. It is useful to have a testing database separate from the main one. Some of the tests will inevitably add to, update, or delete from the database. It is important this doesn't affect important data. However, I had inherited the project with the main database already attached. Attaching another one for the purposes of testing would have been a task I would have needed to dedicate careful work to. I started testing late in the project and therefore decided not to spend time on this and instead work with the main database. I will detail in the next section why this was not as much of a drawback as it might initially appear.

### 5.3 Automated Testing

For my automated testing, I chose to use Selenium IDE [25]. This is a program that allows users to create functional tests. In these tests, inputs are provided to the front end of the application. The output from these inputs is then compared to a desired output. This is black-box testing because it is only concerned with the internal parts in as much as they are providing the outputs that will be tested. It will not be testing individual methods within the code. Selenium IDE was particularly useful because of its shallow learning curve. It is possible to write rough tests simply by recording the user running through the application on a computer. From here it is easy to learn how to edit these recorded actions into effective tests. Using Selenium IDE, I was able to start writing test cases almost immediately. See Appendix D.3 for an example of a Selenium IDE test I wrote.

My decision not to create a dedicated testing database was aided by the fact that the most important module to test, the 'MyCase' module, was much less affected by this constraint. The module was important because it contained the logic that dealt with, and gave the appropriate feedback to, user prescriptions. It was the most important part of ensuring P1 i.e. that the application function (Section 3.1). Its only interaction with the database is the creating of prescriptions. Because there was no delete function, there was no danger of deleting important information. In addition, anything added to this module would only be displayed on my account. This was not true for the Fluid Setup and Case Setup modules in which anything added would be displayed to every user on the application. In the most important module, interactions with the database were much safer. This allowed me to create a full set of automated tests. In doing so I tried to pay attention to edge cases. For example, in test suite TS6 I ensured that the input would work with leading 0s and would not work with a negative value.

Because of the constraints mentioned above the other modules did not have such comprehensive automated tests. The impact of this was mitigated by two factors. In Section 3.4 I wrote about how some requirements were based heavily on features already present in the project. I felt more confident in ensuring these requirements were



functional. In effect, I had templates to work from. When implementing the table seen in Section 4.3.a I could look at the way that the tables were structured in Case and User Setup Modules. The added challenge that came with these requirements was to ensure that they adhered with the design conventions of the rest of the project (more details on this can be found in Section 3.4). It would never be possible to check this with automated tests.

However, comprehensive testing was still important for these modules. My strategy for these modules was to do negative tests when something needed to be added or updated. These would ensure that if the wrong inputs were added, the correct validations would be displayed, and the item would not be added or updated. The tests in the test suite TS1 (*table 3*) all make sure that certain inputs will not allow the creation of a fluid. I did manual tests to ensure that items could be added or updated.

*Table 3* shows all the Selenium IDE tests that I created. They are split into test suites, each of which are used for different parts of the project. For more detail see Appendix D.1 where there is a table detailing how each requirement was tested. This includes the manual tests executed.

*Table 3*

| ID         | Test Suite            | Tests  | Requirements    |
|------------|-----------------------|--|-----------------|
| <b>TS1</b> | Fluid Library         | View Fluid Library<br>Fluid Big Number<br>Fluid Non-Number<br>Fluid Very Big Number<br>Fluid-ve Number | R19             |
| <b>TS2</b> | Prescription Tier 3*  | Prescription Totals and Tier 3   | R5-R8, R10, R20 |
| <b>TS3</b> | Prescription Additive | Additive Correct Volume<br>Additive with Bolus<br>Additive Correct Fluid                               | R1, R2          |
| <b>TS4</b> | Prescription Bolus    | Bolus Sodium<br>Bolus 15 mins<br>Bolus Volume<br>Prescription -ve Num<br>Bolus Rate > 1000             | R1, R2          |
| <b>TS5</b> | Prescription O/D/M**  | O&D without M<br>Rate < 84<br>M without O and D<br>Nothing Checked                                     | R1, R2          |

|            |                         |  |          |
|------------|-------------------------|--|----------|
| <b>TS6</b> | Prescription Edge Cases | Prescription -ve Num<br>Prescription very large Number<br>Prescription Leading 0s<br>Prescription Non-Number       | R1, R2   |
| <b>TS7</b> | Thumbnail Image         | Display Case Thumbnail   | R12-R13  |
| <b>TS8</b> | NEWS Score              | NEWS Score too High<br>NEWS Score Negative<br>NEWS Score leading 0s<br>NEWS Score Non Number<br>NEWS Score Present | R10, R11 |

\* This single test case is responsible for so many requirements due to the necessity to make new prescriptions for each test case. As stated above, this is not a massive problem as the test cases will only show up on my account. However, it is still good to avoid adding new items to the database as much as possible.

\*\* Fluid types: O (Ongoing), D (Deficit), M (Maintenance)

#### 5.4 Manual Testing

I used extensive manual testing throughout the project. I manually tested code each time I committed to GitHub. I also used it to complement my testing with Selenium IDE at the end of the project. My manual tests were informed by my requirements and, in particular, my use cases. Each time I completed a feature I would run through the use case associated with it. Whilst doing this, I would try to input boundary cases such as very long strings or unfamiliar characters. Because so much of this project was spent on understanding the technology I spent more time implementing fewer features than I would have done were I starting from the ground up. Fewer features meant I had time for effective and extensive manual testing when adding new features.

Manual testing was also used for the features (mentioned above) that proved difficult to test with Selenium IDE. Generally, these were quite simple. In many cases, they required testing the positive to a negative test I had used Selenium IDE for. For example, I manually tested successfully adding a new fluid to the database. I then deleted the new fluid directly from the database. I did the same for editing the fluid.

## Chapter 6: Conclusions

### 6.1 Goals and Aims

The aims of the project were dealt with in the design strategy. A detailing of how I used these to create, and then successfully implement, a design strategy can be found in Sections 2.3 and 4.3 respectively. In evaluating the overall success of my project, I will now go through the goals I set out at the start of my work.

#### *6.1.a Ensuring the Project was Completely Functional*

I completed the feedback element of the project (R1-R7). The validations were in line with National Fluid Prescription Guidelines and at a level of detail that Dr Davey was satisfied with. I have discussed the addition of the potassium dose display in Section 4.2.c (R12). The addition of the NEWs Score was simply a matter of adding a field to the 'UserCase' database table and displaying it on the prescription description page (see Appendix E.9 MyCase/View(lines48-53) for code).

The only part of this section I did not complete was the User Feedback section. I did ensure that users would have the experience of being able to give feedback on the case (R8). This was the most important part of the feature; it was a point in the application that was thought to be a key part of the project flow. If users were frustrated, but unable to vent their frustration, they may develop a negative view of the application. The feature had to be part of the project when it was user-tested by clinicians.

I did not create a means by which Case Administrators could see the feedback that users had submitted on their case (R9). This would have required the creation of an entirely new module in the project and thus had always been a 'Might Have' requirement. It was thought this would also be slightly less important for testing. There would be fewer Case administrators, so it could be explained to them that this feature would become part of the project. It would also be possible to show them their feedback using an application such as

Microsoft SQL Server Management System. I did, therefore, create a functional application ready for the extensive user-testing the project will now undergo.

### *6.1.b Improving the Look and Feel of the Application*

This was also a success. The implementation of the Case Images Section was accomplished (R13-R16). Ensuring the system would have a professional feel (R17) was slightly more difficult to evaluate as it is a much more general goal that encompasses the entire project. I believe that the features I added that needed to cohere with other, similar and already present, parts of the project were implemented in a professional way. I think the Fluid Setup and Global Setting Modules (R18-23, R2) adhere to the design conventions of the 'Case Setup' and 'User Setup' modules (see Appendix C.8 for examples).

The other addition that needed careful consideration, aesthetics-wise, was the display of Generic and Case Specific as well as the User Feedback form (R5, R7-R8). I have discussed my work on this in Section 4.3.c. One improvement I would have made is to pay closer attention to Don Norman's feedback heuristic [21]. At the moment there is no confirmation that feedback has been saved. This is something that must be addressed eventually but will not prelude successful testing in the hospital.

### *6.1.c Ensuring the Project can be Scaled up for use in other Hospitals.*

This goal could never have been accomplished in its entirety. It is impossible to know all the different features that different hospitals might require. My goal was just to demonstrate the possible flexibility of the project. Requirements R18-R23 (a system for adding new fluids to the database) were all implemented. I did not have time to implement requirements R24-R28 (a system for adding new additives to the database), but these were labelled by Dr Davey as low priority. There a much smaller variety of additives compared to fluids in Northern Irish hospitals. I also spent time using 'inspect' in chrome and testing the application on different screen sizes (R28). Whilst the application is not quite as easy to use on much smaller screen sizes (e.g. Galaxy S5 or iPhone 5) the functionality of the application had been preserved. Particularly for screen sizes of the iPhone X and above, the user will

barely have any difficulty using the application on a small screen. See Appendix C.9 for a good example of how the application works on different screen sizes.

#### *6.1.d Creating set of Automated Tests for the Project*

This part of project was a qualified success. I did manage to create a set of Selenium IDE tests for the project. However, these were not as encompassing as I had intended (more detail in Section 5.3.a). I did manage to rectify this with manual testing. I believe that, whilst I was not able to create a comprehensive set of tests for the next developer (as would have been ideal), I did implement a testing strategy that ensured that the code I handed over was reliable. At the end, I did a final run through of testing and Appendix D.2 shows the errors that were found by doing this and how I responded to them. It should also be noted that handing over the project with comprehensive tests was my own plan rather than something I had been asked to do. From Dr Davey's perspective, the important thing was that the code was reliable.

I also do believe that my tests on the 'MyCase' module were comprehensive enough to be used by another developer. The drawback would be if they wanted to change the content of the validation messages. This would fail the Selenium verify statements. However, if they just wanted to ensure user inputs were outputting correct validation messages, my tests would be useful. As stated in Section 5.3, the 'MyCase' section was probably the most important part of the project and I think my tests for this section could be used in the future.

#### *6.2 Mid July Deadline*

I have written about the Mid-July Deadline provisionally set for the completion of the coding part of this project. In a fashion, the deadline was met. Even though I was not able to complete all the prescription logic in the 'MyCase' module by mid-July, I was able to address goals P2 and P3 (Section 3.1). Dr Davey would have been able to demonstrate a substantially improved application by this deadline. As I have stated in Section 2.3.b, it was not a hard deadline and it always would have been unrealistic.

### 6.3 Critical Evaluation

A change that I would have made to my approach was to try to finalise the feedback requirements before I started the implementation process. I have stated in Section 4.3.b that some flexibility was advantageous to the project. However, the process of showing Dr Davey new interpretations of the requirements and receiving feedback did drag on too long and left me with even less time for automated testing than I would have had. I believe that, had Dr Davey and I sat down at the start and talked in detail about what we would and would not implement, some of this delay would have been lessened. We would still have developed them iteratively, but perhaps with fewer iterations.

The other way I could have approached this would have been to do fewer requirements. As well as fulfilling all the 'Must Have' and 'Should Have' requirements, I also fulfilled a number of 'Could Have' requirements. R6 was, in particular, fairly time consuming. I would have preferred a more comprehensive testing strategy to this requirement. However, I think this is a value judgement, and perhaps not something Dr Davey would have agreed with. In addition, I don't think I ever would have had time to use a more complex piece of technology for my testing. I think the biggest improvement I could have made would have been to free up enough time to create a testing database.

In an ideal world, I believe that the biggest addition to the project would have been the use of automated-user-testing from the beginning of the project. It would have been useful for the next developer and would have further improved my confidence in my own code. Sadly, with the time constraints this was not realistic. I consider this project to have been a success with this qualification.

## Bibliography

- [1] "nhs.co.uk," [Online]. Available: <https://www.nhs.uk/news/medical-practice/nice-issues-warning-over-dangerous-iv-drip-use/>. [Accessed 01 September 2018].
- [2] "england.nhs.uk," [Online]. Available: <https://www.england.nhs.uk/ourwork/ltc-op-eolc/sepsis/news-frequently-asked-questions/#1-what-is-a-national-early-warning-score-news>. [Accessed 01 09 2018].
- [3] "Lynda.com," [Online]. Available: <https://www.lynda.com/>. [Accessed 01 09 2018].
- [4] "Microsoft.com," [Online]. Available: 17. <https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-2.1>. [Accessed 01 09 2018].
- [5] "GitHub," [Online]. Available: <https://github.com/>.
- [6] "syntevo.com," [Online]. Available: <https://www.syntevo.com/smartgit/>. [Accessed 01 09 2018].
- [7] "Visual Studio Community," [Online]. Available: <https://visualstudio.microsoft.com/vs/community/>.
- [8] "adtmag," [Online]. Available: <https://adtmag.com/articles/2017/05/10/vs-for-mac.aspx>. [Accessed 01 09 2018].
- [9] "Microsoft - IIS 10.0 Express," [Online]. Available: <https://www.microsoft.com/en-us/download/details.aspx?id=48264>. [Accessed 01 09 2018].
- [10] "microsoft - SQL Server Management Studio," [Online]. Available: <https://www.microsoft.com/en-us/download/details.aspx?id=48264>.
- [11] "Apple Boot Camp," [Online]. Available: <https://support.apple.com/en-gb/HT201468>. [Accessed 01 09 2018].
- [12] "Virtual Box," [Online]. Available: <https://www.virtualbox.org/>.
- [13] "Wine," [Online]. Available: <https://www.winehq.org/>. [Accessed 02 09 2018].
- [14] "Macworld," [Online]. Available: <https://www.macworld.co.uk/feature/mac-software/best-virtual-machine-software-3671133/>. [Accessed 01 09 2018].
- [15] "Youtube," [Online]. Available: <https://www.youtube.com/watch?v=rNV5JmxtKP4>. [Accessed 01 09 2018].
- [16] "Interaction Design Foundation," 01 March 2018. [Online]. Available: <https://www.interaction-design.org/literature/article/designing-for-the-mobile-environment-some-simple-guidelines>. [Accessed 02 September 2018].
- [17] Y. Rogers, H. Sharp and J. Preece, Interaction design: beyond human-computer interaction., John Wiley & Sons, 2011.
- [18] S. Hatton, Choosing the right prioritisation method in Software Engineering, 19th Australian Conference , 2008.
- [19] J. Arlow and I. Nuestadt, UML 2 and the Unified Process., 2nd ed. Pearson India., 2005.
- [20] a. cockburn, "alistair.cockburn," [Online]. Available: <http://alistair.cockburn.us/Why+I+still+use+use+cases>. [Accessed 01 09 2018].
- [21] D. Norman, "nngroup," [Online]. Available: <https://www.nngroup.com/articles/ten-usability-heuristics/>. [Accessed 01 09 2018].

- [22] "Microsoft ASP.NET Views," [Online]. Available: <https://docs.microsoft.com/en-us/aspnet/core/mvc/views/partial?view=aspnetcore-2.1>.
- [23] "deviq," [Online]. Available: <https://deviq.com/separation-of-concerns/>. [Accessed 01 09 2018].
- [24] "Docs Microsoft LINQ Syntax," [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/query-syntax-and-method-syntax-in-linq>. [Accessed 01 09 2018].
- [25] "Selenium," [Online]. Available: <https://www.seleniumhq.org/projects/ide/>. [Accessed 01 09 2018].



