Feeding Dashboard

# Aims and Objectives

The main aim of this project is to create an application, with the primary goal of identifying which patients need referring to a dietitian. The Critical Care Unit (CCU) is a branch of the hospital which is underfunded: prioritising those most in need is key. Using Electron as a base framework, combined with Next.js and React.js for the rendering, it is possible to create an application which will run efficiently on all operating systems.

The critical path, detailed in the Project Planning phase, will be continually monitored to prevent any potential delays. This involves identifying key milestones in the project. Said milestones will be used as an easy way to measure how much of the project has been completed: ensuring the project will be completed within the planned timeframes. A risk assessment will be taken to ensure that if issues do arise, there is at least one fallback option that can help to deliver the project on time. This is a time sensitive project which has a definitive end point, finishing behind schedule is not an option: hence the severity of maintaining the critical path. Without this software being delivered on time; patients are at risk of going untreated, and the CCU is forced to use more of its budget to spread themselves thinner. Proactive measures will be taken once if the critical path suffers deviation. As an example, the project will be planned for a 10% leniency in duration. This ensures that if a large, unforeseen issue is discovered, it can be met with accordance without disrupting the deadline. A simple solution for these issues can be in the form of focus-group sessions, with emphasis on diagnosing specific problems, to switching the internal structure allowing for the best use of resources. Following this, the project should have no issue with the due date of 17th April.

Creating an application, capable of running on multiple operating systems, is the first major step. With their limited budget, it is unknown how much can be spent on proprietary licences. Allowing for access to multiple operating systems gives the CCU freedom to expand, or constrict, their technology budget according to their needs, not the needs of the software. Using the same hardware, Virtual Machines will allow for collection of metric data on different operating systems. Examples of metric data collected include Memory Usage and speed at which the Machine Learning algorithm returns a value. The chosen approach for this objective: using a mixture of Electron and Next.js. This will free the project team, letting them focus on the project itself, instead of the different versions traditionally needed. While it would be possible to develop this as a Web Application, given the confidentiality of the CCU, it is important to note the security implications. Restricting this to a desktop application allows entire separation from the internet, widely restricting the possibility of a data breach. The use of Electron as a Multi-Operating-System framework means that there is not much consideration needed to add extra platform support. This boost in efficiency helps to deliver the project on time because code errors are no longer unique to the OS, but project instead.

Given the CCU is dealing with too many patients for them to manage without assistance, it is crucial that the application is capable of accurately predicting those who would benefit from seeing a dietitian. This will be accomplished by allowing users to upload new csv files, containing patient data; then analysing it with the bespoke Machine Learning algorithm. Patients who meet the criteria will be automatically flagged for referral, upon review. It is vital that this algorithm is validated as having a high accuracy, there is no sense delivering an extra feature which will just further complicate the CCU worker. Accuracy can be clearly measured by determining the ratio of perfectly categorised patients to those who were mischaracterised. This objective will be achieved by executing Python code within the Electron application, then outputting the returned python data to graphs for easy viewing.

Creating and exporting graphs and forms of patient data is necessary for continued operations. It allows the workers in the CCU to easily access the data one time, then continue carrying it throughout the day for easy references. Examples of graphs and forms generated include individualised patient overviews, a full list of referrals for the dietitian, and a full list of patients currently admitted. These graphs and forms will also have 3 quantifiable metrics, visual aesthetics, accessibility, and exportability, their importance is paramount. Without the styling having careful consideration, there is a risk of it being too complicated: left unused. To ensure that these are met, different design options will be presented to a small focus group, independent to the development group. This focus group will consist of people from a variety of age brackets and technological ability, ensuring fair accessibility for all users. Options should also be included for different themes, with classic examples being Light and Dark mode, including high contrast mode and options for text size. As previously mentioned, the CCU is spread quite thin. Giving them the ability to export these generated reports would mean they are able to spend less time running back and forth to a computer, and more time spent with the patients who need them. To ensure that this doesn’t cause the projects time constraint to dwindle, the designs shown to the focus group will take place before any development begins. This allows for any revisions to be made before concrete steps have been taken.

# Literature Review

## Methodology

The linear methodology, first presented by Royce (1970), documented how was presumed best to handle Software Development Lifecycles (SDLC) of large-scale applications. His original theory stems from the idea that development of any size can be split into two hypothetical partitions: Analysis, and Coding. It was believed that for small scale projects, it “is in fact all that is required” to maintain successful development (Royce 1970).

If larger scale projects attempt to adopt this small-scale practice, multitudes of risks enter the scenario. One of the biggest is the potential for large scale setbacks. In cases where the product is not what the customer envisioned; projects often must be reset entirely. This can extra cost time, equating to lost money and business reputation (Royce 1970). For the development of an application for the healthcare industry, this cannot be an option. Delayed deployment would result in a continued overwhelmed medical support staff, and patients who are possibly going without treatment they desperately need.

Given larger projects, Royce suggested his linear model. This aimed to split analysis and coding phases into seven sequential and separated steps. Analysis was given two precursors, System Requirements and Software Requirements. Coding phases were also given two additions in the form of Program Design and Testing, the former before and the latter afterwards. These two precursory stages were believed to be the root of any issues in the final product (Royce 1970).

It wasn’t until Bell and Thayer (1976) questioned this principle that the methodology was coined “Waterfall”, largely based on its theoretical design structure. Their analysis of this methodology under the microscope proved Royce’s initial theory correct, most errors in the SDLC could be attributed to a failure to understand the Requirements. These failures could commonly be defined by subcategories: missing, incomplete, inadequate, or unclear. (Bell and Thayer 1976)

Waterfall is not without its pitfalls. During early stages in the history of Software engineering, the methodology practices were very similar to that of standard engineering; namely the practice of front-loading design stages to negate time from the development stage. As discussed, the Waterfall method, designed by Royce, attributes a heavy weight on the initial Requirements to the final product’s success. This ignores a crucial element of Software Development that the environment and requirements can continually change.

One of the biggest challenges faced when approaching development with Waterfall was the progressive timeline. Given the severity of the level of planning which must go into a project, when mistakes are made, and phases must be reset, lots of time becomes wasted. While this does provide a slight betterment over the initial Analysis and Coding scheme laid out by Royce, waterfall can still involve being sent back to the previous core-stage of functional development. This emphasised the effects of changes in environment and dynamic requirements of the application. Given waterfalls linear nature, if requirements are to change: the entire project must often be redesigned from the origin point (Royce 1970).

Many mythologies were derived from this initial framework, of which, most sought to fix oversights in how the project viewed Requirements. These were all radicalised and had no real over-arching agreements on how successful Software Engineering projects should exist. This changed in 2001 when a group, including seventeen of the most progressive project planners for software engineering, collected in Utah for a summit to discuss a set of ground rules referred to as the Manifesto for Agile Software Development (Highsmith 2001). These were to be the proceeding best practices to deliver customer orientated light weight project methodologies.

The mantra for The Agile Alliance, signed by all seventeen attendees:

**Individuals and Interactions** over processes and tools,

**Working software over** comprehensive documentation,

**Customer collaboration** over contract negotiation,

**Responding to change** over following a plan.

(Highsmith and Fowler 2001)

Loosely interpreted, the principles define how projects must: continually involve the critical end user, prioritise simplicity and most importantly a low-scale timeline. This aimed to increase development teams focus on the end user and increase flexibility over the pre-existing accepted models’ emphasis on static Requirements (McQuade, Moore and Hunter 2019).

Analysing the effectiveness of these practices came later, at the expense of Dayton and Barnum (2009). Completion of surveys given to the development team and analysis of the product, made researchers able to determine many key factors in the relevance of different principles laid out in the initial Manifesto (Highsmith and Fowler 2001). The primary focus of these surveys was to determine the effect of User Centred Design on the final product and employee attitudes and opinions were to the newly adopted Agile methodology.

They found the biggest stand out of the Agile methodology was the use of end-user collaboration. This often led to developers feeling more engaged with their projects and result in more user-centric design. Employees felt more connected to the end-user through formal collaboration, leading to more nuanced and intuitive solutions for the design. It was also observed that employees’ attitudes to the projects would often shift, more positively, after formal user testing. The physical witnessing of a user attempting their application greatly motivated developers, often making them work much more vigorously as they could better understand challenges associated with the current design (Dayton and Barnum 2009).

Without the formal user testing, employees regularly felt like management would fake feedback. to tweak the application without any pushback from the development team: as this feedback would often be a direct change to the existing understood project requirements. Witnessing these events allowed for real connection to the issues, instead of reading the reports in an email. These findings of Dayton and Barnum (2009) directly correlate with the Manifesto published by Highsmith and Fowler (2001), who stated that “the most effective method of conveying information within the development team is face-to-face”.

This makes the agile methodology a perfectly suited application to the project at hand, developing an application for the CCU to assist in referring patients to a dietitian. Its lightweight nature will allow the project to move at a fast pace, helping to deliver the project within the timescale. The end-client orientation of the methodology will also help to ensure the project delivered will be easily accessible to the members of the health staff, who might not be technologically advanced.

## Existing Solutions

There are a few ways to attack the problem of multi-platform applications. Given different system requirements, they often require vastly different code to create the same application. This form of development would not be acceptable for Agile development. The process would add extended timelines, resulting in three undeniably different projects. To maintain accordance with the principles of Agile, saving time where possible, a single code-base and project would be preferable. This should also help to reduce operational costs, as there are less end-projects to manage.

Many of the biggest application development companies employ this same methodology, using a singular codebase stretched across multiple operating systems. This can be accomplished through the use of frameworks, like Electron. Developed by GitHub; this provides developers a way of creating desktop applications using standard web frameworks. This was originally developed solely for their custom code editor, Atom.

When GitHub decided to release this framework as an Open-Source project, many developers jumped on the opportunity (Electron, 2024). The clear advantages of not having to split focus between multiple projects far outweigh the lessened customisability for systems seen in individualised projects. Not only freeing expenses on development costs but also operational costs. Two of the biggest names who’ve adopted this framework are Discord and Skype, two rivalling communication applications.