## Appendix

### Appendix A: System Manual

#### A.1 Database Firewall

In this section I will detail how to run the code I have submitted on a local machine. I also will include instructions on how to set up VirtualBox virtual machine in order to run the code on a non-Windows environment. However, in order to successfully run the code, it is necessary to connect to the database. The Azure database used has a firewall which prevents access to the database for unknown IP addresses. Because this project was started by 'Create Software' I do not have access to the Azure account. In order to add an IP address to the firewall it is necessary to contact Simon Smith at 'Create Software'. On the other hand, it is possible to access the current version of the deployed code at:

<https://www.efluidchart.com/>

#### A.2 Virtual Machine Setup

ASP.NET Core MVC is a framework that works best on a Windows operating system. If the reader is working on a non-Windows environment this section will outline how to setup VirtualBox, a virtual machine.

##### *A.2.a Create Virtual Machine*


1. Download the program at this link:  
<https://www.virtualbox.org/wiki/Downloads>
2. Download an ISO file for Windows 10. This can be found at this link:  
<https://www.microsoft.com/en-gb/software-download/windows10ISO>
3. Boot up VirtualBox and click on the 'New' button.



4. Enter the name of the system. It is recommended to call the virtual machine Windows 10. Virtual box will then automatically fill in the other fields on this page.

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

Type:  

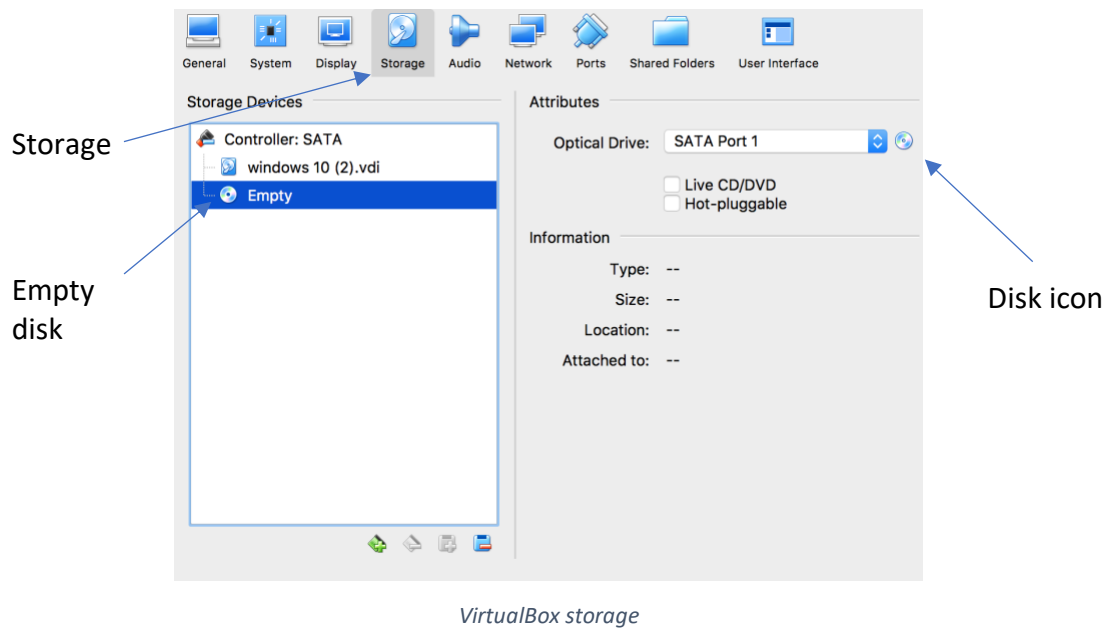
Version:

*Set name of system*

5. Set the memory size. For this project the memory size was set as 4096 MB.
6. Create a virtual hard disk. VDI (VirtualBox Disk Image) should be selected for the file type.
7. Select how the memory will be allocated. For a better performance select Fixed Size. Select 64 GB for the hard disk size and click create.

### *A.2.b Configure Virtual Machine*

It will take a while to create the virtual machine. On completion, select the virtual machine in the menu on the right and click the settings button (see the *VirtualBox home screen figure*). This will display the *VirtualBox storage* pictured below. Next, select storage from the bar at the top.



From here select the empty disk in the menu on the left. Then, select the Disk icon pictured on the right. This will open a menu from which it will be possible to select the copy of Windows 10 previously downloaded. Finally, click ok. After this select the 'Start' button from the VirtualBox home screen. This will boot up the virtual machine. From here, follow the Windows 10 start up instructions. If you do not have a Microsoft account, you will need to create one.

### A.3 Setup from Windows Environment.

Once the virtual machine has been set up the installation process is the same whatever the underlying technology.

1. Ensure that ISS 10.0 Express is installed. This can be checked in Apps & Features.
2. Install Virtual Studio Community. This can be found at this link:  
<https://visualstudio.microsoft.com/vs/community/>
3. Boot up Visual Studio and install the workloads:
  - Universal Windows Platform development
  - .NET desktop development
  - ASP.NET and web development
4. Download and Install SQL Server Management Studio 17.7 from this link:  
<https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-2017>

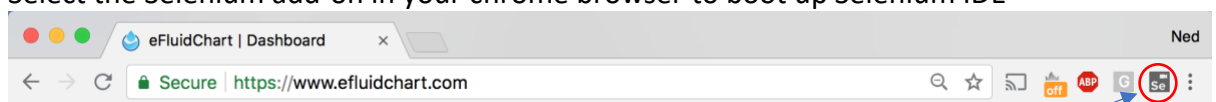
5. Download and Install 'SmartGit'. This will require a licence. In this project it was provided by 'Create Software'
6. Download and Install 'Entity Developer'. This will also require a licence that was provided by 'Create Software for this project. It can be found at this link:

<https://www.devart.com/entitydeveloper/download.html/>

Once these steps have been completed it is simply a question of opening the code, contained in the zip file uploaded with this project, in Visual Studio. In order to deploy the application on your local machine simply select 'IIS Express' from the Visual Studio menu with your chosen browser.

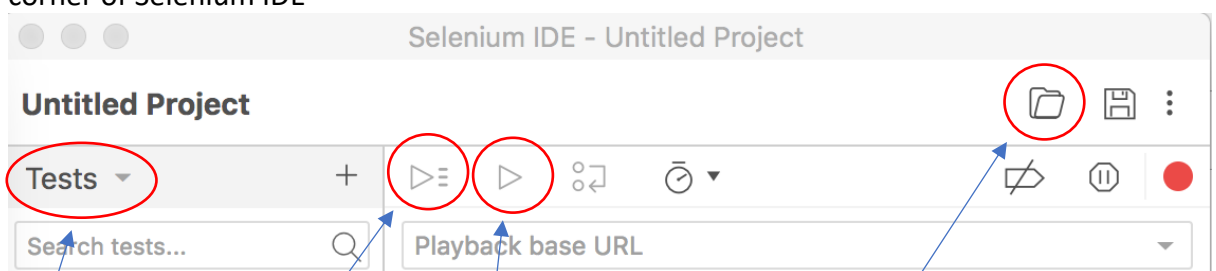
#### A.4 Run Selenium IDE tests

1. Download and Install Selenium ID found at this link
2. [https://www.seleniumhq.org/download/#selenium\\_ide](https://www.seleniumhq.org/download/#selenium_ide)
3. Either run the code on a local machine or use the Azure WebApp using the link at the head of this chapter.
4. Select the Selenium add-on in your chrome browser to boot up Selenium IDE



Selenium IDE add-on in Chrome browser

5. Open tests file (uploaded with this project) by selecting open icon in top right-hand corner of Selenium IDE



Select to change view to test suites

Play test suite

Play single test

Selenium IDE add-on in Chrome browser

6. Tests can then be played:
  - a. As single tests by selecting the 'Play single test' button

- b. As test suites. This is done by selecting the 'Tests' button on the left and selecting from the drop-down menu 'Test Suites'. Then select a test and click the 'Play test suites' button.

If the tests are green it means they have passed. If they are red it means they have failed.

## Appendix B: User Manual

### B.1 URL

The application can be found at:

<https://www.efluidchart.com/>

### B.2 Application Roles

There are three user roles in the application. These are:

1. Prescription User: Users can respond to cases by creating prescription and receiving feedback.
2. Case Administrator: Users can do all the above as well as creating cases.
3. System Administrator: Users can do all the above as well as adding and editing users and fluids. They can also edit the generic feedback.

### B.3 Toolbar:

All modules can be accessed by clicking on the toolbar on the left. The title of each module is displayed by hovering the mouse over the toolbar icon.

### B.4 Login

The application does not have a feature for users to create their own account. Instead the admin can create a new account with the new user's email address. The new user will be sent an email inviting them to activate their account.

### B.5 Dashboard

The dashboard is split into two sections. The Case Library displays the active cases in the database. Each can be opened by clicking on the magnifying glass icon on the right of the table. Opening the case brings the user to the MyCase section for that case.

The My Prescription Activity section below is a record of all the prescriptions that the current user has attempted. Completed and Uncompleted prescriptions are displayed. The uncompleted ones can be opened (using the same magnifying glass icon) and completed. The completed ones can be opened and reviewed.

## B.6 MyCase

### B.6.a MyCase index

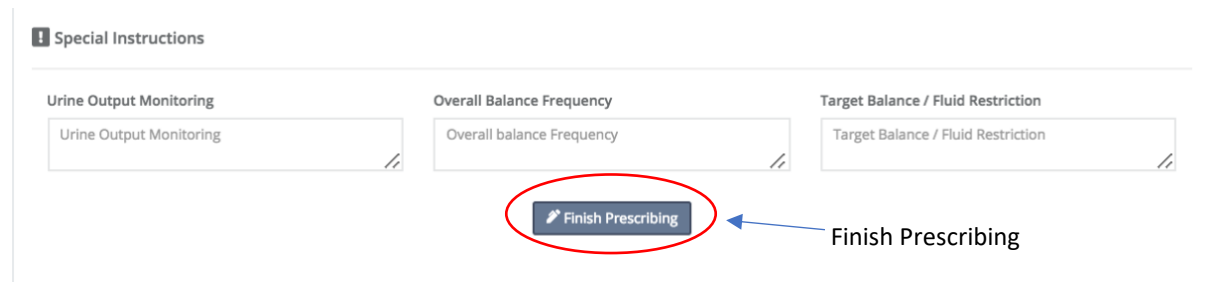
Users can respond to cases with prescriptions. After selecting a case from the dashboard, the user will arrive at the MyCase index page. The user will be able to view the case details (NEWS score, narrative, patient weight etc.) and then can select 'Create Prescription' to go to the MyCase view page.

### B.6.b MyCase View

Here, the user can create a prescription by filling in the various fields in the 'Add to Fluid Balance Chart' section. First and second tier feedback will appear as validations in response to these user inputs. The user will then select 'sign prescription' to complete this stage.

The screenshot displays the 'Fluid Chart' interface. At the top left, there is a water drop icon and the title 'Fluid Chart'. Below this, a button labeled 'Add to Fluid Balance Chart' is circled in red. A blue arrow points from the text 'Create Prescription Section' to this button. The main area contains several input fields: 'Indication' with buttons for '(M)aintenance', '(O)ngoing', '(D)eficit', and '(B)olus'; 'Type' with a dropdown menu set to 'Fluid Type'; 'Volume' with a text input set to '0' and a unit selector 'ml'; 'Additives' with a dropdown menu set to 'None'; and 'Rate' with a text input set to '0' and a unit selector 'ml/hr'. At the bottom center, a button labeled 'Sign Prescription' is circled in red. A blue arrow points from the text 'Sign Prescription to complete current prescription' to this button. The text 'Fluid Chart' is also visible at the bottom left of the interface.

The user can create multiple prescriptions and they can be viewed in the 'Current Fluid Balance Chart' section. The total volume prescribed as well as total amounts of sodium, chloride and potassium can be seen in the 'Volume and Electrolyte Totals' section. These can both be seen on the *Prescriptions finished page* (below). When the user decides to finish the case, there is an option to add any special instructions (see the *Special instructions page* below). For some prescriptions this is required. To finish the case simply select the 'Finish Prescribing' button at the bottom of the page.



**Special Instructions**

| Urine Output Monitoring                              | Overall Balance Frequency                              | Target Balance / Fluid Restriction                              |
|--|--|---|
| <input type="text" value="Urine Output Monitoring"/> | <input type="text" value="Overall balance Frequency"/> | <input type="text" value="Target Balance / Fluid Restriction"/> |

**Finish Prescribing**

#### *Special Instructions*

With the prescription finished, the user can see a review of the whole case. This is pictured on the *Prescriptions finished page*. Here, the generic and case specific feedback is displayed. There is also a form for the user to leave their feedback on the case



**Acute abdominal pain**

Specialty: Adult Surgical

Clinical Problem: Small Bowel Obstruction

NEWS Score: 3

Narrative: S: You have been asked to review a patient transferred from ED with diagnosis of Small Bowel obstruction as their fluid infusion is finished.  
B: Surgical history of previous right hemicolectomy with anastomoses two years ago. Past medical history includes Hypertension on Ramipril and Hypothyroidism on levothyroxine.  
A: Alert with mild tachycardia and NEWS score of 2.  
Laboratory investigations taken in ED are:  
Na 137mmol/L, K 4.0mmol/L, Cl 92mmol/L, Urea 3.2mmol/L, Creatinine 40micromol/L, eGFR >60ml/hr, Hb 104g/L, WCC 11.2x10<sup>9</sup>/L, PLT 100x10<sup>9</sup>/L, CRP 36mg/L  
R: Please prescribe further intravenous fluids while awaiting definitive investigation and management.

Patient Weight: 70kg

**Fluid Chart**

**Current Fluid Balance Chart**

| Prescribed At   | Indication          | Volume  | Type          | Additives       | Rate       | Signature |
|-----------------|---------------------|---------|---------------|-----------------|------------|-----------|
| 08-Jul-18 16:02 | Maintenance         | 20 ml   | Hartmann's    | None            | 20 ml/hr   | NL 123456 |
| 08-Jul-18 16:37 | Maintenance Deficit | 1000 ml | Hartmann's    | None            | 1000 ml/hr | NL 123456 |
| 08-Jul-18 16:37 | Maintenance Ongoing | 1000 ml | Normal Saline | 20 mmol / L KCl | 100 ml/hr  | NL 123456 |

**Volume and Electrolyte Totals**

Total Volume: 2020 ml (28.86 mmol/kg/da)

Sodium - 2.13 mmol/kg/day

Chloride - 1.97 mmol/kg/day

Potassium - 0.09 mmol/kg/day

Note: Volume and Electrolyte levels do not include any existing fluid intake figures from narrative ONLY those that are prescribed in the Current Fluid Balance Chart

**Special Instructions**

Urine Output Monitoring check

Overall Balance Frequency check

Target Balance / Fluid Restriction check

**General feedback**

General Feedback test

**Case Specific Feedback**

Case Feedback specific feedback that complements general approach feedback can be added here by case administrator

**Your Feedback**

If you have any comments specific to this case please feel free to enter them below, your feedback helps us make this learning tool better so any feedback is greatly appreciated.

Feedback

Save

Prescriptions finished page

Your Feedback form

## B.7 Case, User and Fluid Setup

### B.7.a Index Page

Each of these sections are structured in a very similar way. When selecting the relevant icon from the toolbar on the left the user is first taken to a table displaying the relevant information for that part of the project. In the User modules a table of users is displayed. For the case module it is cases, for the fluid module it is fluids. The table can be searched or organised by the different column names. Items are not deleted in this application. Instead they are soft-deleted by making them inactive. This can be done from the edit page. To see the items that have been made inactive deselect the active button at the top right of the screen. To edit an item the user can click the pencil icon at the right of each table row. It is also possible to add a new item to the database by selecting the 'plus' icon in the top right-hand corner of the screen.

Fluid Types - Search Results (6)

10 items

Active

plus icon to add new item to the database

Unselect active to see the items that have been soft-deleted

pencil icon to edit item

Each row corresponds to a single item in the database

| Abbreviation | Description                   | Sodium (mmol/l) | Chloride (mmol/l) | Potassium (mmol/l) | Bicarbonate (mmol/l) | Glucose (mmol/l) | Magnesium (mmol/l) | Calcium (mmol/l) | Can Have Additives |
|--------------|-------------------------------|-----------------|-------------------|--------------------|----------------------|------------------|--------------------|------------------|--------------------|
| NS           | Normal Saline                 | 154.00          | 154.00            | 0.00               | 0.00                 | 0.00             | 0.00               | 0.00             | ✓                  |
| 0.45 NS      | Half normal Saline            | 72.00           | 72.00             | 0.00               | 0.00                 | 0.00             | 0.00               | 0.00             |                    |
| CSL          | Hartmann's                    | 131.00          | 120.00            | 4.50               | 25.00                | 0.00             | 0.00               | 0.00             |                    |
| D10          | 10% Glucose                   | 0.00            | 0.00              | 0.00               | 0.00                 | 100.00           | 0.00               | 0.00             |                    |
| D5           | 5% Glucose                    | 0.00            | 0.00              | 0.00               | 0.00                 | 50.00            | 0.00               | 0.00             | ✓                  |
| 0.45 NS 20k  | Half normal saline 20mmol/KCl | 72.00           | 72.00             | 20.00              | 0.00                 | 0.00             | 0.00               | 0.00             |                    |

### B.7.b Edit Page

The user can then edit each item. For fluids this means title, abbreviation and electrolyte contents. For the User, it is fields such as name, email and role. For Case, is the case information and feedback as well as case images. In order to make a fluid inactive simply deselect the 'Active' checkbox at the bottom of the page.

Fluid Type >> 0.45 NS 20k

Abbreviation: 0.45 NS 20k

Description: Half normal saline 20mmol/KCl

Sodium: 72.00 mmol/l

Chloride: 72.00 mmol/l

Potassium: 20.00 mmol/l

Bicarbonate: 0.00 mmol/l

Glucose: 0.00 mmol/l

Magnesium: 0.00 mmol/l

Calcium: 0.00 mmol/l

Can Have Additives: ☐

Active: ☒

Save Cancel

Make fluid inactive

## B.8 Global Setting Setup

The Global Setting Module is slightly different to the modules described above. Here, there is only one relevant field in the database and that is the Generic Feedback displayed at the end of every case. This Module is composed of a form with just this one field.

## Appendix C: Supporting Documentation and Diagrams

### C.1 Project Outline Provided by Dr Alexander Davey

This section of the appendix contains the outline of the second stage of development of the application. I did all the programming for this stage. All documents are authored by Dr Alexander Davey.

#### *C.1.a General Project background*

##### **eFluidchart phase two development document: MVP for pilot in VIRTUE QI project**

Prescribing of intravenous fluids is cognitively challenging in a high volume high acuity general ward setting. Barriers to application of national guidelines at point of care include a theory practice gap that junior clinicians struggle to bridge in busy environments. To test potential feasibility of rapid cycle learning approach to support an iterative development and maintenance of knowledge and skills in prescription of intravenous fluids a proof of concept fluid prescribing simulator was developed with Create Software LTD. on a small private budget. Functionally this prototype allowed end users to log in and access a test clinical scenario and complete a digital fluid prescription. Logic based on regional modification of the NICE clinical guideline 174 (CG 174) on intravenous fluid therapy for adult patients in hospital was developed as a forcing function to ensure prescriptions made in simulation adhered to the key priorities for implementation in CG 174. Feedback was encouraging and the opportunity to use this tool as an adjunct to a multimodal education intervention to support improved fluid management at ward level in emergency surgical patients. Unfortunately, funding for improvement innovations in informatics is complex, highly competitive and has significant lag times. Further affordable and agile development is now possible with participation in the Code For Health University College London computer science masters collaborative. This collaboration involves participation of final year masters

students in development of a software project. Over the next six weeks mentors from local health and digital industry will support a student in phase two development of the electronic fluid chart intravenous fluid prescribing simulator application.

A minimum viable prototype that is fit for use as a simulation tool for rapid cycle learning among the ward team currently involved in the VIRTUE QI project requires six main areas of further development.

Target completion date weekending 15<sup>th</sup> July 2018 (TBC – Possibly a week earlier).

1. Self-registration for prescription user
2. Add Feedback details to prescription scenarios completed by user to explain how CG174 could be applied in this case to make a safe prescription
3. Add feedback option by user to submit comments or questions. Could be at end of case or on case over view page
4. Develop ability for System Administrator (top level access) to manage Fluid Type details
5. Add Thumbnail images function to speed up loading of pages with artefacts with associated clinical information required for fluid prescription.
6. Alter Prescription, monitoring & reassessment rule fixes as needed
7. Change summary dosing box to include prescribed and non-prescribed information. Include Potassium dose and change the units to mmol/kg/d for electrolytes and ml/kg/d for water.

More detailed description of purpose is contained below.

### *C.1.b Feedback Background*

#### **Self registration function**

#### **Feedback to user prescription scenarios completed**

This is a tricky function. How do we generate bespoke feed back on the structured application of tacit knowledge as per CG174(fluid guideline) with out the burden of having to create it individually. The structure below outlines a template that we could use to help with this. There are two tiers of feedback. Advice on how the guidelines are applied in a general sense is the diet tier. Our second tier would be structured statements that could be

pulled in for each scenario created. Essentially the scenario authors also follow this template answer when they are creating a case. I know there are implications for the CA role.

Scenario reference: [TITLE]

Feedback in general on application of NICE CG 174 & use of NI regional fluid balance chart

1. When approaching a fluid problem at the end of a bed always consider:
  - a) Does the patient need resuscitation: using ABCDE & 6P approach to emergency cardiovascular assessment [Figure below]
  - b) What abnormal fluid fluxes does the patient have and what is your estimate of their fluid deficit or excess: using History, examination, investigations & charts
  - c) Are there any complex clinical issues that make fluid management more hazardous (Such as liver, heart, renal, endothelial (sepsis, SIRS, pre-eclampsia) or nutritional failure)?
2. Then estimate the daily intake required. There is no hard and fast rule about how this is done but is should consider
  - a) Current deficit or excess. Plan to replace deficit over 48hours while avoiding large swings in overall balance if allowing for excess. Beware of the term fluid restriction - this could lead to inadvertent hypovolaemia in certain states. Consider special instructions for a target balance and frequency of overall balance to optimise safety of fluid management plan and prescription
  - b) Add in maintenance between 20 – 30ml/kg/d. Consider liberal or lower doses depending on underlying pathology. Frail, complex issues tend towards lower dose. As long as intravascular volume is maintained in patients at risk of AKI.
  - c) Subtract any bolus fluids from overall volume required
3. Consider type of fluid needed bearing in mind higher rates for multiple reasons generally implies deficit or losses from fluid of various electrolyte content. Diabetes insipidus causes high free water losses with polyuria.
  - a) Can't give 0.18% solutions as per CMO circular
  - b) Consider any special circumstances like variable rate intravenous insulin
  - c) Remember to use special instructions box on fluid balance chart and communicate monitoring frequency and any review triggers to nursing colleagues.

Scenario specific feedback for [Title]

1. Algorithm based summary
  - a. [statement]
  - b. [statement]
  - c. [statement]
2. Daily intake required
  - a. [statement]
  - b. [statement]
  - c. [statement]
3. Special considerations
  - a. [statement]
  - b. [statement]
  - c. [statement]

### **Feedback by user for scenario development**

This is important but might be technically easier to do. To maintain educational and clinical impact the scenarios must continuously evolve. Increased number of users may also highlight ways of applying CG174 that my magnificent brain has not yet thought of nor experienced. User experience is also critical. If the application is to be used by the 1000s of “junior” doctors in No alone we must ensure the experience is highly satisfactory.

Feedback must therefore allow us to broadly measure both user satisfaction and clinical validity there are a number of considerations I can see for this function. Location of feedback button. Content and method of answering. .

### **System administrator fluid library management**

As we start to think about scaling the application it will become inevitable that other fluid types have to be added. This is important to measure doses of water and electrolytes prescribed by the user. This feeds in to calculations about dosage limits and plans for reassessment and monitoring that are documented in special instructions boxes after the fluid prescription information is entered

### **Clinical information artefact thumbnails**

To maintain the high level of cognitive fidelity required certain additional sources of clinical information must be manufactured and presented with the SBAR story in each scenario.

These artefacts are usually in the form of vital sign and fluid balance charts or blood results forms that are scanned in as images. To ensure our plan for rapid cycle learning is actually rapid these images must be displayed as thumbnails as to not delay the page.

### Prescription, monitoring & reassessment rules

Logic used in the feedback when completing a prescription or special instructions on a chart will probably require some refinement. We can explore this in more detail in person or teleconference.

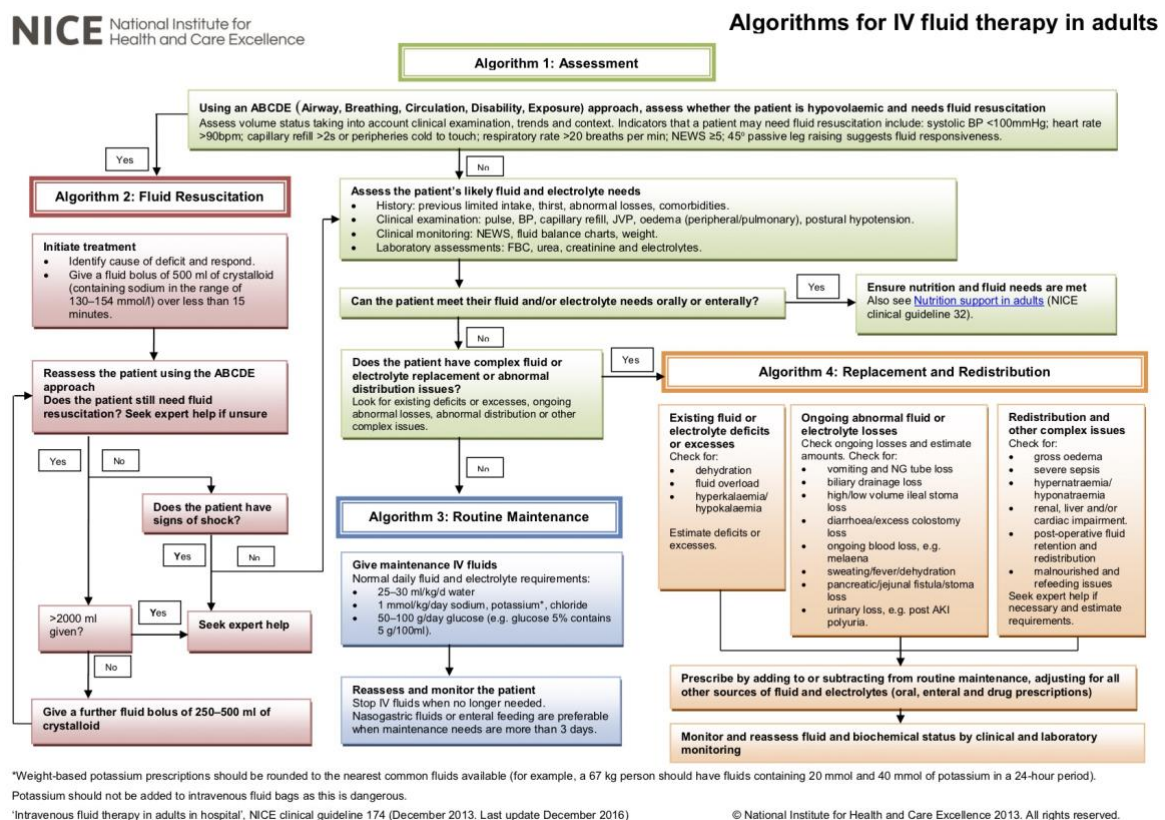
### Dosing summary box content

Doses of NA, Cl, K, H<sub>2</sub>O (and any other thing we may include in the future) are provided. Just remember there may be some non-prescribed fluids in the clinical scenario that may need pre-loaded into calculations. Will discuss options for handling this in more detail.

## C.2 NICE National Fluid Prescription Guidelines

This section contains the NICE Fluid Prescription Guidelines in full.

### C.2.a General Guidelines



## C.2.b Fluid Prescription Type Guidelines

This document provides further information regarding the type of fluid prescription

| VIRTUE IV Fluids process measures guide<br>(Volume. Indication. Rate & Reassessment / monitoring plan. Type of fluid UsEd)   |                |  |
|--|----------------|--|
| <b>1 – Record Reason(s)* for prescription in indication field of fluid balance chart</b>   |                |  |
| <b>2 – Rate +/- reassessment or monitoring plan should match the reasons recorded</b>  |                |  |
| Dose: H <sub>2</sub> O 25+/-5ml/kg/day; Na/Cl/K -1mmol/kg/day . Glucose 50 – 100g/day<br>*Abbreviations (with NICE CG174 Rs) - <b>M</b> : (Routine) Maintenance <b>D</b> : Deficit (Replacement) <b>O</b> : Ongoing Losses (Redistribution) <b>B</b> : Bolus (Resuscitation) |                |  |
| Rate (ml/hr)   | Reason(s)*     | Fluid Management & R/V Plan (Suggested minimum content & frequency)  |
| <84  | M              | Daily (Consider day time only plan, enhanced recovery, IV medicine & oral intake)  |
| 100 – 249  | M&D/M&O/M&D &O | 12 hourly (or fluid balance +/-10ml/kg) review of fluid management plan<br>Consider special instructions i.e. hourly urometry & overall balance record |
| M&D/M&O/M&D &O   | 250 - 499      | 6 hourly (or fluid balance +/-10ml/kg) review of fluid management plan<br>Must use special instructions i.e. hourly urometry & overall balance record  |
|  | 500 - 998      | In addition to above consider bolus & adjusting continuous infusion dose   |
| 999 (minimum)  | B (NEWS <5)    | NEWS Immediately after bolus & consider ABCDE Assessment.  |
| 500/<15mins  | B (NEWS >4)    | Both ABCDE Assessment & NEWS Immediately after bolus.  |
| <b>3 – Each prescription must have a Readable (printed surname / GMC n.o.) prescribing ID</b>  |                |  |