By choosing to use this framework, the development team for the CCU application can focus their time on progressing new features and ensuring a high level of accessibility, via continual end-user collaboration. Conversely, choosing to develop three individual applications, without the use of a framework like electron, will result in a protracted timeline three-fold and the potential for unequal applications: leading to confusion in the workers using different versions. This would undercut the main aim for this project, to provide an easy-to-use application for the use of medical staff to aid in dietitian referral.

Other similar projects have been undertaken with results published (Pimenidis, Soomro, Papaleonidas, and Psathas 2023). Their work showed the need for a system like this in critical treatment centres, like the CCU. Typically, these hospital wards can become overrun with patients. This can lead to hospital staff not being able to focus their attention on those who truly need it. Employing this system successfully would eliminate part of the staff’s workflow, or at least dramatically reduce it: helping to resolve some of the congestion seen in the ward’s current state.

## Tools

### Multi-Platform and Interfacing

The chosen Framework for managing the cross-platform support is Electron. As stated, this will allow developers to focus on the important elements of the project requirements, without delegating extra time to ensuring compatibility on multiple platforms. Electrons heavy documentation show possibilities of using Front-End web libraries like React and Next. These will be the User Interface (UI) Frameworks chosen for this project.

Frameworks, such as these, have been widely accepted as the best options for UI development in recent years, with additional libraries like Tailwind providing easy uniformity and customisation. Preloaded with tons of configuration options, these UI frameworks allow for quick development in accordance with the agile methodology: delivering the project faster to the CC, helping to expediate necessary referrals.

### Machine Learning

At the core of this application lies a Machine Learning algorithm used for binary classification, meaning to classify between “is” and “isn’t”, “True”, or “False”: In need of referral, or not in need of referral. Many options for effective binary classification exist, the most relevant for the type of data processed are Search Vector Machine and Ensemble learning methods. These both offer relatively light weight solutions to binary classification and are idealistic under different scenarios.

A comparison of these two models has previously been independently conducted (Hornblower 2024), highlighting the benefits of an Adaptive Boosted ensembles learning method. In a diagnostic setting such as this, where the number of patients in need missed are of the highest importance: an adaptive boosted model typically preforms much better. The aggregation of multiple weak-learners allows the final prediction to be taken at a vote, aiming to provide a more comprehensive prediction based on more discovered relationships between patient data.

The choice of library for this functionality will be SKLearn. They offer a multitude of excellent python functions necessary for machine learning. Given these premade classifiers, it will be much easier to keep in line with Agile development and stay swift in moving through different phases.

# Requirements

## Functional

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Description | Source | Priority | Status | Assigned To | Test Cases |
| F-1 | List All patients | Specification | High | Complete | Callum |  |
| F-2 | Filter Patients | Specification | low | Planned | Callum |  |
| F-3 | View Individual Patients | Specification | High | Complete | Callum |  |
| F-4 | Patient Analysis | Specification | Very High | Complete | Callum |  |
| F-5 | Upload Patient CSV | Specification | High | Complete | Callum |  |
| F-6 | Data Imputing | Specification-inferred | High | Complete | Callum |  |
| F-7 | Generate graphs showing all individuals | Specification | Medium | Complete | Callum |  |
| F-8 | Export Graphs | Recommendation | Medium | Planned | Callum |  |
| F-9 | Run on different platforms | Specification | High | Complete | Callum |  |
| F-10 | Testing | Reccomendation | High | Complete | Callum |  |

A key functionality of the proposed healthcare management system is the ability to display a detailed list of all patients within the system. There should also be options allowing users to filter the results and display who is most in need of seeing a specialised dietitian. This feature is designed to help streamline the patient management of the Critical Care Unit, allowing for better delegation of resources.

The application must also be capable of presenting a comprehensive overview of individual patient data. This helps to give doctors, working in the ward, data in a more accessible way which can be analysed quicker. Aims of presenting individual patients in this manor are largely for quality of life for medical staff, including shorter wait times for patients given their data should be easily printable and ready for transport inside of the CCU.

Without the ability to analyse the patients’ records, the application serves no more purpose than a standard patient record viewing tool. This is where the Machine Learning algorithm takes physiological measurements of patients within the Critical Care Unit, flagging them if they should be referred to a specialist. This step needs to happen before the Operator can filter results, otherwise no new records would be processed for referral.

Operators must be able to upload new patient records for analysis under the new Machine Learning model. The format accepted should be CSV, allowing for easy processing inside the algorithm. Giving Operators the option to add new CSV files means new admissions to the ward can be easily analysed without any technical knowledge of the system.

Operators must also be able to generate graphs overviewing all patients inside the CCU, with a bonus ability of exporting when needed. This will assist the medical staff by allowing for instant bulk analysis, gaining an overall understanding of the patients in the ward’s care. This can help to plan what the remaining of the day might look like for the medical staff, expediating the process of referring for diagnosis by allowing for task prioritisation.

One of the most fundamental requirements is that the application must run identically on different operating systems. This mitigates the need for multiple training sessions and promotes uniformity. Additionally, it is understood that the budget may shift in the future: forcing the CCU to adopt different operating systems. Ensuing the program can run on all systems allows the CCU this freedom, meaning their budget will not limit the success of the application.

## Non-Functional

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Description | Source | Priority | Status | Assigned To | Test Cases |
| NF-1 | Efficient Design | Highsmith and Fowler (2001) | High | Complete | Callum |  |
| NF-2 | Shared Codebase | Specification-implied | Medium | Complete | Callum |  |
| NF-3 | End-User Testing and collaboration | Dayton and Barnum (2009) & Highsmith and Fowler (2001) | High | Complete | Callum |  |
| NF-4 | Highly Intuitive | Dayton and Barnum (2009) | Med | Complete | Callum |  |
| NF-5 | Security | Data Protection Act (2018) | High | In Progress | Callum |  |
| NF-6 | Fast Analysis | Specification | Medium | Complete | Callum |  |
| NF-7 | Feedback on Loading | Recommendation | Medium | Complete | Callum |  |
| NF-8 | Accessibility Options | Recommendation | Medium | Complete | Callum |  |

Producing an efficient design capable of running on minimal hardware and any operating system, using the same codebase, is paramount in allowing the CCU’s budget to stay as flexible as possible. Additionally, reducing three individual projects to one shared project helps to minimise the amount of maintenance a project will require post release: reducing financial strain on the CCU. If a bug is discovered, it only needs to be fixed in one place: not multiple.

Not every staff member of the CCU will be technologically advanced. Given this, the solution’s design must be intuitive and minimalistic. Simplicity is a key aspect associated with a successful agile delivery, as stated by the Manifesto (Highsmith and Fowler 2001), so this aligns perfectly with the goals of the project.

With the confidential nature of medical documents, the completed application must have some form of security in place. A simple option would be a pass key that has been encrypted. This should be able to be changed, but never be able to be decrypted. This is designed to protect the workstations from potential security threats. Another option is to setup a custom account for each practitioner in the ward, but this introduces complications with people forgetting passwords. Security protocols should also be introduced, forcing the change of a passcode every week.

Different viewing options must be made available to the users of this application, ranging from contrasting colours to resizable text. This is in the effort of assisting those with impaired vision, making the application more widely accessible to workers in the CCU.

Analysis of patients should not take longer than 5 minutes. Time is critical in the CCU, and the less time spent waiting for a computer to analyse the better. Longer analysis times can also create frustration in the staff, leading to a negatively impacted workplace culture. This toxic workplace culture would directly negate an aim of the project: making the CCU flow far smoother and lessen stress on the staff.

Given the potential for a delay whilst waiting for analysis to complete, there is a high likelihood that operators may become frustrated or think that the application has stopped working. It is important that they are made aware that the analysis is taking place, and they should expect delays. This can help to raise the overall effectiveness of the application on reducing the congestion faced in the CCU.

The model must have a high recall score, the metric used to calculate the correctness of the model’s predictions. If the recall score is low, this means more positive predictions were incorrect. This is valuable information, as keeping referrals decongested is a key element in the project’s goals. With a higher recall score meaning lower congestion, a good aim for this is a recall ability of 70%. End results of the model should mean no less than 70% of referrals made are valid patients in need of seeing a specialist dietitian.

# Project Plan

A screenshot of a computer

Description automatically generated

Arguably one of the most critical stages, following the rules of the Agile Manifesto (Highsmith and Fowler 2001), is the design phase. This allows developers the opportunity to gain feedback from the end-users, affording the “fail fast and often” approach. Collaboration of this nature helps to reduce the bias towards a developer’s own creation, also showing how User Centric Design can vastly improve the final project outcome. The approach followed in this project is a round of designs, followed by a round of feedback. This feedback is then taken forwards and used in the creation of the final product.

Backend design focus is also highlighted in its importance when working in a group dynamic. If someone were to work on a project by themselves, this wouldn’t be so drastic. Creating these designs helps to ensure everyone is on the same page about what needs to be done to create a finished product. Designs, like Use Case diagram, are also highly useful for showing clients before development to ensure nothing has been missed. These are soft knowledge diagrams used to convey how a user will interact with a system on the most basic level.

The middle ground for the project is setting up the programming environment. Services like GitHub offer developers ways of quickly sharing their code without blocking each other’s development. This is excellent and aligns perfectly with the methodology chosen as it serves the task of simplifying development: allowing the development team to instead focus on the code, not the collaboration. An extended part of setting up this environment is ensuring that the project will run on multiple systems, as this is a crucial part of the core requirements. A small test should be preformed to confirm the validity of the environment on different platforms.

Once the environment is setup, the development team can start their work on the front and back-end implementation simultaneously: ensuring the flow of the critical path. It’s important to note that the critical path must be adjusted if situations, unforeseen by the Risk Assessment, occur.

After the development cycle has completed, the team is able to conduct tests to ensure viability of a finished product and that it meets the agreed upon requirements. Once the tests are concluded, the project is considered finished and moves onto the Maintenance (Operations) phase. Given the multiplatform framework, allowing for a singular codebase, the maintenance for the finalised Dietitian will be minimal: helping to ensure costs associated with the application stay low. This can be considerably useful for areas like the public healthcare sector which are critically underfunded.