## C.2.c Feedback Explainer

**Feedback Explainer**

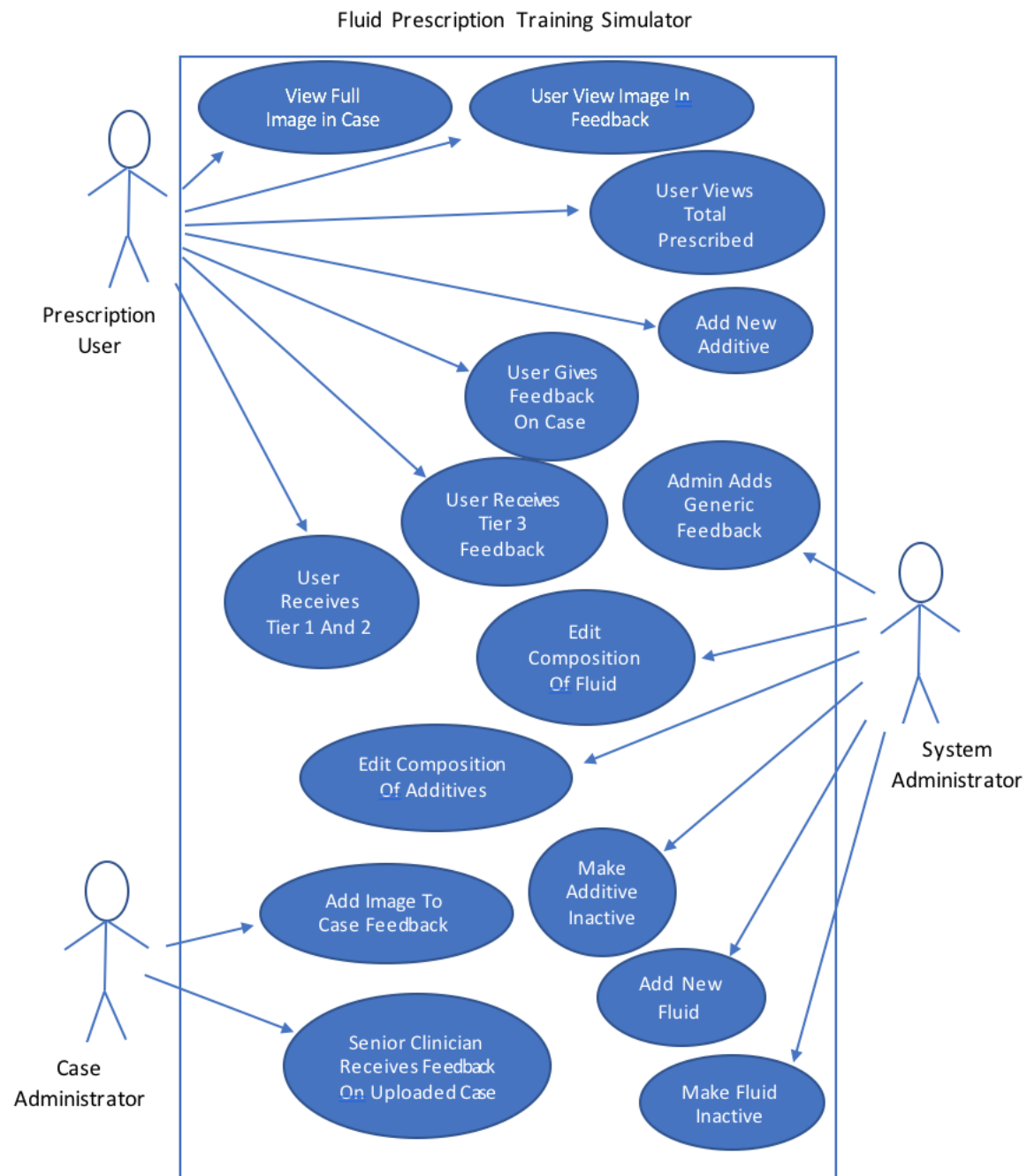
*The feedback the user will receive is split into three tiers.*

- Feedback for amount of fluid prescribed, which fluid is prescribed and the rate at which it is prescribed.*
  - E.g. Additives can only be added to fluid volumes of 500ml and 1000ml*
  - This is handled by validations*
- Feedback for the type of fluid prescription (i.e. Bolus, Maintenance, Ongoing or Deficit) this suggests when the patient be reassessed. This tier also takes care of whether the user has added any extra comments (which are necessary for certain fluid prescriptions).*
  - E.g. With a NEWS score of more than 4 only Bolus can be selected*
  - This is also handled by validations*
- Feedback for when the prescription has been:*
  - Generic Feedback: outlining the national fluid prescription guidelines*
  - Case Feedback: provided by the doctor who has added the case outlining how these guidelines apply to the case in question.*
  - These will appear as longer sections of text.*



## C.3 Use Cases

## C.3.a Use Case Diagram



## C.3.b Use Cases

## Use Case: AddNewFluid

|                |   |
|----------------|---|
| ID             | 1   |
| Description    | System Administrator adds new fluid to fluid library  |
| Actor          | System Administrator  |
| Preconditions  | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>   |
| Main Flow      | <ol style="list-style-type: none"> <li>User clicks on Fluid Setting in navigation panel on the left</li> <li>User Selects the 'Add-Fluid' button</li> <li>User enters the electrolyte composition of the fluid using the fields provided</li> <li>User submits the fluid</li> </ol> |
| Postconditions | <ul style="list-style-type: none"> <li>The fluid will be added to the database</li> <li>The user will be redirected to the fluid index page</li> </ul>  |

| Use Case: EditCompositionOfFluid |  |
|----------------------------------|--|
| ID                               | 2  |
| Description                      | System Administrator edits a fluid from the fluid library  |
| Actor                            | System Administrator   |
| Preconditions                    | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>  |
| Main Flow                        | <ol style="list-style-type: none"> <li>User clicks on Fluid Setting in navigation panel on the left</li> <li>User searches for fluid on the fluid index page               <ol style="list-style-type: none"> <li>The user can use the search bar</li> <li>Or can order fluids by electrolyte content</li> </ol> </li> <li>User edits the fluid using the fields provided</li> <li>User submits the fluid</li> </ol> |
| Postconditions                   | <ul style="list-style-type: none"> <li>The fluid will be updated in the database</li> <li>The user will be redirected to the 'Search Fluid' page</li> </ul>  |

| Use Case: MakeFluidInactive |  |
|-----------------------------|--|
| ID                          | 3  |
| Description                 | System Administrator makes a fluid inactive, so it no longer shows up as an option in prescriptions  |
| Actor                       | System Administrator   |
| Preconditions               | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>  |
| Main Flow                   | <ol style="list-style-type: none"> <li>User clicks on Fluid Setting in navigation panel on the left</li> <li>User searches for fluid on the fluid index page               <ol style="list-style-type: none"> <li>The user can use the search bar</li> <li>Or can order fluids by electrolyte content</li> </ol> </li> <li>User unchecks the 'make active' checkbox</li> <li>User submits the fluid</li> </ol> |
| Postconditions              | <ul style="list-style-type: none"> <li>The fluid will be updated in the database</li> <li>The fluid will not show when user is making prescriptions</li> <li>The user will be redirected to the 'Search Fluid' page</li> </ul>   |

| Use Case: ViewFullImageInCase |  |
|-------------------------------|--|
| ID                            | 4  |
| Description                   | Prescription User view full image by clicking on thumbnail |

|                |  |
|----------------|--|
| Actor          | Prescription User  |
| Preconditions  | <ul style="list-style-type: none"> <li>Case has been uploaded by Case Administrator with photo</li> <li>User is logged on as Prescription User</li> </ul>  |
| Main Flow      | <ol style="list-style-type: none"> <li>User selects a case to work on</li> <li>The page loads in a reasonable time</li> <li>User clicks on the thumbnail</li> <li>The full image is displayed</li> </ol> |
| Postconditions | <ul style="list-style-type: none"> <li>Full quality image will be display</li> </ul>   |

| Use Case: UserReceivesTier1And2Feedback |  |
|---|--|
| ID                                      | 5  |
| Description                             | User receives instant feedback for fluid type, rate and volume as well as how soon the patient should be reassessed. The feedback is displayed as simple validations.  |
| Actor                                   | Prescription User  |
| Preconditions                           | <ul style="list-style-type: none"> <li>User is logged in as Prescription User</li> </ul>   |
| Main Flow                               | <ol style="list-style-type: none"> <li>User selects from list of cases on home screen</li> <li>User Clicks the 'Make Prescription' button</li> <li>User enters fluid type, volume, rate and selects from Bolus, Maintenance, Deficit and Ongoing (which are measures of how soon the patient should next be assessed)</li> <li>User clicks 'create prescription'</li> </ol>                          |
| Postconditions                          | <ul style="list-style-type: none"> <li>Participant sees constructive feedback in the form of validations on what they have prescribed <ul style="list-style-type: none"> <li>e.g. "maintenance can only be selected with fluid rates of less than 10 ml/hr"</li> </ul> </li> <li>User is given the option of adding any extra comments to their prescription or adding a new prescription</li> </ul> |

| Use Case: UserReceivesTier3Feedback |  |
|-------------------------------------|--|
| ID                                  | 6  |
| Description                         | User receives dense textual feedback covering the national fluid prescription guidelines (generic feedback) and how they relate to the case currently being worked on (case specific feedback).  |
| Actor                               | Prescription User  |
| Preconditions                       | <ul style="list-style-type: none"> <li>User is logged in as Prescription User</li> <li>Case has been added by Case Administrator</li> <li>User has just created a prescription for a case</li> <li>System Administrator has added generic feedback</li> </ul>  |
| Main Flow                           | <ol style="list-style-type: none"> <li>User chooses to whether or not to fill in extra feedback section for the case</li> <li>User clicks 'finish prescription' (even if the extra feedback has been left blank) <ol style="list-style-type: none"> <li>On certain cases user will get a validation saying that they must include some extra feedback</li> </ol> </li> </ol> |

|                |  |
|----------------|--|
| Postconditions | <ul style="list-style-type: none"> <li>User sees textual description of national fluid guidelines and how they apply to the particular case</li> <li>User stays on the same page and is able to review the case and their prescription in light of feedback</li> </ul> |
|----------------|--|

| Use Case: AdminAddsGenericFeedback |  |
|------------------------------------|--|
| ID                                 | 7  |
| Description                        | System Administrator can add or edit generic feedback. This will be the national fluid prescription guidelines and will be displayed on the completion of cases.   |
| Actor                              | System Administrator   |
| Preconditions                      | <ul style="list-style-type: none"> <li>User is logged in as admin</li> </ul>   |
| Main Flow                          | <ol style="list-style-type: none"> <li>User selects global settings from navigation bar on the right</li> <li>User enters or edits global feedback in text box provided</li> </ol>   |
| Postconditions                     | <ul style="list-style-type: none"> <li>Global feedback is added-to/updated-in the database</li> <li>Admin remains on global settings page and can navigate to other pages using navigation bar on right of screen</li> </ul> |

| Use Case: SeniorClinicianAddsCaseSpecificFeedback |  |
|---|--|
| ID  | 8  |
| Description                                       | Case Administrator can add or edit case specific feedback. This will explain how the national fluid prescription guidelines apply to the case and will be displayed on the completion of cases.  |
| Actor   | System Administrator   |
| Preconditions                                     | <ul style="list-style-type: none"> <li>User is logged in as Case Administrator</li> </ul>  |
| Main Flow   | <ol style="list-style-type: none"> <li>User selects case settings from navigation bar on the right</li> <li>User Creates new case or selects case she would like to add feedback to</li> <li>User enters or edits case specific feedback in text box provided</li> </ol> |
| Postconditions                                    | <ul style="list-style-type: none"> <li>Case specific feedback is added-to/updated-in the database</li> <li>Case Administrator is redirected to the Case index page and can navigate to other pages using navigation bar on right of screen</li> </ul>                    |

| Use Case: UserGivesFeedbackOnCase |  |
|-----------------------------------|--|
| ID                                | 9  |
| Description                       | After having finished a prescription for a case user is able to leave their own feedback on what the case. They may want to say that they didn't understand part of the case or that they felt part of it didn't make sense. |
| Actor                             | Prescription User  |
| Preconditions                     | <ul style="list-style-type: none"> <li>User is logged in as Prescription User</li> <li>User has just created a prescription for a case</li> </ul>  |
| Main Flow                         | <ol style="list-style-type: none"> <li>User chooses to whether or not to fill in extra feedback section for the case</li> </ol>  |

|                |   |
|----------------|---|
|                | <ol style="list-style-type: none"> <li>2. User clicks 'finish prescription' (even if the extra feedback has been left blank) <ol style="list-style-type: none"> <li>a. On certain cases user will get a validation saying that they must include some extra feedback</li> </ol> </li> <li>3. User receives generic and case specific feedback</li> <li>4. User is presented with a text field in which they are able to leave feedback in textual form</li> </ol> |
| Postconditions | <ul style="list-style-type: none"> <li>• User can see that feedback has been successfully logged</li> <li>• User stays on the same page and is able to review the case and their prescription in light of feedback</li> </ul>   |

| Use Case: SeniorClinicianReceivesFeedbackOnUploadedCase |  |
|---|--|
| ID  | 10   |
| Description   | A Case Administrator who has uploaded a case is able to log in and be able to view feedback from Prescription Users who have attempted the case  |
| Actor   | Case Administrator   |
| Preconditions   | <ul style="list-style-type: none"> <li>• User is logged in as Case Administrator</li> <li>• User has previously added case the library</li> <li>• Prescription Users have completed the case and left some feedback on it</li> </ul> |
| Main Flow   | <ol style="list-style-type: none"> <li>1. User selects case from dashboard</li> <li>2. User clicks view feedback</li> <li>3. User is able to view all feedback placed on case by Prescription Users</li> </ol>                       |
| Postconditions  | <ul style="list-style-type: none"> <li>• User can see all Prescription User feedback</li> </ul>  |

| Use Case: UserViewsTotalPrescribedPotassium |  |
|---|--|
| ID  | 11   |
| Description                                 | After multiple prescriptions including additives (prescribed on top of regular fluid types) user is able to see the total amount of potassium prescribed in the case   |
| Actor                                       | Prescription User  |
| Preconditions                               | <ul style="list-style-type: none"> <li>• User is logged in as Prescription User</li> <li>• Case Administrator has previously added case the library</li> <li>• User has added multiple prescriptions including additives</li> </ul>  |
| Main Flow                                   | <ol style="list-style-type: none"> <li>1. User looks over the electrolyte totals on right of page <ol style="list-style-type: none"> <li>a. On smaller screen sizes totals will be below prescription summary</li> </ol> </li> <li>2. User is able to view how much potassium has been prescribed</li> </ol> |
| Postconditions                              | <ul style="list-style-type: none"> <li>• User can see total prescribed potassium</li> </ul>  |

| Use Case: AddNewAdditive |
|--------------------------|
|--------------------------|

|                |   |
|----------------|---|
| ID             | 12  |
| Description    | System Administrator adds new additive to additive library  |
| Actor          | System Administrator  |
| Preconditions  | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>   |
| Main Flow      | <ol style="list-style-type: none"> <li>User clicks on Additive Setting in navigation panel on the left</li> <li>User Selects the 'Add-Additive' button</li> <li>User enters the electrolyte composition of the additive using the fields provided</li> <li>User submits the additive</li> </ol> |
| Postconditions | <ul style="list-style-type: none"> <li>The additive will be added to the database</li> <li>The user will be redirected to the additive index page</li> </ul>  |

| Use Case: EditCompositionOfAdditives |  |
|--------------------------------------|--|
| ID                                   | 13   |
| Description                          | System Administrator edits an additive from the additive library   |
| Actor                                | System Administrator   |
| Preconditions                        | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>  |
| Main Flow                            | <ol style="list-style-type: none"> <li>User clicks on Additive Setting in navigation panel on the left</li> <li>User searches for additive on the additive index page               <ol style="list-style-type: none"> <li>The user can use the search bar</li> <li>Or can order additives by electrolyte content</li> </ol> </li> <li>User edits the additive using the fields provided</li> <li>User submits the additive</li> </ol> |
| Postconditions                       | <ul style="list-style-type: none"> <li>The additive will be updated in the database</li> <li>The user will be redirected to the 'Search Additive' page</li> </ul>  |

| Use Case: MakeAdditiveInactive |   |
|--------------------------------|---|
| ID                             | 14  |
| Description                    | System Administrator makes an additive inactive, so it no longer shows up as an option in prescriptions   |
| Actor                          | System Administrator  |
| Preconditions                  | <ul style="list-style-type: none"> <li>User is logged in as System Administrator</li> </ul>   |
| Main Flow                      | <ol style="list-style-type: none"> <li>User clicks on Additive Setting in navigation panel on the left</li> <li>User searches for additive on the additive index page               <ol style="list-style-type: none"> <li>The user can use the search bar</li> <li>Or can order additives by electrolyte content</li> </ol> </li> <li>User unchecks the 'make active' checkbox</li> <li>User submits the additive</li> </ol> |
| Postconditions                 | <ul style="list-style-type: none"> <li>The additive will be updated in the database</li> <li>The additive will not show when user is making prescriptions</li> <li>The user will be redirected to the 'Search Additive' page</li> </ul>   |

| Use Case: AddImageToCaseFeedback |    |
|----------------------------------|----|
| ID                               | 15 |