### Risk Register

The Risk Register is a formal way of declaring what could probabilistically go wrong within the bounds of the project. It helps developers and contractors alike know what they are signing on for so that nobody feels like they have been given the short end. It also provides an excellent resource for teams to strategise, leaning into ideas which take proactive measure to ensure the risks do not arise.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Risk ID | Description | Cause | Effect | Probability | Impact | Risk | Response |
| 1 | Not enough staff to maintain critical path | Walk outs or Illnesses | This can cause an increased workload on the remaining developers, possibly resulting in a diminished final product due to stress to maintain the critical path | Moderate | Major | High | Ensure enough “slip” time is allocated to the end of the project. This is designed to mitigate some effect of staff members being ill and allow time for compensation without requiring extra budgets. |
| 2 | Misunderstanding of the requirements | Designs were not appropriately tested | This can cause entire resets in the project to an origin point which satisfies the client | Unlikely | Severe | High | Due to the severity of this risk, a fail fast and early approach must be employed: ironing out and misunderstandings. |
| 3 | Predictions made by the model are not sufficiently accurate enough | Hyper-parameters were not finely tuned enough | This would be a moderately time-consuming venture to fix as fine-tuning parameters can take an unknown amount of time depending on the complexity of the problem. | Likely | Minor | High | Hyperparameter tuning algorithms must be employed to ensure the best parameters are found. This mitigates the concern but does not eliminate it. Machine Learning algorithms have lots of moving parts, so claiming the “best parameters” is foolish. Better considerations would be that the “best found parameters” are used |
| 4 | Loss of project work | While code is remotely backed up, it is entirely possible that access may be taken away | This could have drastic impacts as not every developer will have the same version of work stored on their machine.  Additionally, the developers may lose access entirely in instances where code development is happening on a remote server.  Overall, this would be catastrophic and could result in entire developmental setbacks. | Rare | Severe | Medium | The main branch of the codebase must be frequently backed up in a reliable method that someone has physical access to. This, in the worst case imaginable, still enables the developer to recover a hard drive from a faulty computer and extract the last save point achieved. |
| 5 | Legal and Compliance implication | Given technologies ever expanding domain, the law is constantly evolving with it. it’s entirely conceivable that regulations may change throughout the course of development, potentially adding new Requirements to the project | Any time a change in requirements occurs, the project must be reset to a standard point to allow for redesigns | Rare | Severe | Med | While there is nothing directly pre-emptive that can be done to prevent this from occurring, the project management team can still follow trends in the news in order to stay informed about upcoming changes to legislation. |

1. Within Agile management, the critical path relates to the shortest time taken from start to finish: aggregating different sections which can be completed synchronously. Given the small development team size, it is entirely possible that someone may fall ill and cause slight delays to the development progress as a result. This risk of time slip is greatly increased when factoring in the volatile position the development team are currently in, facing other deadlines which may take priority. This can have a dramatic consequence for the completion time of the project and has the potential to result in an extended budget to help cover the extra expenses. Additionally, this is a time sensitive project but not only because of the deadline. The more time taken the develop the application, the more patients are potentially discharged from the CCU without seeing a dietitian. To ensure this risk does not occur, a buffer must be assigned to the end of the project’s lifecycle to account for any potential slips.
2. As shown by many referenced Agile development projects, the Requirements phase is the most important. It is said that any misgiving later in a products development will likely stem from a misunderstanding of the requirements. This is a high-risk statement as requirements are constantly changing withing a projects lifecycle, but precautions can be taken to prevent overextended losses. The fail fast and often approach gives developers the freedom to collaborate with End-Users in the design phase of development. If these designs are not satisfactory, or something is missing from the asked specification, the development team can make additions or tweaks without impacting the critical path too heavily.
3. Given the machine learning nature of the project, it is entirely possible that predictions made will not hold any validity. This is highly dangerous as incorrect predictions can lead to required patients not getting the help they need, conversely it could also result in a mass submission of patients for consultation: committing to the issue of a congested healthcare system. Using the CSV training data provided, the model must be vigorously tested to ensure predictions made by the model have a high recall and precision. The two metrics, in combination, allow the development team to see the ratio of true positives correctly predicted out of all predictions made. This number must remain at a high value to distil confidence in the algorithm. Machine learning algorithms are highly complex, slight parameter changes can result in large differences in performance. As a result of this, different parameters must be extensively tested to ensure the best parameters are found for this dataset. If these parameters aren’t tuned correctly, and the recall and precision is negatively impacted as a result, it can result in the project taking extra time to complete whilst accuracy standards are being met.
4. While the probability is very low, it is foolish not to reference the possibility of developers access to the codebase being taken away. This could happen from a multitude of reasons, ranging from GitHub authentication services being offline to code bases being removed through negligence or malicious intent. With this, a copy of the “Master” branch for the project must be regularly backed up, ensuring a constant workflow is possible in the worst of cases. This data must be backed up on a machine that developers have physical access to, as it enables them to still recover files from drives if the computer’s other components were to break.
5. One of the biggest unforeseen risks in application development is the changing of public legislation. This can impact how data is stored or regulatory requirements about how Artificial Intelligence algorithms are trained. These types of risks are unavoidable and must be adapted to post fact. There are however precautions which can be taken. The project management team must consistently be staying up to date with trends in technology and be aware of any potential Bills coming into place which may impact development. Risks of severe natures can easily be detrimental to a software’s development lifecycle and can often lead to projects taking vastly longer to complete as new research may need to be done, or new staff training could take place. Additionally, if an act blind-sides the development team and greatly impacts the applications release, this can quickly lead to an entire reset to the project’s requirement and design phase.

## Design and Test Plans

### Front End Designs

The front-end designs were some of the most important in User Centric Development within the agile methodology (Dayton and Barnum 2009). With this, high levels of end user collaboration were needed. Initial designs were created and then presented to End-Users who provided feedback on areas they thought required improvement. While this didn’t align directly with the principles laid out in the Agile manifesto, the chosen form of end-user testing employed was Informal. While this does take away some validity of the tests, it provides development teams a quicker on-demand access to user-tests and can variably gain more technical insight into what is missing from the product. The three classes used for defining the end users were Non-Technical, Semi-Technical and Advanced-Technical. This aimed to bring feedback on designs from those varying degrees of knowledge, replicating what may be seen within the CCU.

### Design Phase

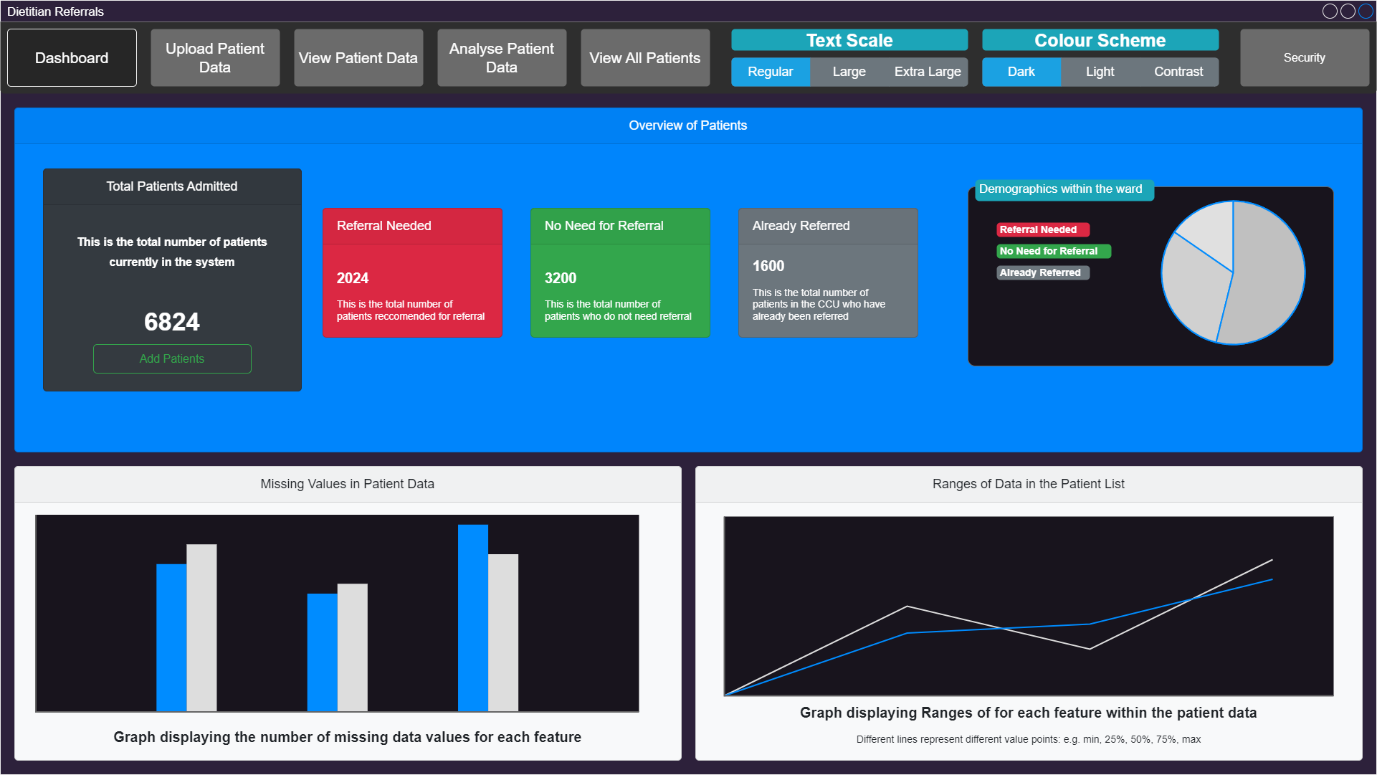
Given the user centric design model, it was paramount that designs be completed with agile in mind: namely that the product must be minimalistic and prefer effective use over heavy documentation. The designs created were all served with the question “is this needed?”. When the answer came up no, it was removed from the design. This aimed to reduce the clutter of the application and allow intuitive usage. Additionally, given the potential for different access requirements, healthcare operators must have options to increase their usage of the application: for example, different colour themes which may present data in a better way for them. The end goal of this User Centric design is to produce an application which fits the users, not a userbase which fits the application.