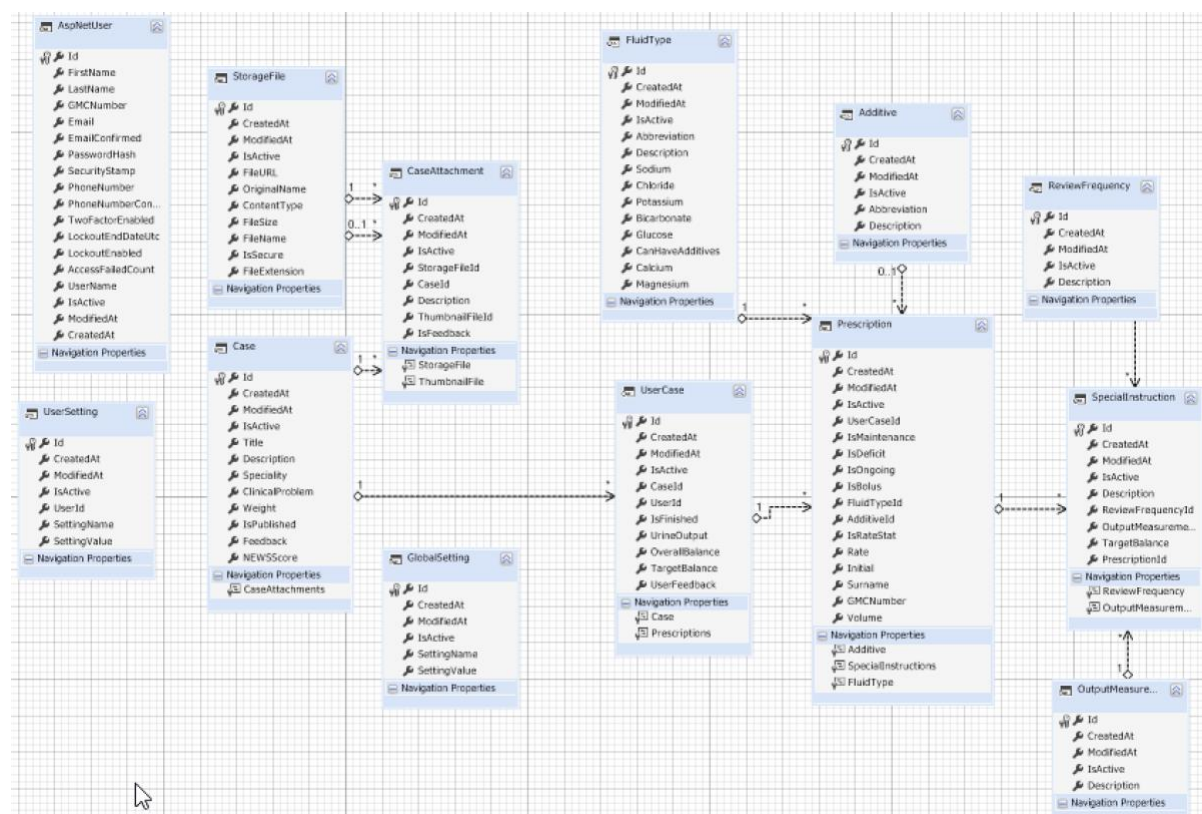
|                |  |
|----------------|--|
| Description    | Case Administrator adds a picture to case feedback   |
| Actor          | Case Administrator   |
| Preconditions  | <ul style="list-style-type: none"> <li>User is logged in as Case Administrator</li> <li>Case has already been created</li> </ul>   |
| Main Flow      | <ol style="list-style-type: none"> <li>User clicks on case setting on navigation panel on the left of page</li> <li>User selects case to edit               <ol style="list-style-type: none"> <li>User can search cases</li> <li>Or user can order by case attributes</li> </ol> </li> <li>User uses the upload file field to add image to case feedback</li> </ol> |
| Postconditions | <ul style="list-style-type: none"> <li>The image will be added in the database to the case</li> <li>The user will be redirected to the case index page</li> </ul>  |

| Use Case: UserViewImageInFeedback |   |
|-----------------------------------|---|
| ID                                | 16  |
| Description                       | User can see the full image when seeing generic or specific feedback at the end of the case   |
| Actor                             | Prescription User   |
| Preconditions                     | <ul style="list-style-type: none"> <li>Case has been uploaded by Case Administrator with photo in generic or case feedback</li> <li>User is logged on as Prescription User</li> <li>User has created a prescription</li> </ul>  |
| Main Flow                         | <ol style="list-style-type: none"> <li>User selects finish prescription               <ol style="list-style-type: none"> <li>On certain cases user will get a validation saying that they must include some extra feedback</li> </ol> </li> <li>Generic and case specific feedback will be displayed to the user including the picture uploaded by Case Administrator who has uploaded the case</li> <li>User will be able to view thumbnail images as part of this feedback</li> <li>User clicks on image which will display full quality image</li> </ol> |
| Postconditions                    | <ul style="list-style-type: none"> <li>Full quality image will be displayed</li> </ul>  |

| Use Case: CaseAdminAddsNewsScore |  |
|----------------------------------|--|
| ID                               | 17   |
| Description                      | Case Administrator can add or edit case NEWS score.  |
| Actor                            | Case Admin   |
| Preconditions                    | <ul style="list-style-type: none"> <li>User is logged in as Case Admin</li> </ul>  |
| Main Flow                        | <ol style="list-style-type: none"> <li>User selects case settings from navigation bar on the right</li> <li>User Creates new case or selects case she would like to add feedback to</li> <li>User enters or edits case specific feedback in text box provided</li> </ol> |
| Postconditions                   | <ul style="list-style-type: none"> <li>NEWS score is added-to/updated-in the database</li> <li>User is redirected to the Case index page and can navigate to other pages using navigation bar on right of screen</li> </ul>  |

| Use Case: ViewFullImageInCase |   |
|-------------------------------|---|
| ID                            | 18  |
| Description                   | UserViewsImageInFeedback  |
| Actor                         | Prescription User   |
| Preconditions                 | <ul style="list-style-type: none"> <li>News Score has been uploaded by Case Admin to case</li> <li>User is logged on as Prescription User</li> </ul>  |
| Main Flow                     | <ol style="list-style-type: none"> <li>1. User selects a case to work on</li> <li>2. User Views News Score</li> <li>3. User Creates Prescription in response to NEWS score</li> <li>4.</li> </ol> |
| Postconditions                | <ul style="list-style-type: none"> <li>User can fill in prescription in response to NEWS score</li> </ul>   |

## C.4 Database Schema



## C.5 My Changes in Full

| Id  | Requirement   | Code   |
|---|---|--|
| <b>P1: How well application functions in original hospital.</b> |   |  |
| R1  | System will have validation logic for first and second tier feedback  | There were some validations before I began. I have rewritten most of them that were there and added many that weren't. |
| R2  | System will allow System Admin to create and edit generic feedback. This will be set in a Global Setting module that will need to be created. | Whole module was created by me   |



|     |  |   |
|-----|--|---|
| R3  | System will allow Case Administrator to add Case Specific feedback   | The database table field and everything that comes with that (i.e. additions to view models and POCO items) was created by me.                                  |
| R4  | System will display Generic and Case Specific feedback on completion of prescription   | Created the partial view and ensured that it displayed correctly.   |
| R5  | System will allow Case Administrator to add image to feedback  | Added checkbox so Case admin can select whether an image they upload is for feedback or case narrative. Also added field in database to store this information. |
| R6  | System will display feedback including images added by Case Administrator to User on completion of prescription                | Created the entirety of the partial views and displayed them. This includes the display of the images .   |
| R7  | System will ensure that validations contain constructive feedback. These must adhere to National Fluid Prescription Guidelines | No code   |
| R8  | System will give user option to provide feedback on Case when prescription has been finished                                   | Created the partial view with the form and the database tables.   |
| R9  | System will have module for Case Administrators to see feedback on their cases.  | Did not implement   |
| R10 | System will allow Case Administrator to add NEWS score to the database   | Created database table and form field.  |
| R11 | System will display NEWS score to users as part of case information  | Created the display for NEWS Score part of the view.  |
| R12 | System will display total potassium dose of all fluids prescribed including any potassium in prescribed additives              | Outlined in 4.2.c. Created PotassiumInAdditive, PotassiumDose and PotassiumByWeight methods.  |

## P2: The look and feel of the application.

|     |  |   |
|-----|--|---|
| R13 | System will generate a thumbnail of a new image added in Case Setup. This thumbnail will be added along with the original image to the Azure container | Section 4.2.f explains some of the code I wrote. I also created the code to display the image as a thumbnail and I created the field in the database to store the thumbnailId. There already was a method to upload images to the azure container and to upload the image ID to the database. |
| R14 | System will display thumbnail image on MyCase Index page   | Before I started the full image displayed as a thumbnail and could be selected to be seen in a larger form. I changed the code so that only the thumbnail displays as the thumbnail image. When you click on it the full image displays. Index file in MyCase folder lines 86-106             |
| R15 | System will allow user to view original full quality image by clicking on thumbnail image  | See above   |
| R16 | System will display Case Index page in reasonable time   | See above   |
| R17 | Front end of system will have professional feel  | Lots of little fixes throughout the project.  |

## P3: How well application will function at another hospital

|     |  |   |
|-----|--|---|
| R18 | System will allow System Admins to add a new fluid to the library            | I created the whole edit page in fluid section  |
| R19 | System will allow System Admins to make a fluid inactive                     | This was part of the edit page so see above   |
| R20 | System will allow System Admin to edit composition of fluid                  | See above   |
| R21 | System will allow System Admin to view fluid library                         | I created the fluid table explained in Section 4.3.a.   |
| R22 | System will allow user to select fluid from library when making prescription | This was already in place. It just required that I be able to create an interface with which an admin could add a fluid to the database so see above. |
| R23 | System will display only fluid name to user                                  | Small fix in javascript file found in NHS.Fluid.WebApp/Scripts/index.js in selectize method (line 242)  |

|            |  |   |
|------------|--|---|
|            |  | had change to ensure that it only displayed description of the fluid not the abbreviation.  |
| <b>R24</b> | System will allow System Admins to add a new Additive to the library | Not implemented   |
| <b>R25</b> | System will allow System Admins to make an Additive inactive         | Not implemented   |
| <b>R26</b> | System will allow System Admin to edit composition of Additive       | Not implemented   |
| <b>R27</b> | System will allow System Admin to view Additive library              | Not implemented   |
| <b>R28</b> | System will display on most screen sizes                             | I did most of the code changed to make sure the application worked on smaller screen sizes. There are lots of small changes throughout the application. |

### C.6 GetInclude() method

This method is found in the 'UserService' class in the Service layer. I did not write it myself, I only called it.

```
public UserCaseItem GetInclude(int id)
{
    return Mapper.Map<UserCase, UserCaseItem>(persistenceManager.GetInclude(x => x.Id == id, "Case", "Case.CaseAttachments", "Case.CaseAttachments.StorageFile", "Prescriptions", "Prescriptions.Additive", "Prescriptions.SpecialInstructions", "Prescriptions.FluidType", "Prescriptions.SpecialInstructions.ReviewFrequency", "Prescriptions.SpecialInstructions.OutputMeasurementFrequency"));
}
```

### C.7 Validation Error

First, part of the guidelines states: If Maintenance is selected as the fluid type the rate must be 84 ml/hr or less. Now, when making a prescription, if Bolus is selected and the rate is more than 84 ml/hr then the validation displayed should state the rate at which a Bolus fluid should be set. It should not be state that Maintenance should be selected for that rate. The type of fluid selected is more important than the rate when choosing which validation to display.

### C.8 Adherence to Design Conventions

Displayed here are the index pages for the Case Setup and User modules (which were already part of the project) and the Fluid and Global Setting Setup modules (that I added to the project). It is possible to see how the two modules I have added adhere to the design conventions of the two modules which were already part of the project. It must be noted

that the Global Setting module is slightly different. There will only ever be one item in the GlobalSetting table in the database so there will be no table Partial View rendered on the index page.

| Case Title                              | Speciality     | Clinical Problem        | Attachments | Published |
|---|----------------|-------------------------|-------------|-----------|
| Acute abdominal pain                    | Adult Surgical | Small Bowel Obstruction | 2           | Published |
| Oliguria                                | Adult Surgical | Acute Kidney Injury     | 0           |           |
| Anorexia, vomiting & worsening mobility | Adult Medical  | Dehydration             | 0           |           |
| Right upper quadrant pain               | Adult Surgical | Biliary colic           | 4           | Published |

Case Index

| First Name | Last Name  | Email                               | GMC Number | Permissions   |
|------------|------------|-------------------------------------|------------|---|
| Simon      | Smith      | simon.smith@sms720.com              | 123456     | Case Administrator<br>System Administrator<br>Prescription User |
| Alexander  | Davey      | D1406199@qub.ac.uk                  | 6162717    | Case Administrator<br>System Administrator<br>Prescription User |
| Colin      | Munn       | Cjmunnn@hotmail.com                 | 0001000    | Prescription User   |
| Mark       | Mcferran   | Mark.mcferran@kcl.ac.uk             | 0002000    | Prescription User   |
| Kavi       | Manektella | Kavi.manektella@doctors.org.uk      | 0003000    | Prescription User   |
| Noel       | Sharkey    | noel.sharkey@belfasttrust.hscni.net | 0005000    | Prescription User   |
| Patrick    | Grant      | Pgrant13@qub.ac.uk                  | 0004000    | Prescription User   |
| Rachael    | Allen      | rallen18@qub.ac.uk                  | 0007000    | Prescription User   |
| Simon      | Smith      | simon@createsoftware.co.uk          | 99999      | Prescription User   |

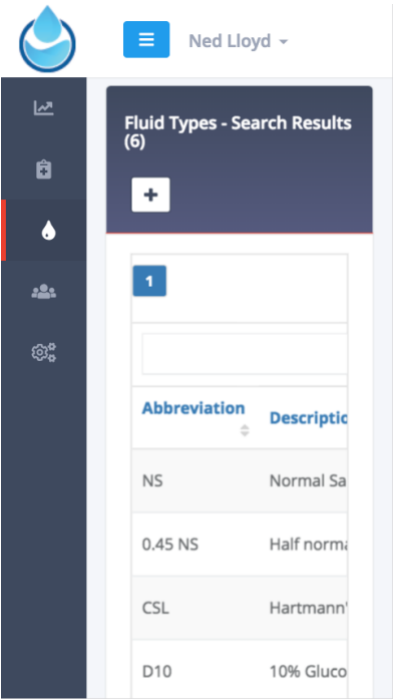
Users Index

*Fluid Index*

## Global Setting Index

## 76

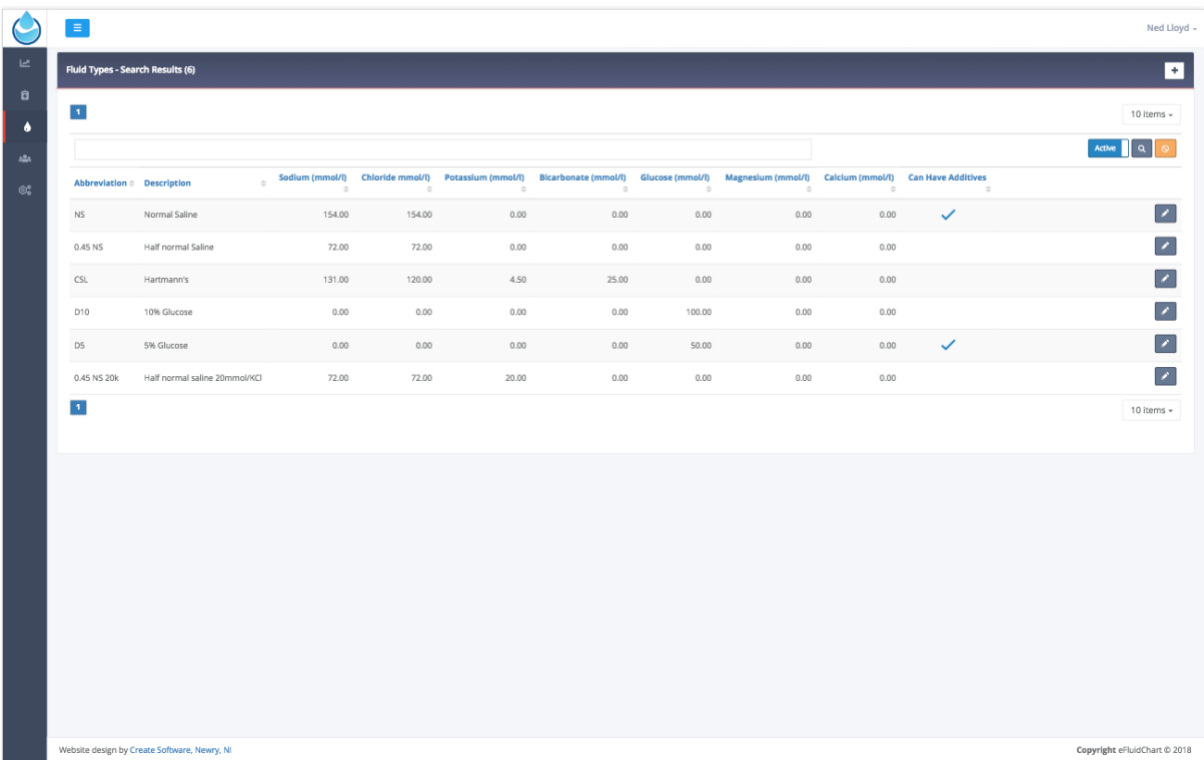
iPhone 5 - Dashboard



Here, it is possible to see a good example of the differences in functionality between different screen sizes. The iPhone 5 still has the same functionality as a full screen and does not look too cramped. However, it is slightly more difficult to use because in order to view a case in full the user must scroll across in the table. In the large computer screen example, the edit button can be seen and clicked immediately. This is a good example of some of the small losses in functionality come from using the project with different screen sizes. It should be noted that not all pages are like this and even fairly complex pages such as the prescription page experience no loss in functionality on

smaller screen sizes.

Large Computer Screen- Dashboard



## Appendix D: Testing

## D.1 Requirements With Outline of How They Were Tested

| Id  | Requirement  | Type | Comments  | Tests Suites |
|-----|--|------|---|--------------|
| R1  | System will have validation logic for first and second tier feedback   | A    | Done with all possible feedback messages  | TS3-TS6      |
| R2  | System will allow Admin to create and edit generic feedback. This will be set in a Global Setting module that will need to be created.                 | M    | Selenium IDE would not automatically type into 'Selectize' text boxes I used. I made sure that up to 8000 characters would display with manual testing. I used <b>Use case U7</b>   | TS3-TS6      |
| R3  | System will allow Case Administrator to add Case Specific feedback   | M    | Selenium IDE would not automatically type into 'Selectize' text boxes I used. I made sure that up to 8000 characters would display with manual testing. Manual tested using <b>Use case U8</b>  |              |
| R4  | System will display Generic and Case Specific feedback on completion of prescription   | A    | Presence of Feedback tested.  | TS2          |
| R5  | System will allow Case Administrator to add image to feedback  | M    | Was difficult with Selenium with IDE to automate adding an image. Any images added would also display throughout the application. I manual tested this by running through <b>Use case U16</b>   |              |
| R6  | System will display feedback including images added by Case Administrator to User on completion of prescription  | A    | Tests ensure that there are images present. It is possible though that the test would fail if the images were removed from the case. This would be unlikely though.   | TS2          |
| R7  | System will ensure that validations contain constructive feedback. These must adhere to National Fluid Prescription Guidelines                         | M    | Not able to write tests for this. This was confirmed by Dr Davey on the most recent version of the project.   | TS2          |
| R8  | System will give user option to provide feedback on Case when prescription has been finished   | A&M  | Automated test ensuring that the feedback form does display at end of prescription. Not possible to create automated tests checking that feedback does go into database. Instead I manual tested this using SQL Server Management Studio.   | TS2          |
| R9  | System will have module for Case Administrators to see feedback on their cases.  | N/A  | Did not implemented ('Might Have')  |              |
| R10 | System will allow Case Administrator to add NEWS score to the database   | A&M  | Did not want to automate the positive test as the addition of the NEWS score would be viewable throughout the project. Instead I automated negative tests and manual teste the positive case. Using <b>Use case U17</b>   | TS8          |
| R11 | System will display NEWS score to users as part of case information  | A    | Automated test ensuring NEWS score is present. Manual test ensuring it matches the score given by the Case Admin  | TS8          |
| R12 | System will display total potassium dose of all fluids prescribed including any potassium in prescribed additives                                      | A    | Automated test ensuring correct dose corresponding to prescription.   | TS2          |
| R13 | System will generate a thumbnail of a new image added in Case Setup. This thumbnail will be added along with the original image to the Azure container | M    | Difficult to test because I do not have access to the Azure container. Instead I used <b>Use case U4</b> to ensure that an image is displayed. It is possible to test whether images are reduced size by observing loading time. It would also be very easy to test with access to Azure container. |              |

|            |   |     |  |     |
|------------|---|-----|--|-----|
| <b>R14</b> | System will display thumbnail image on MyCase Index page                                  | A   | Automated test ensuring that there is an image present. This could however fail in the unlikely event the image is deleted (there is no reason it ever should be)            | TS7 |
| <b>R15</b> | System will allow user to view original full quality image by clicking on thumbnail image | A   |  | TS7 |
| <b>R16</b> | System will display Case Index page in reasonable time                                    | M   | Not suitable for automated testing. Worked with Dr Davey until he was satisfied with loading times   |     |
| <b>R17</b> | Front end of system will have professional feel   | N/A | Difficult to test for. But I did try to apply Don Norman's heuristics to my work. This will be tested in the next stage of development by user testing in the hospital.      |     |
| <b>R18</b> | System will allow System Administrators to add a new fluid to the library                 | A&M | Negative tests automated. Positive test done manually. Tried to test for boundary cases such as adding a very long number. Used <b>Use cases U1 and U2</b> for manual tests. | TS1 |
| <b>R19</b> | System will allow System Administrators to make a fluid inactive                          | M   | Manual test. Used <b>Use case U3</b>   |     |
| <b>R20</b> | System will allow System Administrator to edit composition of fluid                       | A&M | See R18  | TS1 |
| <b>R21</b> | System will allow System Administrator to view fluid library                              | A   | Manually Tested  | TS1 |
| <b>R22</b> | System will allow user to select fluid from library when making prescription              |     | Automated testing for Sodium and Hartman's. It would fail if one of these were removed but there is no reason why this should happen.  | TS2 |
| <b>R23</b> | System will display only fluid name to user   | M   | Very Simple Manual test. As part of step three on <b>Use case 5</b> this was confirmed.  |     |
| <b>R24</b> | System will allow System Administrators to add a new Additive to the library              |     | Did not implement ('Could Have')   |     |
| <b>R25</b> | System will allow System Administrators to make an Additive inactive                      |     | Did not implement ('Could Have')   |     |
| <b>R26</b> | System will allow System Administrator to edit composition of Additive                    |     | Did not implement ('Could Have')   |     |
| <b>R27</b> | System will allow System Administrator to view Additive library                           |     | Did not implement ('Could Have')   |     |
| <b>R28</b> | System will display on most screen sizes  | M   | Manual tested to ensure does work on most screen sizes.  |     |

## D.2 Final Testing Errors

Below is a brief summary of the errors that were discovered by running the tests, specified in Appendix D.2, at the end of the project. I have also detailed how I went about fixing them.

| Test       | Error  | Solution  |
|------------|--|---|
| <b>TS1</b> | Fluid can be added with negative number for electrolyte content.   | Added validation on the 'FluidViewVM' View Model. |
| <b>TS1</b> | No custom validation for adding a number too big for the database field. Before, validation simply displayed that the item could not be added to the database. | Add Custom Validation.                            |

|                                     |   |  |
|-------------------------------------|---|--|
| <b>Manual Test:<br/>Use Case U7</b> | Unnecessary validation prohibiting input of more than 8000 characters. In fact, global feedback could be unlimited. | Removed validation   |
| <b>TS8</b>                          | Validation for NEWS score displayed wrong messages.   | Changed the messages to display the correct message (that the NEWS score could be in the range of 0 to 15) |
| <b>Manual Test</b>                  | On the prescription page lots of the description falls off the edge on a small screen size.                         | Removed minimum size of the text box so that it would continue to condense.                                |
| <b>TS4</b>                          | Incorrect message for Bolus Volume test   | Corrected message  |
| <b>TS4</b>                          | Incorrect message in  |  |

### D.3 Test Example

Below is an example of a Selenium tests checking that the NEWS Score cannot be a negative number.

```
{
  "id": "889fffc1-7f05-474b-a706-3fe40049be80",
  "name": "NEWSScore-veNumber",
  "commands": [{
    "id": "7ed8956a-a760-4f5a-9dfb-d74b102d092e",
    "comment": "",
    "command": "open",
    "target": "/Case",
    "targets": [],
    "value": ""
  }, {
    "id": "a8ea05a6-079c-4a5e-992a-37d9728d5d5a",
    "comment": "",
    "command": "click",
    "target": "xpath=//table[@id='dataTable']/tbody/tr[4]/td[6]/a",
    "targets": [
      ["xpath=//table[@id='dataTable']/tbody/tr[4]/td[6]/a", "xpath:idRelative"],
      ["xpath=//a[contains(@href, '/Case/Edit/4')][2]", "xpath:href"],
      ["xpath=//tr[4]/td[6]/a", "xpath:position"]
    ],
    "value": ""
  }, {
    "id": "629edf1a-1774-46e9-941d-0da7e6270373",
    "comment": "",
    "command": "verifyTitle",
    "target": "eFluidChart | Case",
    "targets": [],
    "value": ""
  }, {
    "id": "26002257-f58f-4378-bd39-a8cea44cb008",
    "comment": "",
    "command": "type",
    "target": "id=NEWSScore",
    "targets": [
      ["id=NEWSScore", "id"],
      ["name=NEWSScore", "name"],
      ["css=#NEWSScore", "css"],
      ["xpath=//input[@id='NEWSScore']", "xpath:attributes"],
      ["xpath=//div[@id='page-wrapper']/div[2]/div/div/div/div[2]/form/div[5]/div/input", "xpath:idRelative"],
      ["xpath=//div[5]/div/input", "xpath:position"]
    ],
    "value": "-1"
  }, {
    "id": "d6933acf-e6e0-4826-8bf0-9dd39016bd22",
```



```

    "comment": "",
    "command": "click",
    "target": "xpath=//button[@type='submit']][7]",
    "targets": [
      ["xpath=//button[@type='submit']][7]", "xpath:attributes"],
      ["xpath=//div[@id='page-wrapper']/div[2]/div/div/div/div[2]/form/div[10]/div/button", "xpath:idRelative"],
      ["xpath=//div[10]/div/button", "xpath:position"]
    ],
    "value": ""
  }, {
    "id": "ef7ae046-ba2c-4a4a-ba28-d703c7d96a45",
    "comment": "",
    "command": "verifyText",
    "target": "css=div.validation-summary-errors > ul > li",
    "targets": [
      ["css=div.validation-summary-errors > ul > li", "css"],
      ["xpath=//div[@id='page-wrapper']/div[2]/div/div/div/div[2]/form/div/ul/li", "xpath:idRelative"],
      ["xpath=//form/div/ul/li", "xpath:position"]
    ],
    "value": "NEWS score cannot be less than 1 or more than 15"
  }
}

```

## Appendix E: Code Listing

Below is a fairly comprehensive selection of the code that I added to the project. There is code from each section of the project that I added to. For all code I have not added I have tried to include code that is very similar. For example, I have not included all the code that I added to View Models. However, I have added the PrescriptionViewVM View Model. This includes examples of both the data annotation validations and the validations present in the validate() method. These are all the validations that will be found in View Models. I also have not included the Fluid/edit View. It is a long page of code and is very similar in content to the two examples from the Case/edit page I have included.

The other point to note is that whilst I have tried only to include code that I have added, there will inevitably be code created by the previous developer. A good example is the CaseAttachmentService. A lot of the code here is mine. However, this was an example of adapting, rather than creating, a method. The table in Section C.5 gives an overview of exactly what I changed and how I changed it. Finally, there is also code that I did not write out line by line but generated using Entity Developer. I have noted which code this is.