### Logging inA screenshot of a computer Description automatically generated

As mentioned by the requirements, the application must incorporate some form of security to protect the confidentiality of the information stored about the patients in the CCU. Given the type of application and its uses, being on an already secure computer on a secure network, minimal work needs to be done here: and so only a passkey was chosen for entry. This allows the hospital staff to share one code, ensuring that nobody is accidently locked out and aversely slowing the system down. There are however two login portals. If an admin password is used, it signs into an admin panel restricting uses of the application but allowing for password resets. If a regular password is used, it proceeds to the application’s dashboard.

When asked for feedback, these were the responses:

|  |
| --- |
| The colour scheme is pretty horrible (no offence). Lime green with blue, burgundy and cream????? I think a black border to the centre box would help and just choose a better overall scheme |
| The login button is thicker than the text field which I would change. |
| Change the green to blue and the blue into grey |
| yes i would maybe abit more modern it looks like you used css from old times |

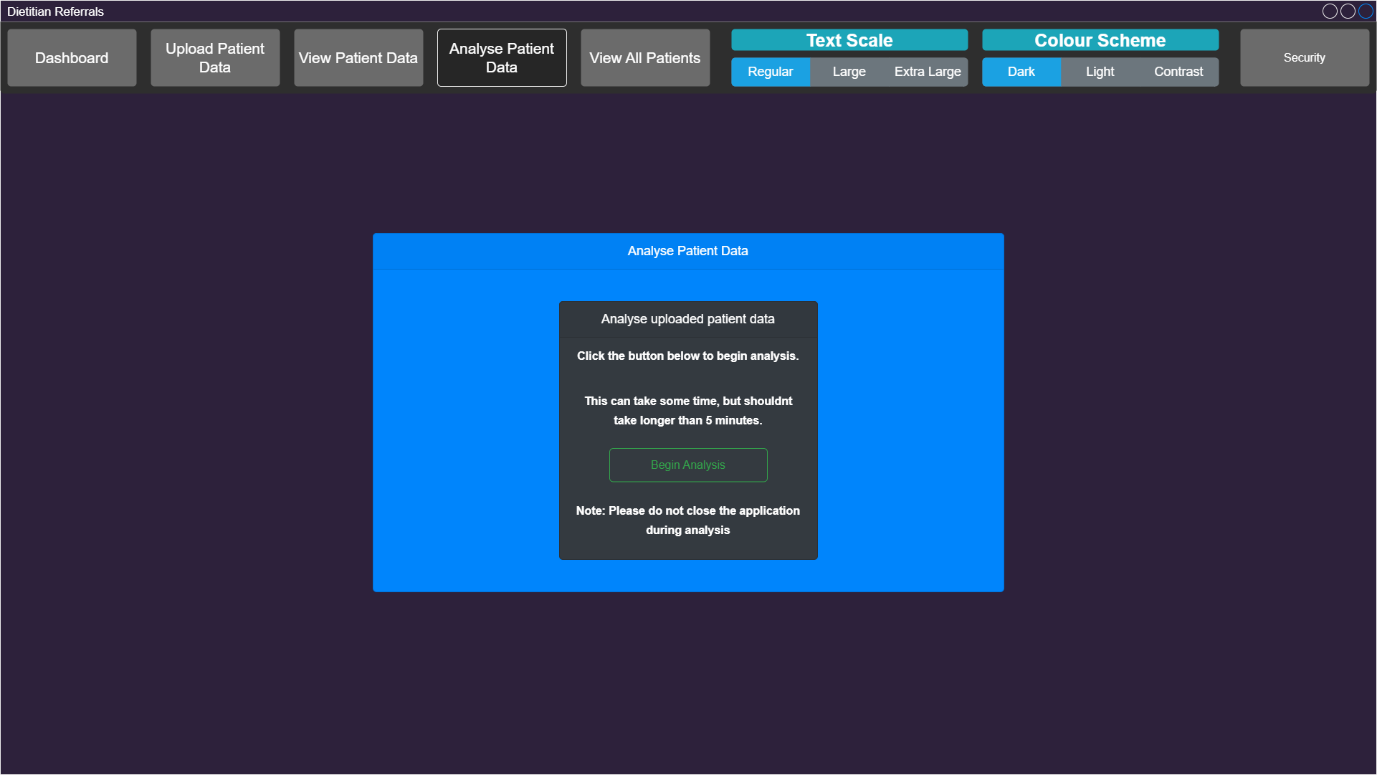
Dashboard

The dashboard is a critical part of this project. It allows the operators to get an overview of how the patients in the CCU should have time delegated. There is lots of information which needs to be conveyed here, but remaining minimalistic is a must to ensure cooperation with User Centric Designs. Graphs must be available for viewing, as some people respond much more to visual data and not just spreadsheet data.

When asked for feedback these were the responses:

|  |
| --- |
| Its a bit of an overload of information, theres too much to look at. Can you reorganise them into buttons that lead to other tabs or options. Colour scheme is too contrasting, maybe make it more muted |
| The blue background is a little bit out of place but other than that everything is clear and looks easy to follow. |
| no this one is good |

Patient Analysis

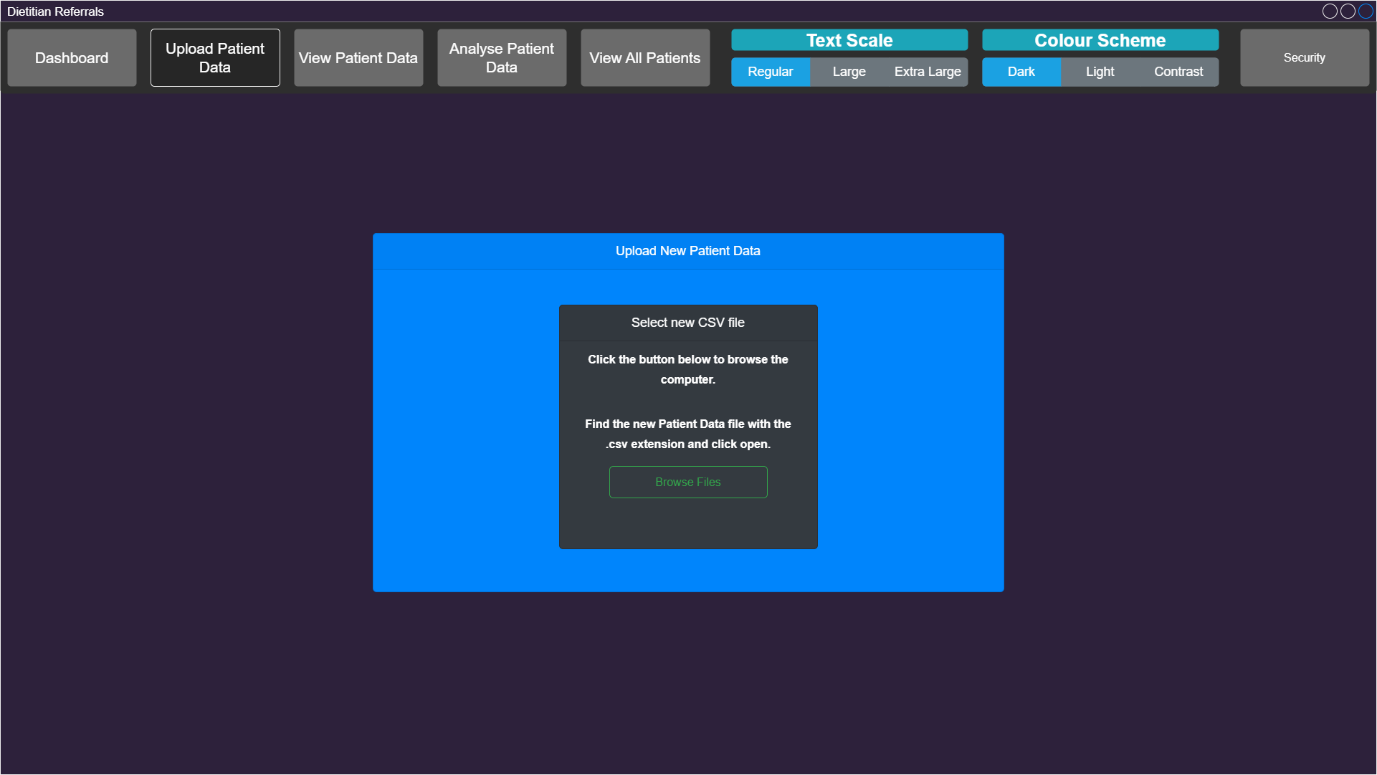


This page is minimal and acts only as a back-end interface. It interacts with the python script which analyses the uploaded patient CSV file and outputs predictions to a JSON file format for easy GET request. This process can take a varying amount of time, so a message must be shown to the operator to inform them that analysis is indeed taking place. Without this message, operators could become frustrated or disbelieve that analysis is taking place: causing them to issue multiple analysis requests and slow the system down.