### E.1 NHS.Fluid.Model

GlobalSettingModel (generated originally by Entity Framework )

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Collections.Specialized;

namespace NHS.Fluid.Model
{
    public partial class GlobalSetting : IEntity
    {
    }
}
```

## Global Setting

GlobalSettingModel.Generated (generated by Entity Framework)

```
//-----  
// This is auto-generated code.  
//-----  
// This code was generated by Devart Entity Developer tool using Entity Framework DbContext template.  
// Code is generated on: 08/08/2018 09:40:39  
//  
// Changes to this file may cause incorrect behavior and will be lost if  
// the code is regenerated.  
//-----  
  
using System;  
using System.Collections;  
using System.Collections.Generic;  
using System.Collections.Specialized;  
  
namespace NHS.Fluid.Model  
{  
    /// <summary>  
    /// There are no comments for NHS.Fluid.Model.GlobalSetting in the schema.  
    /// </summary>  
    public partial class GlobalSetting {  
  
        public GlobalSetting()  
        {  
            OnCreated();  
        }  
  
        #region Properties
```

```
/// <summary>
/// There are no comments for Id in the schema.
/// </summary>
public virtual int Id
{
    get;
    set;
}
/// <summary>
/// There are no comments for CreatedAt in the schema.
/// </summary>
public virtual global::System.DateTime CreatedAt
{
    get;
    set;
}
/// <summary>
/// There are no comments for ModifiedAt in the schema.
/// </summary>
public virtual global::System.DateTime ModifiedAt
{
    get;
    set;
}
/// <summary>
/// There are no comments for IsActive in the schema.
/// </summary>
public virtual bool IsActive
{
    get;
```

```
        set;
    }
    /// <summary>
    /// There are no comments for SettingName in the schema.
    /// </summary>
    public virtual string SettingName
    {
        get;
        set;
    }

    /// <summary>
    /// There are no comments for SettingValue in the schema.
    /// </summary>
    public virtual string SettingValue
    {
        get;
        set;
    }
}
#endregion
#region Extensibility Method Definitions
partial void OnCreated();
#endregion
}
}
```

GlobalSettingItem (generated originally by Entity Framework – the file also contains changes that I made manually)

```
using System.ComponentModel.DataAnnotations;

namespace NHS.Fluid.Contracts
{
    [MetadataType(typeof(GlobalSettingItemMetaData))]
    public partial class GlobalSettingItem
    {
        internal sealed class GlobalSettingItemMetaData
        {
            public string Id { get; set; }

            public string SettingValue { get; set; }
        }
    }
}
```

GlobalSettingItem.Generated (generated by Entity Framework)

```
//-----
// This is auto-generated code.
//-----
// This code was generated by Devart Entity Developer tool using Data Transfer Object template.
// Code is generated on: 08/08/2018 09:40:39
//
// Changes to this file may cause incorrect behavior and will be lost if
```

```
// the code is regenerated.
//-----

using System.Collections.Generic;
using System.Runtime.Serialization;

namespace NHS.Fluid.Contracts
{
    [DataContractAttribute(IsReference=true)]
    public partial class GlobalSettingItem : IEntityItem
    {
        #region Constructors
        public GlobalSettingItem() {
        }
        public GlobalSettingItem(int id, global::System.DateTime createdAt, global::System.DateTime modifiedAt, bool isActive, string
settingName, string settingValue) {

            this.Id = id;
            this.CreatedAt = createdAt;
            this.ModifiedAt = modifiedAt;
            this.IsActive = isActive;
            this.SettingName = settingName;
            this.SettingValue = settingValue;
        }
        #endregion
        #region Properties
        [DataMember]
        public int Id { get; set; }
        [DataMember]
        public global::System.DateTime CreatedAt { get; set; }
```

```
[DataMember]
public global::System.DateTime ModifiedAt { get; set; }
[DataMember]
public bool IsActive { get; set; }
[DataMember]
public string SettingName { get; set; }
[DataMember]
public string SettingValue { get; set; }
#endregion
}
}
```

UserCaseItem (lines 87-150)

```
public decimal PotassiumInAdditive(int? additiveId, int volume)
{
    if (additiveId == 1)
    {
        return ((Convert.ToDecimal(volume) / 1000) * 20);
    }
    else if (additiveId == 2)
    {
        return ((Convert.ToDecimal(volume) / 1000) * 40);
    }
    return 0;
}
public decimal PotassiumDose
{
    get
    {
        if (Prescriptions == null || !Prescriptions.Any())
```



```
{
    return 0;
}
decimal count = 0;

foreach (PrescriptionItem item in Prescriptions)
{
    count = count + (PotassiumInAdditive(item.AdditiveId, item.Volume) * (Convert.ToDecimal(item.Rate ?? 1)) / 1000);
}

decimal countInFluid = Prescriptions.Sum(x => x.FluidType.Potassium * ((Convert.ToDecimal(x.Rate ?? 1)) / 1000));
return countInFluid + count;
}
}
public decimal PotassiumByWeight
{
    get
    {
        var totalDose = PotassiumDose;
        if (totalDose == 0 || Case.Weight <= 0)
        {
            return 0;
        }
        else
        {
            return totalDose / Case.Weight;
        }
    }
}
```

```
}
```

## E.3 NHS.Fluid.AutoMapper

## DataTierProfile

```
CreateMap<GlobalSetting, GlobalSettingItem>();  
CreateMap<GlobalSettingItem, GlobalSetting>().Ignore(x => x.CreatedAt).Ignore(x => x.ModifiedAt);
```

## E.4 NHS.Fluid.Persistence

## GlobalSettingPersistenceManager

```
using NHS.Fluid.Contracts;  
using NHS.Fluid.Contracts.Criteria;  
using NHS.Fluid.Contracts.PersistenceInterfaces;  
using NHS.Fluid.Model;  
using System;  
using System.Collections.Generic;  
using System.Data.Entity;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
  
namespace NHS.Fluid.Persistence  
{  
    public class GlobalSettingPersistenceManager : GenericPersistenceManager<GlobalSetting>, IGlobalSettingPersistenceManager  
    {  
        public GlobalSettingPersistenceManager(DbContext context) : base(context) { }  
    }  
}
```

```
}  
}
```

#### FluidTypePersistenceManager (lines 38-51)

```
if (criteria is ISortableCriteria && !String.IsNullOrEmpty(criteria.SortField))  
{  
    query.AddSortCriteria(new FieldSortCriteria<FluidType>(criteria.SortField, criteria.SortAscending ? SortDirectionEnum.Ascending :  
SortDirectionEnum.Descending));  
}
```

#### E.5 NHS.Fluid.Services

##### CaseAttachmentService (lines 24-63)

```
public bool StoreLineAttachment(CaseAttachmentItem item, byte[] file, string fileName, string extension, string contentType)  
{  
    byte[] newFile = null;  
    // check it is an image  
    if (imageTypes.Contains(contentType.ToLower()))  
    {  
        // take byte[] and create thumbnail  
        Bitmap bmp = (Bitmap)((new ImageConverter()).ConvertFrom(file));  
        Image thumbnail = bmp.GetThumbnailImage(100, 100, () => false, IntPtr.Zero);  
  
        ImageConverter converter = new ImageConverter();  
        newFile = (byte[])converter.ConvertTo(thumbnail, typeof(byte[]));  
    }  
    // end up with a byte[] or thumbnail image  
    var storedFile = fileService.CreateSecure(file, "Cases", fileName, extension, contentType);  
    if (storedFile != null)
```

```
{
    item.StorageFileId = storedFile.Id;
    if (newFile != null && newFile.Length > 0)
    {
        var thumbnailStoredFile = fileService.CreateSecure(newFile, "Cases", fileName, extension, contentType);
        item.ThumbnailFileId = thumbnailStoredFile.Id;
    }
    Save(item);
    return true;
}
return false;
}
```

#### GlobalSettingService

```
GlobalSettingService
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using NHS.Fluid.Contracts;
using NHS.Fluid.Contracts.Criteria;
using NHS.Fluid.Contracts.PersistencelInterfaces;
using NHS.Fluid.Contracts.ServiceInterfaces;
using NHS.Fluid.Model;

namespace NHS.Fluid.Services
{
```

```
public class GlobalSettingService : BaseService<IPersistence<GlobalSetting>, GlobalSettingItem, GlobalSetting, GlobalSettingSearchCriteria>,
IGlobalSettingService
{
    public GlobalSettingService(IGlobalSettingPersistenceManager persistenceManager) : base(persistenceManager) { }
    public IList<GlobalSettingItem> Search(FluidTypeSearchCriteria criteria, out int totalResults)
    {
        throw new NotImplementedException();
    }
}
```

## E.6 NHS.Fluid.PersitenceSupport

## PersistenceModule (line 25)

```
Kernel.Bind<IGlobalSettingPersistenceManager>().To<GlobalSettingPersistenceManager>().InRequestScope();
```

## E.7 NHS.Fluid.WebApp: Controller

## GlobalSettingController

```
using AutoMapper;
using NHS.Fluid.Contracts;
using NHS.Fluid.Contracts.ServiceInterfaces;
using NHS.Fluid.WebApp.Models;
using Ninject.Extensions.Logging;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
```

```
namespace NHS.Fluid.WebApp.Controllers
{
    [Authorize(Roles = "SysAdministrator")]
    public class GlobalSettingController : BaseController
    {
        private readonly IGlobalSettingService globalSettingService;
        public GlobalSettingController(ILogger loggerService, IUserSettingService userSettingService, IGlobalSettingService globalSettingService) :
        base(loggerService, userSettingService)
        {
            this.globalSettingService = globalSettingService;
        }
        // GET: GlobalSetting
        public ActionResult Index()
        {
            GlobalSettingViewVM model = new GlobalSettingViewVM() { Id = 1 };
            GlobalSettingItem item = globalSettingService.GetById(1);
            if (item == null)
            {
                return RedirectToAction("PageNotFound", "Public");
            }
            Mapper.Map<GlobalSettingItem, GlobalSettingViewVM>(item, model);
            return View("Index", model);
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Index(GlobalSettingViewVM model)
        {
            if (ModelState.IsValid)
```

```
{
    var itemToPersist = Mapper.Map<GlobalSettingViewVM, GlobalSettingItem>(model);
    GlobalSettingItem persistedItem = globalSettingService.Save(itemToPersist);
    if (persistedItem == null)
    {
        ModelState.AddModelError("CustomError", "There has been a problem with saving the item, the error has been logged and will be investigated.");
    }
    else
    {
        return RedirectToAction("Index", "GlobalSetting");
    }
}
return View("Index", model);
}
```

#### Case Controller – Attach()

```
CaseAttachmentItem attachment = new CaseAttachmentItem()
{
    CaseId = model.Id,
    Description = model.Description,
    IsFeedback = model.IsFeedback
};
```

#### MyCaseController – Constructor (lines 20-43)

```
public class MyCaseController : BaseController
```

```
{
    private readonly string CRITERIASESSIONHANDLE = "CaseCriteria";

    private readonly ICaseService caseService;
    private readonly ICaseAttachmentService caseAttachmentService;
    private readonly IStorageFileService storageFileService;
    private readonly IUserCaseService userCaseService;
    private readonly IFluidTypeService fluidTypeService;
    private readonly IAdditiveService additiveService;
    private readonly IPrescriptionService prescriptionService;
    private readonly IGlobalSettingService globalSettingService;

    public MyCaseController(ILogger loggerService, IUserSettingService userSettingService, ICaseService caseService, ICaseAttachmentService
caseAttachmentService, IStorageFileService storageFileService, IUserCaseService userCaseService, IFluidTypeService fluidTypeService,
IGlobalSettingService globalSettingService, IAdditiveService additiveService, IPrescriptionService prescriptionService) : base(loggerService,
userSettingService)
    {
        this.caseService = caseService;
        this.caseAttachmentService = caseAttachmentService;
        this.storageFileService = storageFileService;
        this.userCaseService = userCaseService;
        this.fluidTypeService = fluidTypeService;
        this.additiveService = additiveService;
        this.prescriptionService = prescriptionService;
        this.globalSettingService = globalSettingService;
    }
}
```

MyCase Controller (lines 172-220)

```
public ActionResult GetSpecificFeedback(int id)
{
}
```



```
var myCase = userCaseService.GetById(id);

CaseItem feedbackCase = new CaseItem();
if (myCase.IsFinished)
{
    feedbackCase = caseService.GetInclude(myCase.CaseId);
}
return PartialView("_FeedbackSpecific", feedbackCase);
}

public ActionResult GetGenericFeedback(int id)
{
    var myCase = userCaseService.GetById(id);

    string feedback = "";

    if (myCase.IsFinished)
    {
        feedback = globalSettingService.GetById(1).SettingValue;
    }
    return PartialView("_FeedbackGeneric", feedback);
}

public ActionResult GetUserFeedback(int id)
{
    var myCase = userCaseService.GetById(id);
    var model = new UserFeedbackVM();
    model.IsFinished = myCase.IsFinished;
    model.Feedback = myCase.UserFeedback;
```

```
        return PartialView("_UserFeedbackEditor", model);
    }
    public ActionResult SaveUserFeedback(int id, string feedback)
    {
        var myCase = userCaseService.GetById(id);
        myCase.UserFeedback = feedback;
        userCaseService.Save(myCase);
        return Json(new { Success = true });
    }
```

#### MyCase Controller – PostPrescriptionPartial() (lines 239-277)

```
if (model.IsBolus)
{
    var fluidType = fluidTypeService.GetById(model.FluidTypeId);

    // this is covered by the two below because NS and CSL are the only fluids that have the correct amounts of Sodium and Potassium
in them
    //if (!new List<string>() { "NS", "CSL" }.Contains(fluidType.Abbreviation))
    //{
    //    ModelState.AddModelError("CustomError", "Bolus requires Normal Saline (NS) or Hartmann's (CSL).");
    //}
    if (fluidType.Sodium <= 129)
    {
        ModelState.AddModelError("CustomError", "Bolus cannot be selected when total Sodium prescribed is less than 129 mmol/L.");
    }
    if (fluidType.Potassium > 20)
    {
        ModelState.AddModelError("CustomError", "Bolus cannot be selected when total Potassium prescribed is more than 20
mmol/L.");
    }
}
```

```
    }
    var myCase = userService.GetInclude(model.UserCaseId);
    var currentCase = caseService.GetInclude(myCase.CaseId);

    if (currentCase.NEWScore > 4 && !(model.Volume <= model.Rate / 4))
    {
        ModelState.AddModelError("CustomError", "If Bolus is selected and NEWS score is 5 or above, rate must be such that the patient
recieves the total volume in 15 minutes or less.");
    }

    if (model.Volume < 250 || model.Volume > 500)
    {
        ModelState.AddModelError("CustomError", "If Bolus is selected volume must be between 250ml and 500ml.");
    }
}
```

#### MyCase Controller - PostPrescriptionPartial (lines 346-371)

```
public ActionResult RenderImageThumbnail(int caseId, int attachmentId)
{
    var singleCase = caseService.GetInclude(caseId);
    if (singleCase == null)
    {
        return new EmptyResult();
    }
    if (singleCase.CaseAttachments.AnyOrNotNull())
    {
        var attachment = singleCase.CaseAttachments.FirstOrDefault(x => x.Id == attachmentId);
        if (attachment != null)
        {

```

```
        var fileData = storageFileService.GetSecureItemData(attachment.ThumbnailFileId.GetValueOrDefault());

        if (fileData != null)
        {
            string fileName = attachment.StorageFile.FileName;
            string mimeType = MimeMapping.GetMimeMapping(fileName);
            return File(fileData, mimeType, fileName);
        }
    }
}
```

## E.8 NHS.Fluid.WebApp: Models (View Models)

## CaseViewVM (lines 55-61)

```
[Required(ErrorMessage = "NEWS Score is required")]
[DisplayName("NEWS Score")]
public decimal NEWSScore
{
    get;
    set;
}
```

## CaseViewVM (lines 94-120)

```
#region IValidatableObject Members

public System.Collections.Generic.IEnumerable<ValidationResult> Validate(ValidationContext validationContext)
{
    var results = new List<ValidationResult>();
}
```

```
        if (Description.Length > 8000)
        {
            results.Add(new ValidationResult("Description cannot be more than 8000 characters"));
        }
        if (Feedback.Length > 8000)
        {
            results.Add(new ValidationResult("Feedback cannot be more than 8000 characters"));
        }
        if (NEWSScore < 1 || NEWSScore > 15)
        {
            results.Add(new ValidationResult("NEWS score cannot be less than 1 or more than 15"));
        }
        return results;
    }

#endregion
```

#### PrescriptionViewVM

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using DataAnnotationsExtensions;
using NHS.Fluid.Contracts;

namespace NHS.Fluid.WebApp.Models
{
```

```
public class PrescriptionViewVM : IValidatableObject
{
    public int Id
    {
        get;
        set;
    }

    public int UserCaseId { get; set; }
    public bool CanPrescribe { get; set; }
    public bool IsMaintenance { get; set; }

    public bool IsDeficit { get; set; }

    public bool IsOngoing { get; set; }

    public bool IsBolus { get; set; }

    public bool IsRateStat { get; set; }

    [Required(ErrorMessage = "Volume is required")]
    [Min(1, ErrorMessage = "Volume must be at least 1ml")]
    public int Volume { get; set; }

    [Required(ErrorMessage = "Fluid Type is required")]
    [Min(1, ErrorMessage = "Fluid Type is required")]
    public int FluidTypeId { get; set; }

    public int? AdditiveId { get; set; }
```

```
[Range(10, 6000, ErrorMessage = "Rate must be between 10 and 6000 ml/hr")]
public int Rate { get; set; }

public IList<SelectListItem> Additives { get; set; }

#region IValidatableObject Members

public System.Collections.Generic.IEnumerable<ValidationResult> Validate(ValidationContext validationContext)
{
    var results = new List<ValidationResult>();
    if (IsBolus && AdditiveId != -1)
    {
        results.Add(new ValidationResult("You cannot select Bolus if an Additive has been added."));
    }
    if (!IsMaintenance && !IsDeficit && !IsOngoing && !IsBolus)
    {
        results.Add(new ValidationResult("You must select at least one of the Indication options for the prescription."));
    }
    if ((IsDeficit || IsOngoing || IsBolus) && Rate < 84)
    {
        results.Add(new ValidationResult("With a specified rate of less than 84 ml/hr the Indication must be 'Maintenance' ONLY."));
    }
    if ((IsDeficit || IsOngoing) && !IsMaintenance)
    {
        results.Add(new ValidationResult("'Ongoing' or 'Deficit' (or both) must be always be selected with 'Maintenance.'));
    }
    if (IsMaintenance && (!IsOngoing && !IsDeficit) && (Rate > 84 && Rate < 1000))
    {

```

```
        results.Add(new ValidationResult("'Maintenance' together with 'Ongoing' or 'Deficit' (or both) must be selected with a specified rate of between 84 and 999 ml/hr."));
    }
    if ((IsOngoing || IsDeficit) && (Rate <= 84 || Rate >= 1000))
    {
        results.Add(new ValidationResult("'Maintenance' together with 'Ongoing' or 'Deficit' (or both) must be selected with a specified rate of between 84 and 999 ml/hr."));
    }

    //if (Rate > 1000 && !IsBolus)
    //{
    //    results.Add(new ValidationResult("With a specified rate of greater than 1000 ml/hr only 'Bolus' can be selected."));
    //}

    if (Rate > 1000 && (IsMaintenance))
    {
        results.Add(new ValidationResult("With a specified rate of greater than 1000 ml/hr only 'Bolus' can be selected."));
    }
    if (Rate > 1000 && (IsDeficit))
    {
        results.Add(new ValidationResult("With a specified rate of greater than 1000 ml/hr only 'Bolus' can be selected."));
    }
    if (Rate > 1000 && (IsOngoing))
    {
        results.Add(new ValidationResult("With a specified rate of greater than 1000 ml/hr only 'Bolus' can be selected."));
    }
    if (Volume != 500 && Volume != 1000 && AdditiveId != -1)
    {
        results.Add(new ValidationResult("Additives can only be added to Volumes of 500ml or 1000ml."));
    }
}
```



```
    }  
    for (int i = 0; i < results.Count; i++)  
    {  
        var item = results[i];  
  
        for (int t = 0; t < results.Count; t++)  
        {  
            if (t == i)  
            {  
                continue;  
  
            } else if (item.ToString().Equals(results[t].ToString()))  
            {  
                results.RemoveAt(i);  
            }  
        }  
    }  
    return results;  
#endregion  
}  
}
```

## E.9 NHS.Fluid.WebApp: Views

Case/Edit

Lines (59-65)

```
<div class="form-group">
```

```

        @Html.LabelFor(model => model.NEWScore, new { @class = "col-xs-4 col-sm-3 col-lg-2 control-label" })
        <div class="col-xs-8 col-lg-10">
            @Html.TextBoxFor(model => model.NEWScore, new { placeholder = "NEWS Score", @class = "form-control" })
            @Html.ValidationMessageFor(model => model.NEWScore)
        </div>
    </div>

```

Lines (77-83)

```

<div class="form-group">
    @Html.LabelFor(model => model.Feedback, new { @class = "col-xs-12 col-sm-3 col-lg-2 control-label" })
    <div class="col-xs-12 col-sm-8 col-lg-10">
        @Html.TextAreaFor(model => model.Feedback, new { placeholder = "Narrative", @class = "form-control" })
        @Html.ValidationMessageFor(model => model.Feedback)
    </div>
</div>

```

\_FluidTypeRow

```

@model NHS.Fluid.WebApp.Models.BaseRowVM<FluidTypeItem>
@using NHS.Fluid.Contracts
@using NHS.Fluid.WebApp.Models

<tr role="row">
    <td>
        @Model.Item.Abbreviation
    </td>
    <td>
        @Model.Item.Description
    </td>
    <td class="text-right">
        @Model.Item.Sodium
    </td>
</tr>

```

```

</td>
<td class="text-right">
    @Model.Item.Chloride
</td>
<td class="text-right">
    @Model.Item.Potassium
</td>
<td class="text-right">
    @Model.Item.Bicarbonate
</td>
<td class="text-right">
    @Model.Item.Glucose
</td>
<td class="text-right">
    @Model.Item.Magnesium
</td>
<td class="text-right">
    @Model.Item.Calcium
</td>
<td class="text-center">
    @if (Model.Item.CanHaveAdditives)
    {
        <i class="far fa-check f00c fa-2x" style="color:#1c84c6"></i>
    }
</td>
<td class="searchRowBtns col-xs-2">
    <a href="@Url.Action("Edit", "FluidType", new { Id = Model.Item.Id })" class="btn btn-sm btn-primary" data-toggle="tooltip" data-
placement="top" title="Edit"><i class="fa fa-pencil"></i></a>
</td>
</tr>

```

\_FluidTypeTable (line 30-37)

```
<tbody>
  @foreach (var item in Model.Items)
  {
    Model.RowVM = new BaseRowVM<FluidTypeItem>();
    Model.RowVM.Item = item;
    Html.RenderPartial("_FluidTypeRow", Model.RowVM);
  }
</tbody>
```

Fluid/Index (lines 31-33)

```
<div class="table-responsive widget-table">
  @Html.Partial("_FluidTypeTable", Model.TableVM)
</div>
```

GlobalSetting/Index

```
@model NHS.Fluid.WebApp.Models.GlobalSettingViewVM
@using NHS.Fluid.WebApp.Models
@using NHS.Fluid.Contracts
```

```
@{
  ViewBag.Title = "GlobalSetting";
}
```

```
<div class="wrapper wrapper-content animated fadeInDown">
  <div class="row">
    <div class="col-xs-12">
      <div class="ibox float-e-margins">
```

```

<div class="ibox-title">
  <h5>&nbsp;&nbsp; </h5><a href="@Url.Action("Index", "GlobalSetting")"></a><h5>System Setup</h5>
  <div class="ibox-tools">
    </div>
</div>
<div class="ibox-content">
  @using (Html.BeginForm("Index", "GlobalSetting", FormMethod.Post, new { @class = "form-horizontal nomargin", role = "form" }))
  {
    @Html.AntiForgeryToken()
    @Html.ValidationSummary(true)
    @Html.ValidationMessage("CustomError");
    @Html.HiddenFor(model => model.Id)

    <div class="form-group">
      @Html.LabelFor(model => model.SettingValue, new { @class = "col-xs-12 col-sm-3 col-lg-2 control-label" })
      <div class="col-xs-12 col-lg-6">
        @Html.TextAreaFor(model => model.SettingValue, new { placeholder = "Narrative", @class = "form-control" })
        @Html.ValidationMessageFor(model => model.SettingValue)
      </div>
    </div>

    <div class="form-group">
      <div class="col-xs-12 col-sm-11 col-lg-8">
        <!-- this url.action just redirects to the next page right -->
        <a href="@Url.Action("Index", "GlobalSetting")" class="btn btn-default pull-right" style="margin-left:5px;" title="Case
Administration">Cancel</a>
        <button class="btn btn-primary pull-right" type="submit">
          Save
        </button>
      </div>
    </div>
  }

```

```
        </div>
    }
    </div>
</div>
</div>
</div>

</div>
@section pagespecific {
    @Scripts.Render("~/scripts/globalsetting/index.js")
    @Scripts.Render("~/plugins/attachments")
}
```

#### MyCase/\_FeedbackGeneric

```
@using NHS.Fluid.WebApp.Models
@using NHS.Fluid.Core
@using NHS.Fluid
@model string
```

```
<div>
    @if (!String.IsNullOrEmpty(@Model)){
        <h3>General Feedback</h3>
    }
    <p>
        @Model
    </p>
</div>
```

MyCase/\_FeedbackSpecific

```
@using NHS.Fluid.WebApp.Models
@using NHS.Fluid.Core
@using NHS.Fluid
@model NHS.Fluid.Contracts.CaseItem

<div>
    @if (!String.IsNullOrEmpty(@Model.Feedback))
    {
        <h3>Case Feedback</h3>
        <p>
            @Html.Raw(@Model.Feedback)
        </p>
        <div class="row space-15">
            @foreach (var attachment in Model.CaseAttachments.Where(x => x.StorageFile.IsImage && x.IsFeedback))
            {
                int imageId = 1;
                <div class="col-xs-12 col-sm-6 text-center space-15">
                    <span class="space-15" style="display:block;font-size:12px;font-weight:bold;"> <a href="@Url.Action("DownloadAttachment",
"MyCase", new { caseId = Model.Id, attachmentId = attachment.Id })">@attachment.Description</a></span>
                    @{
                        if (attachment.ThumbnailFile != null)
                        {
                            <a href="@Url.Action("RenderImage", "MyCase", new { caseId = Model.Id, attachmentId = attachment.Id })" data-
lightbox="image-@imageId" data-title="@attachment.Description"></a>
                        }
                        else
                        {

```

```

        <a href="@Url.Action("RenderImage", "MyCase", new { caseId = Model.Id, attachmentId = attachment.Id})" data-
lightbox="image-@imageId" data-title="@attachment.Description"></a>
    }
}
@{
    imageId++;
}
</div>
}
</div>
}
</div>

```

## MyCase/View (lines 48-53)

```

<div class="form-group">
    <label class="col-xs-12 col-sm-4 control-label">NEWS Score</label>
    <div class="col-xs-12 col-sm-8 col-lg-6">
        <p class="form-control-static">@Model.NEWScore</p>
    </div>
</div>

```

## MyCase/View (lines 96-119)

```

<div class="row space-15">
    @foreach (var attachment in Model.CaseAttachments.Where(x => x.StorageFile.IsImage && !x.IsFeedback))
    {
        int imageId = 1;
        <div class="col-xs-12 col-sm-6 text-center space-15">

```



```

        <span class="space-15" style="display:block;font-size:12px;font-weight:bold;"> <a
href="@Url.Action("DownloadAttachment", "MyCase", new { caseId = Model.Id, attachmentId = attachment.Id
})">@attachment.Description</a></span>
        @{
            if (attachment.ThumbnailFile != null)
            {
                <a href="@Url.Action("RenderImage", "MyCase", new { caseId = Model.Id, attachmentId = attachment.Id})"
data-lightbox="image-@imageId" data-title="@attachment.Description"></a>
            }
            else
            {
                <a href="@Url.Action("RenderImage", "MyCase", new { caseId = Model.Id, attachmentId = attachment.Id})"
data-lightbox="image-@imageId" data-title="@attachment.Description"></a>
            }
        }
        @{
            imageId++;
        }
    </div>
}
</div>

```

MyCase/Index (line 129-191)

```

<div class="row">
    <div class="col-xs-12">
        <div class="ibox float-e-margins">

```

```

<div class="ibox-content form-horizontal">

    <div class="row">
        <label class="col-xs-2 col-sm-1"><i style="color:lightblue" class="fal fa-tint fa-4x"></i></label>
        <div class="col-xs-10 col-sm-11">
            <p class="form-control-static" style="font-size:20px;font-weight:bold;margin-top:15px;">Fluid Chart</p>
        </div>
    </div>

    <div id="prescriptionEditorContainer">
        @Html.Partial("_PrescriptionEditor", Model.PrescriptionModel)
    </div>

    <div class="row space-30">
        <h3><i class="fas fa-clipboard-check fa-lg"></i>&nbsp;&nbsp;&nbsp;Current Fluid Balance Chart</h3>
        <hr />

        <div id="prescriptionPartial" class="partialContents" data-url='@Url.Action("GetPrescriptions", "MyCase", new { id =
Model.PrescriptionModel.UserCaseId })'>
            <div class="alert alert-info" role="alert"><i class="fa fa-sync fa-spin"></i>&nbsp;&nbsp;&nbsp;Loading Fluid Balance Chart</div>
        </div>
    </div>
    <div class="row space-30">
        <div id="specialInstructionEditorContainer">
            @if (Model.MyCase.Prescriptions != null && Model.MyCase.Prescriptions.Any())
        {
            @Html.Partial("_UserCaseEditor", Model.UserCaseModel);
        }

    </div>

```

```

        </div>
        <div class="row space-30">
            <div id="feedbackGenericPartial" class="partialContents col-md-6" data-url='@Url.Action("GetGenericFeedback",
"MyCase", new { id = Model.UserCaseModel.Id })'>
                </div>

            <div id="feedbackSpecificPartial" class="partialContents col-md-6" data-url='@Url.Action("GetSpecificFeedback",
"MyCase", new { id = Model.UserCaseModel.Id })'>
                </div>
            </div>
            <div class="row space-30">
                <div id="userFeedbackPartial" class="partialContents col-xs-12" data-url='@Url.Action("GetUserFeedback", "MyCase", new
{ id = Model.UserCaseModel.Id })'>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>

```

MyCase/\_PrescriptionList (line 119)

```
<li class="list-group-item ">Potassium - @Model.PotassiumByWeight.ToString("0.##") mmol/kg/day</li>
```

E.10 NHS.Fluid.WebApp: Scripts

MyCase/index (lines 19-34)

```

$('#FluidTypeId').selectize({
    valueField: 'Id',
    labelField: 'Description',

```

```
searchField: ['Description'],
maxOptions: 10,
create: false,
openOnFocus: false,
render: {
  option: function (item, escape) {
    return '<div>' +
      '<span class="" style="font-weight:bold">' +
      '<span class="">' + item.Description + '</span>' +
      '</span>' +
      '</div>';
  }
},
```

MyCase/index (lines 146-168)

```
$.ajax({
  cache: false, url: posturl,
  dataType: "html",
  data: _form.serialize(),
  error: function (jqXHR, textStatus, errorThrown) {
    alert(errorThrown);
  },
  success: function (data) {
    try {
      var response = JSON.parse(data);
      GrowlSuccess("Operation Complete", "Prescription finished successfully");
      LoadBlankPrescriptionEditor();
      ReloadPartialContents($('#feedbackGenericPartial'));
      ReloadPartialContents($('#feedbackSpecificPartial'));
      ReloadPartialContents($('#userFeedbackPartial'));
    }
  }
});
```

```
        LoadUserCaseEditor();  
    }  
    catch (e) {  
        InitializeUserCaseEditor(data, true);  
    }  
}  
});
```