When asked for feedback these were the responses:

|  |
| --- |
| I don't think the burgundy/purple goes very well with the blue but other than that its good |
| No need to change anything, its clear what this page does and the short messages are nice. |
| no looks good |

### Patient Upload

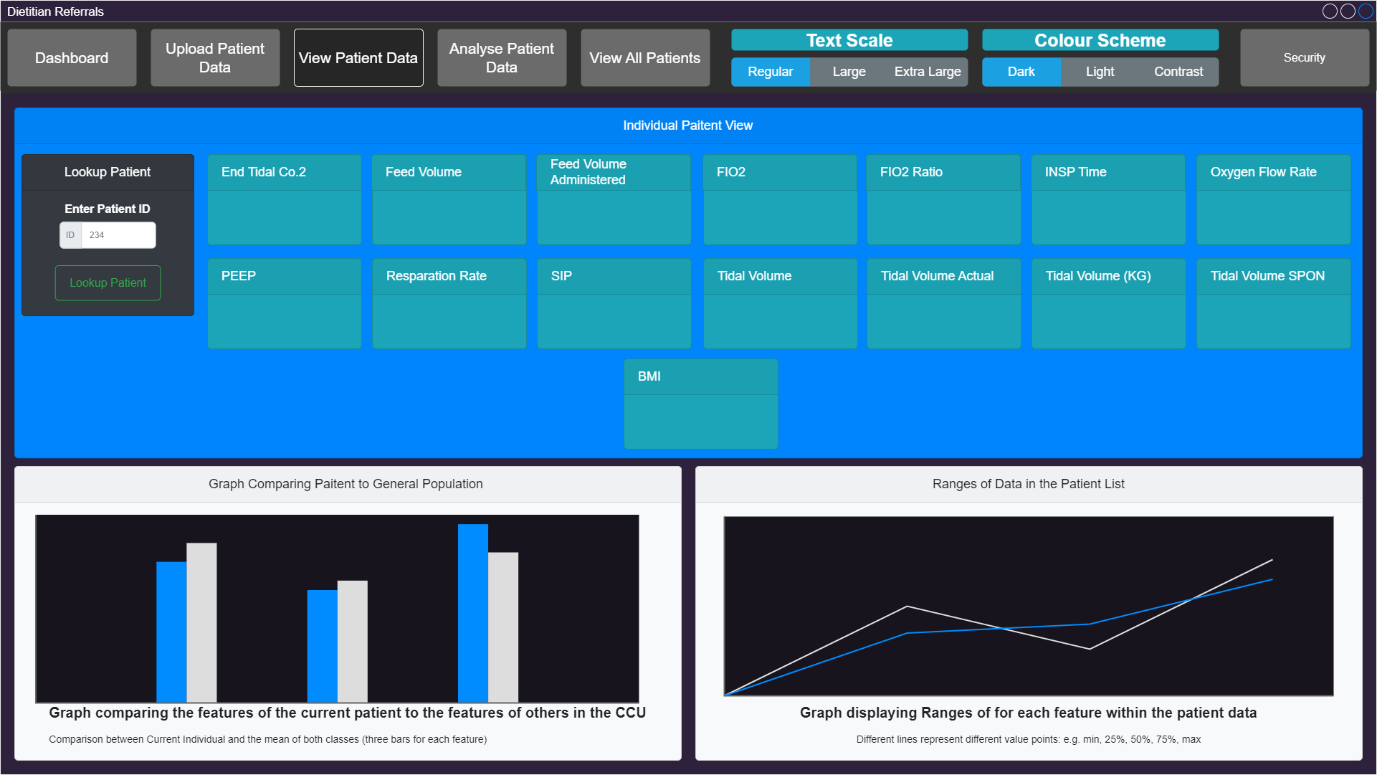


This page is also minimalistic but provides users a way of easily uploading their CSV files to the application for processing and analysis. Alerts should be given to the user, and client limitations should be imposed to restrict the types of files submitted for processing. These should only be limited to “.CSV” files.

When asked for feedback these were the responses:

|  |
| --- |
| I don't think the burgundy/purple goes very well with the blue but other than that its good |
| Very similar to the last page however, seeing this adding an icon to the button would be good. |
| Looks good |

### Patient Lookup (none-selected)

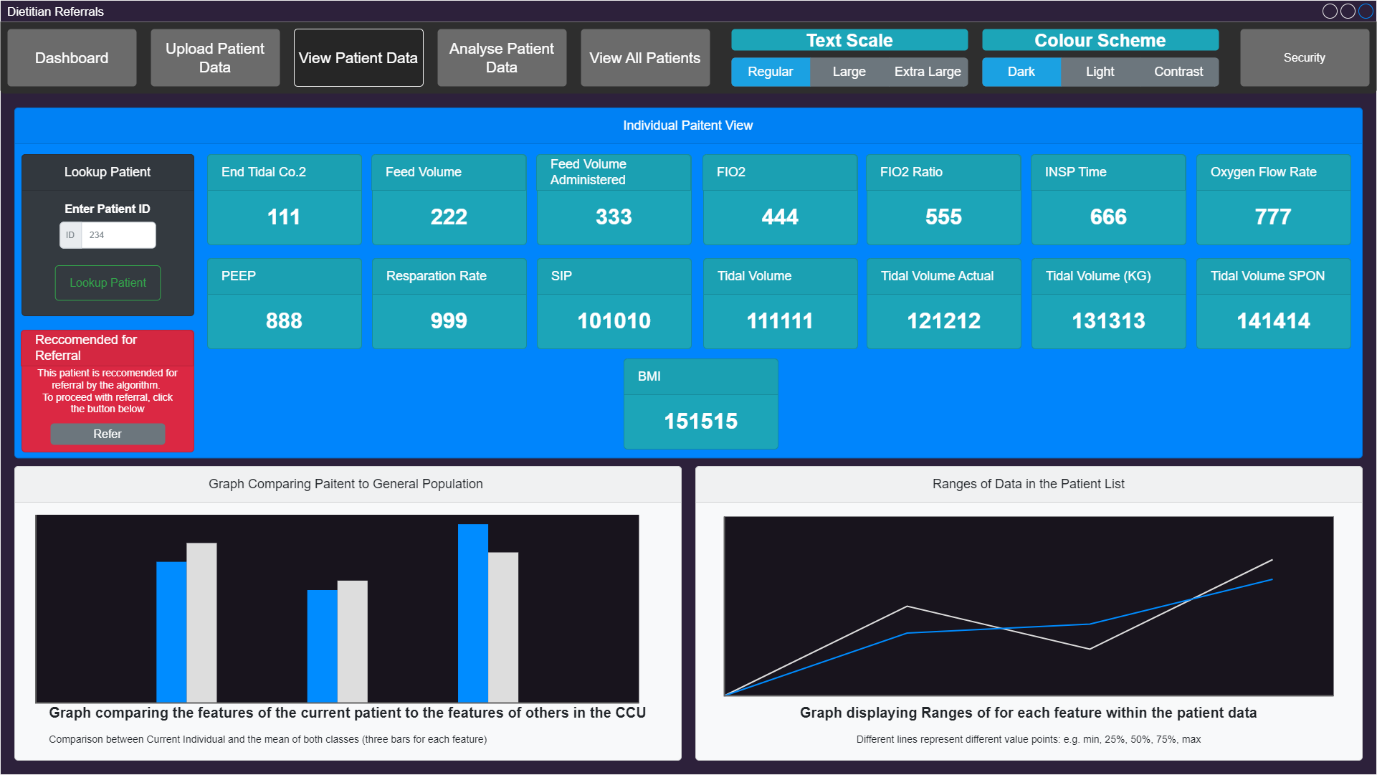


This section allows healthcare professionals to easily reference any patient in the CSV file. There is lots of information to convey, so ensuring the neat presentation is paramount for clear readability. There should not only be information relating to the searched patient, but also how that patient compares to others in the ward. This would allow professionals to directly see the comparison within the population and make more informed decisions about where their time would be best spent.

When asked for feedback these were the responses:

|  |
| --- |
| Turquoise on blue with white text can be straining on eyes |
| The BMI tab being in its own row looks quite bad, i would make all the squares slightly smaller so they can all fit into 2 nice rows |
| The torquoise on blue clashes and may make it harder for those with colour blindness to identify the boxes. |
| no looks good |
| no looks good |

### Patient Lookup (Referral Recommended)

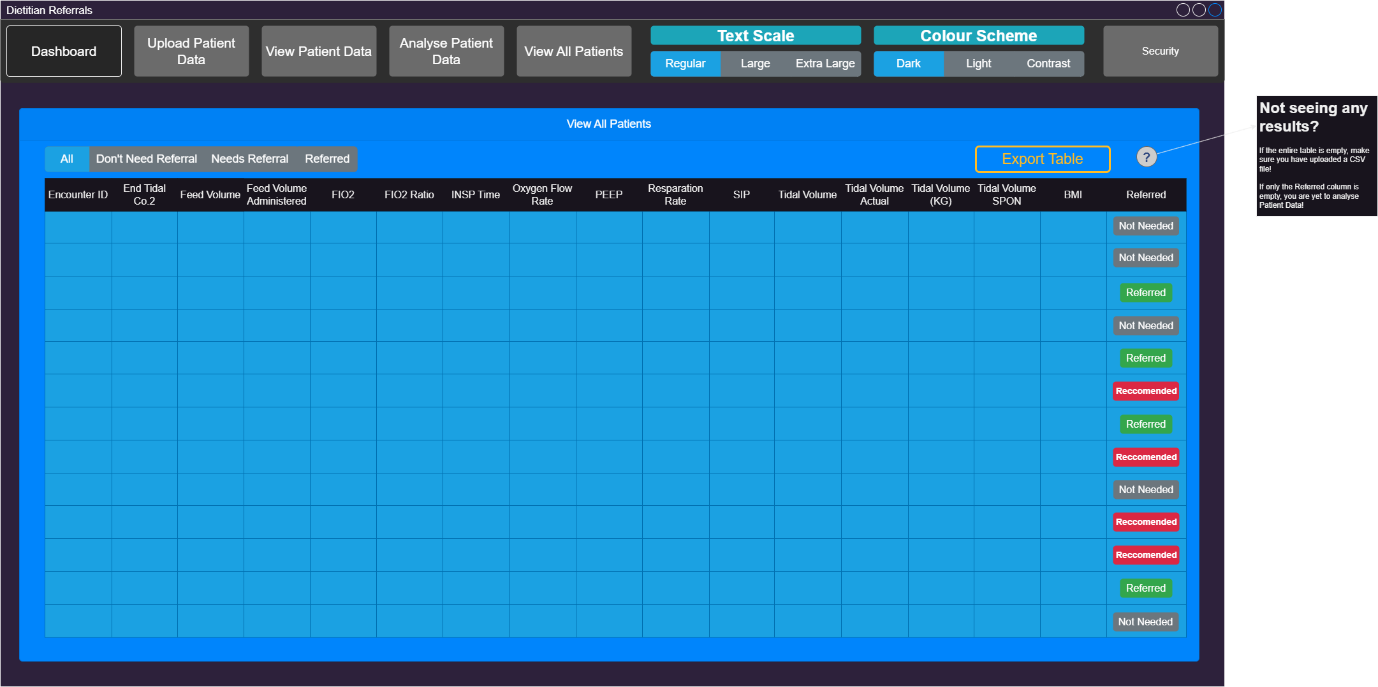


Similar to the previous window, this simply shows the change when a patient is selected by the lookup. The “Recommended for Referral” box is dynamic and changes colour and message depending on the model’s recommendations. Regardless of the recommendations, the operator should be able to manually refer a patient to consultations: overriding the potentially wrong decision made by the algorithm.

When asked for feedback these were the responses:

|  |
| --- |
| The BMI tab being in its own row looks quite bad, i would make all the squares slightly smaller so they can all fit into 2 nice rows |
| No. The message in red stands out and it's very clear with what it says. |
| No it looks good. |
| No it looks good. |

### View All Patients

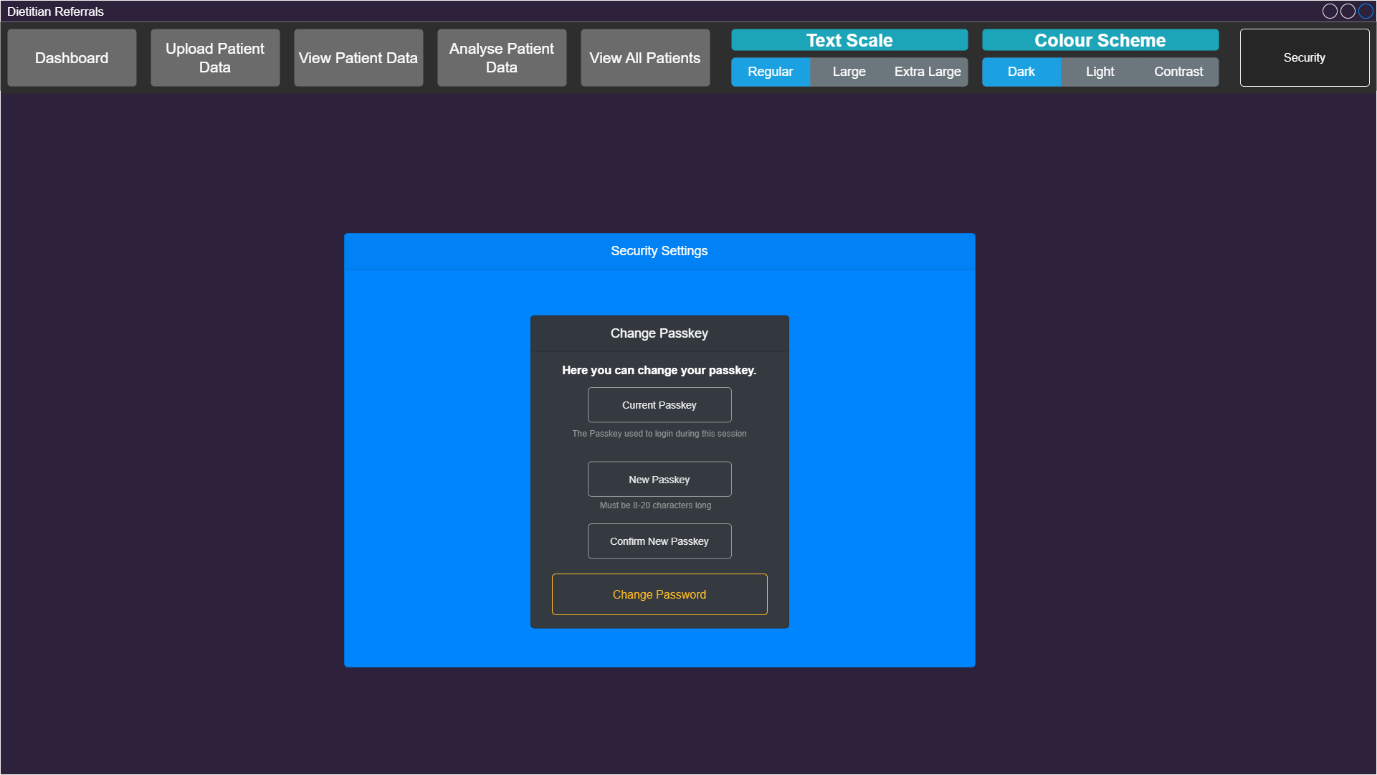


This page is used to show the operator every patient currently admitted to the CCU. It should also provide instant feedback from the analysis to display if the patient should be recommended for referral or not. Similarly to the individual lookup, the operator should be able to easily refer a patient who has been deemed not recommended and override the decision of the machine learning model.

When asked for feedback these were the responses:

|  |
| --- |
| Change the light blue chart to a lighter colour that contrasts better with the rest of the page |
| Similar to a page earlier, thge blue on blue may be hard for some people but other than that the table looks easy to follow. |
| Maybe change the blue background. |
| No it looks good. |

### Security Panel



This is another simple window designed with minimalism. It provides a box to enter the current password, ensuring that the user attempting to make the change has the permission to do so. This negates the possibility of a patient in the CCU, or a visitor, being able to easily change the login code: locking the healthcare professionals out of the application. Small hints are provided to the users to indicate what type of data should be inserted, helping to guide the less technologically advanced.

When asked for feedback these were the responses:

|  |
| --- |
| Further option incase you don't know the current passkey? |
| No its pretty good actually just don't like the burgundy |
| Spacing between buttons and button sizes don't look equal. |
| Standardised word "Passkey" OR "Password", instead of "Passkey" and "Password" |
| nop |

Finally, some multiple-choice questions were asked to determine the intuitive nature of the designs and ensure that an appropriate level of User Centric Design has been accomplished. Results of 1 were taken as a negative agreement and 10 seen as a positive agreement.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Could you change your password if needed?** | **Could you upload patient data if needed** | **Could you Analyse patient data if needed** | **Rate the overall simplicity of the design** | **Rate the how intuitive the design is** |
| 10 | 10 | 10 | 8 | 8 |
| 8 | 9 | 7 | 8 | 6 |
| 10 | 10 | 10 | 9 | 9 |
| 10 | 10 | 9 | 9 | 9 |
| 10 | 10 | 10 | 10 | 10 |

## Considerations Following Feedback

While the design seemed strong with internal testing, opening to informal End-User testing presented some vulnerabilities in the design. One of the biggest flaws identified was the colour scheme. This may be largely attributed to the front-end developer having colour blindness, resulting in colours mismatching. To accommodate this flaw in development, more heavy consideration must be given to the consistency of colour schemes as well as the colour balance. Ensing these colours are not too aggressive is paramount in delivering an appeasing project which provides easy access to the information provided. Additionally, styles must be standardised to approach a more uniform design. This can help users to navigate on first time uses because pages will seem to connect with each other more.

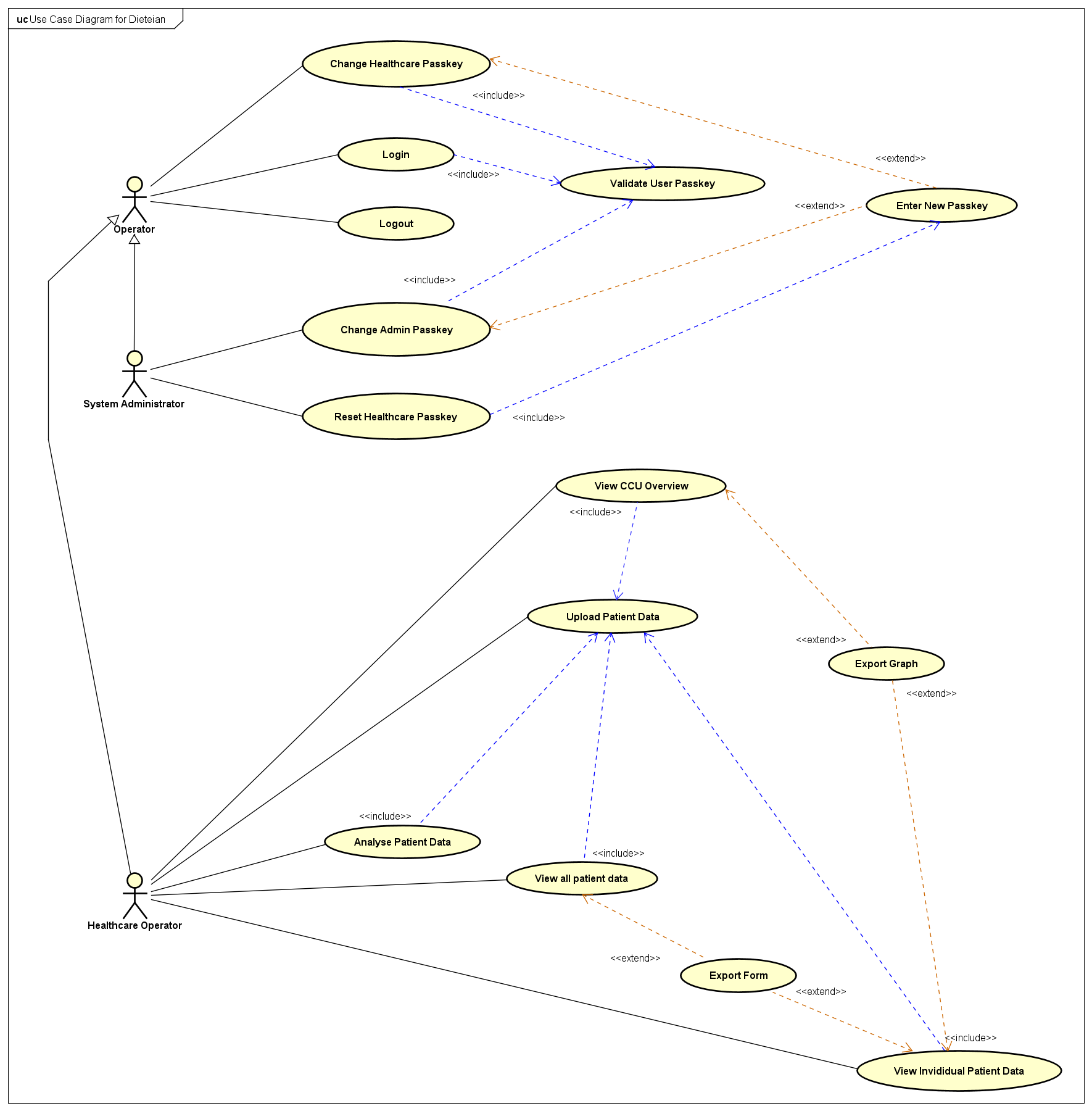
These end-user tests did show that the project’s focus on User Design has paid off tremendously, with a final simplicity rating of 8.8/10. This can immediately show the nature of the designs success in appealing to a non-technologically advanced user and indicates a successful design phase. Interestingly, the designs only scored 8.4/10 even though the average response for questions posed to indicate this value were scored at a 9.7/10. This shows that although the intuitively of the application is high, it may have been brought down by the poor colour scheme, miss-matched text, and inconsistent styling. These are things that will need to be addressed in the implementation of the project.

## Back-End Designs

As mentioned, the backend designs allow the developers to ensure they are constantly on the same page about what needs to happen to accomplish the goals set out in the project objectives and requirements. They can also be highly relevant in ensuring that the requirements have been correctly interpreted, as this will largely be reported in the technical documentation.

### Use Case

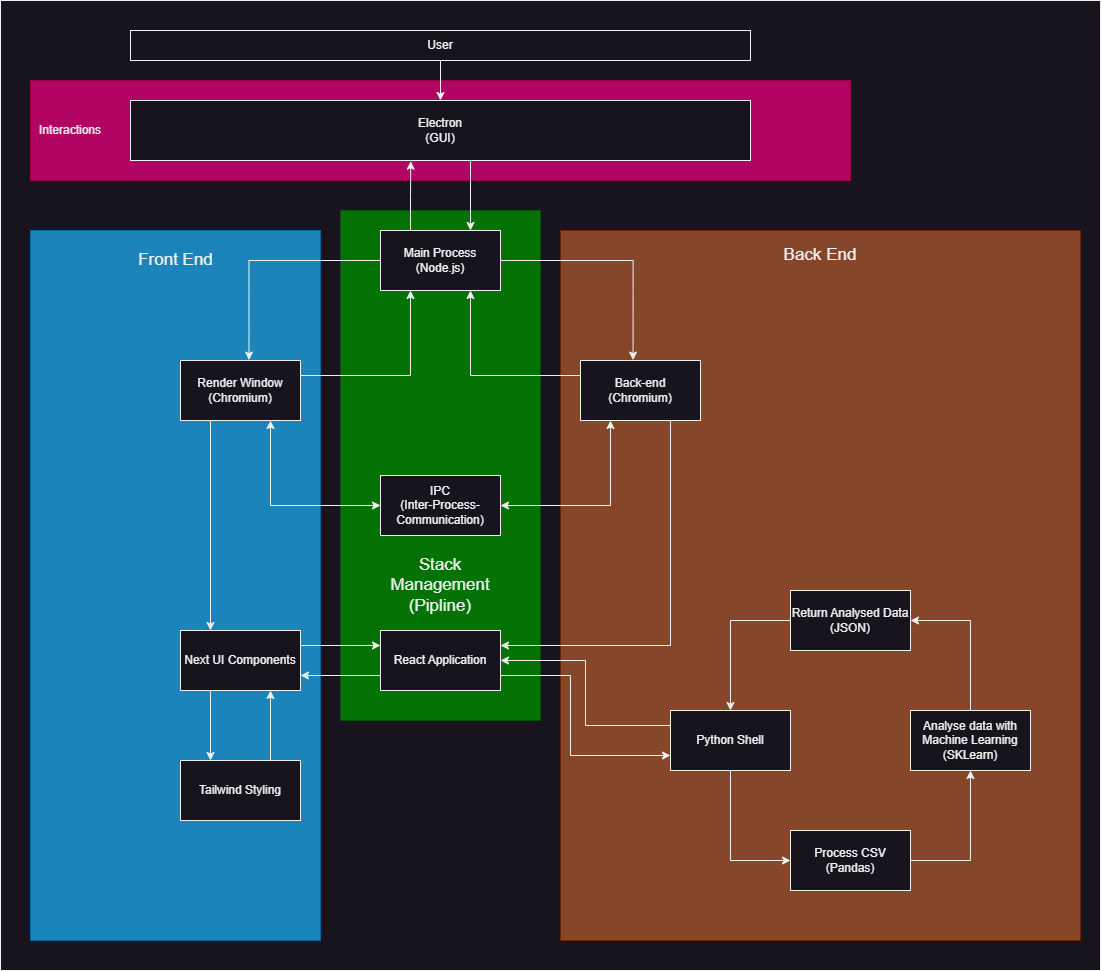
The Use Case diagram is a prime example of something which can be fundamental in showing a clear understanding of the requirements. It documents how users will interact with an application on the highest level, making it highly interpretable to those without technical knowledge. This form of design strategy is not entirely useful for the implementation of the project but can be a critical step in achieving a solid understanding of the task between the contractor and development team.



Once an agreement has been reached between the contractor and the development team, the design phase can move to focus on the integral workings of the applications back-end design.

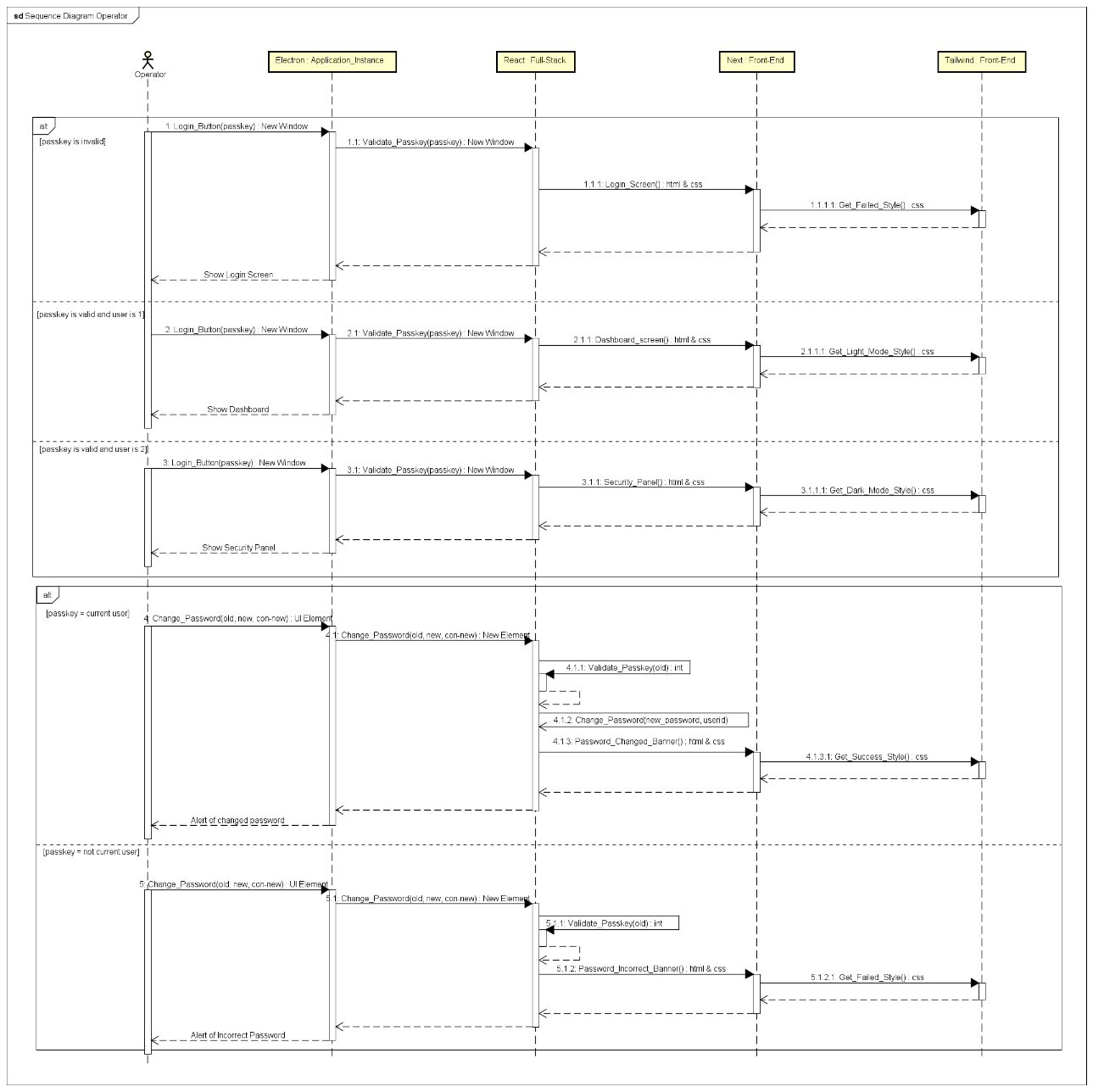
### High Level Architecture Diagram

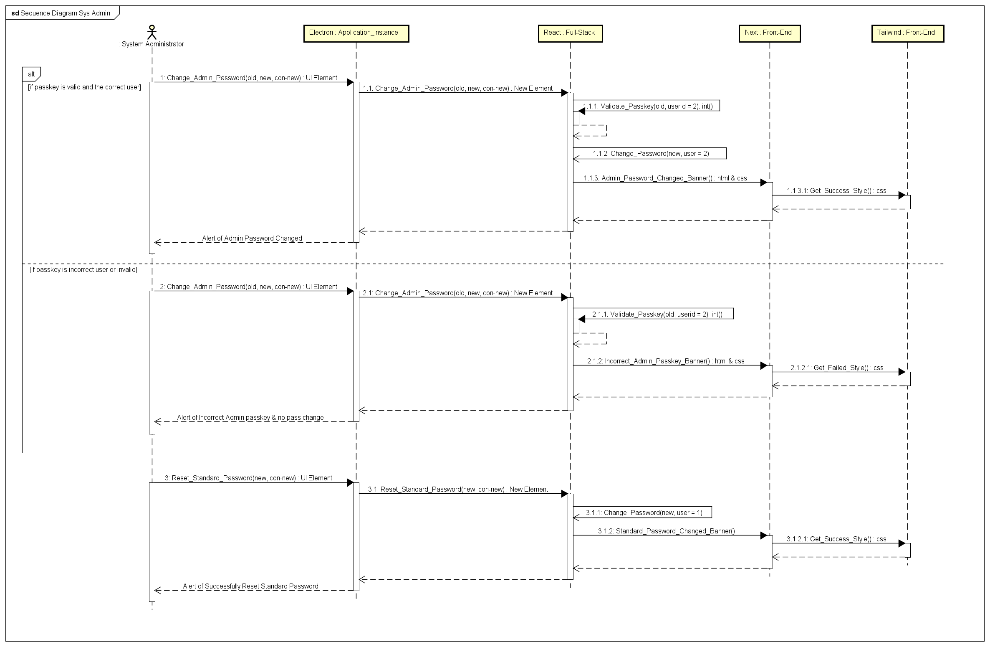
This phase of design can be invaluable to a team-based development project. It can be used as a direct list of ingredients that should be considered for the successful development of the project. It helps to build a team overall understanding of the working environment for the current project and can resolve any potential confusion in the back-end design specifications. It is also a key element in designing the proceeding diagram.



Once all developers have reached an understanding of the proposed design and any improvements have been made, the High-Level Architecture Sequence Diagram can be constructed.

### Sequential Diagrams

These diagrams are provided as the physical recipe for the project. It allows individuals to claim responsibility for individual tasks and ensure that when work is aggregated, it can be easily pieced together without much, if any, redesign. They are also invaluable in ensuring that any oversights have been found.



The main reason these forms of design can be invaluable to development teams, even consisting of a single developer: they allow for quick development of code as the logical flow has already been provided. Designing the logical flow can often be considered one of the most challenging operations in programming environments as constant changes in syntax are inevitable. Creating these diagrams allows for flexibility by not being overly specific, affording some creative freedom, but highlights the rough nature of how tasks should proceed.

This can also help to speed along stand-up meetings associated with agile development. This, combined with the Gantt chart, provides a comprehensive overview of the project’s status, and removes the option for developers to hide behind the “almost finished” façade presented in most forms of software development (Royce W.W.R. 1970).

### Test Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Type of Test | What is Tested | How is it Tested | Result of Test | Actions Taken |
| 1 | Erroneous | Login page | Bad Password is entered | Login is achieved regardless | No actions taken yet |
| 2 | Normal | Login page | Admin Password is entered | Regular login is achieved regardless | No actions taken yet |
| 3 | Normal | Login page | Operator Password is entered | Regular login is successful; | No actions needed |
| 4 | Erroneous | Dashboard statistic load | Loading page | Data is not loaded (figure1) as patients have not been uploaded and analysed | No actions taken, but the message indicating files haven’t been loaded should be much clearer |
| Normal | Dashboard statistic load | Loading page | Data is loaded (figure 2) as patients have been analysed and uploaded | No actions needed |
| 5 | Erroneous | File upload | A Good file is uploaded (.CSV) | File is accepted (figure 3) and analysed for base components (figure 3.5) | No actions needed |
| A bad file is uploaded ( Not .CSV) | File is rejected (figure 4) | No actions needed |
| 6 | Normal | Patient Analysis | Patient Analysis button click and animations. | Animations are successful in informing users of delays and analysis is successful (figure 5.1-5.3) | No actions needed |
| Animations are successful in informing users of delays and analysis is unsuccessful: but does not inform the user of this. Failure to analyse only occurs when a file has not been uploaded. (figure 5.1-5.3) | No actions taken yet, although there should be a fail response from the Post request indicating failure of execution. |
| 7 | Erroneous | Individual patient lookup | A bad patient id is entered for lookup | Lookup is unsuccessful but no message is shown. (figure 6) | No actions taken; this is an intended point of the application. When the input changes, requests are automatically made. If a message were to show for every wrong request, the screen would quickly flood. |
| 8 | Normal | Individual patient lookup | A good patient id is entered for lookup | A lookup is successful and patient information is returned (figure 7) | No actions taken |
| Lookup is unsuccessful as patients have not been uploaded (figure 6) | No actions taken but a message should indicate why the lookup is failing |
| 9 | Normal | View all patients | The page is loaded | Form loading is unsuccessful as patient data has not been uploaded (figure 8) | No actions taken but a message should indicate to upload patient data |
| Form loading is successful (figure 9) | No actions taken |
| 10 | Normal | View all patients | Filter patients by recommendation | There is no option to filter | Actions must be taken as this is part of the requirements |
| 11 | Normal | Run on different platforms | The application is run on Windows | Windows application runs successfully (Figure 10) | No actions needed |
| The application is run on MacOS | MacOS application runs successfully (figure 11) | No actions needed |
| The application is run on Linux | Linux operating system won’t install | N/A |
| 12 | Normal | Responsive Navigation bar | Tabs are changed | Tabs react accordingly moving from dashboard to upload  (figure 12 -> figure 13) | No actions needed |
| 13 | Normal | Accessibility | Dark mode is tested | Styles change to reflect a higher contrast style (figure 14) | No actions needed |
| Light mode is tested | Style change to reflect a more modern approach with more colour used to represent information (figure 15) | No actions needed |

A screenshot of a computer

Description automatically generated

Data is not loaded (Figure 1)

A screenshot of a computer

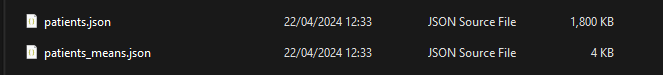
Description automatically generated

Data is loaded (figure 2)

A screenshot of a computer

Description automatically generated

File is accepted (figure 3)



Components are analysed (figure 3.5)

A screenshot of a computer

Description automatically generated

File is Not accepted (figure 3)

A screenshot of a computer

Description automatically generated

Document is ready for analysis (figure 5.1)

A screenshot of a computer

Description automatically generated

Document is analysing (figure 5.2)

A screenshot of a computer

Description automatically generated

analysis is successful (figure 5.3)

A screenshot of a computer

Description automatically generated

Lookup is unsuccessful (figure 6)

A screenshot of a medical analysis

Description automatically generated

A lookup is successful (figure 8)

A screenshot of a computer

Description automatically generated

Form loading is unsuccessful (figure 10)

A screenshot of a computer screen

Description automatically generated

Form loading is successful (figure 11)

A screenshot of a computer

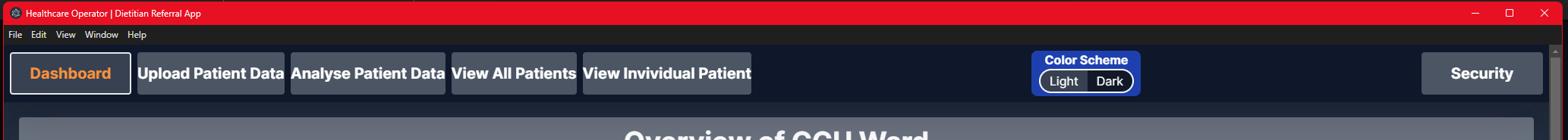
Description automatically generated

Windows application runs successfully (Figure 12)

A screen shot of a computer

Description automatically generated

MacOS application runs successfully (figure 11)



Dashboard tab (figure 12)



Upload Tab (hover seen affecting one tab option) (figure 13)

A screenshot of a computer

Description automatically generated

Dark mode styling (figure 14)

A screenshot of a computer

Description automatically generated

Light mode styling (figure 15)

# Individual Report: Callum Hornblower

Reflecting on one’s own work is essential in maintaining improvement. Given the group nature of the task, communications and interactions with group members must analysed. This project was designed to hold multiple participants, but due to unforeseen circumstances out of the risk-assessments scope, the project members were unavailable for contribution towards the end-result.

The organisation of the team leaves a lot to be desired, with multiple members failing to do anything, and others not being available for meetings. These meeting should have been scheduled at the start of the project and maintained at a steady pace regardless of the number of attendees. This forward passed focus group would allow for progress to be made in the downtime between sprints, not through vertical growth: but instead, linear. While linear growth may not be beneficial to the project’s timeline, it provides a stronger foundation on which to build the upcoming phases.

Admittedly, my own interaction with team meetings was not ideal. Due to circumstances out of my control in my personal life, my start was delayed and caused other projects to run alongside this: forcing my hand in choosing where to best delegate remaining time. This was felt when the project was in the implementation phase. When the bugs grouped up, it made the critical path slip by the exact amount of time spent on each feature. Usually, this wouldn’t have been so much of an issue. However, given the lack of contribution from other group members, slips quickly became landslides.

One of the main identified implied requirements of the project was the security features, allowing for a “locking” of the application when it wasn’t directly in use. This would have featured SHA256 encryption. With the amount of time slip pushing the critical path closer and closer to the deadline, decisions were made to strip the excess requirements and focus on the explicitly mentioned requirements. With that being said, this application in its current state does not hold up to the DPA standards (2018) with impunity. While this is the case, data stored is anonymised inside of this application, and means that there is no technical breach of data if someone were to gain unauthorised access. This is a gray spot, but the application should still be viable for release.

If other members of the group have contributed, this would have definitely been achievable but given the protracted timeline, it was decided to leave the project in a functional state which provides all the necessary features to be useful to the CCU. Leaving this project to be delayed could have been possible in a simulation where the end service use is not diagnosis. Delaying projects of this nature, however, could easily result in theoretical patients going without diagnosis.

Given the agile methodology provided with the project, it is entirely possible for these other features to be addressed in future iterations. Inside the codebase different frameworks exists for all the listed features which have not been added, there is just not enough time in the project’s scope to complete them all.

I believe that my biggest strength within this project has been organisation. Although a late start occurred, the project was still able to deliver with all the necessary functionality to assist in diagnosing patients within the CCU. If I had not been able to delegate time specifically to each task, the project would have surly overrun and resulted in a complete failure. Regardless of the grade achieved in the project itself: I am content with the amount of effort I have personally put into the development of the project to ensure a delivery on schedule.

I think my use of front-end and back-end processing allowed for excellent use of resources. Handling all rendering on the front end meant that development could move much quicker and reduce the overall number of errors occurring. This also allowed for user inputs to change the outcome of the page’s direction. One of my favourite implementations of this is the table for displaying all patients that have been analysed. It uses an array of TypeScriptReact elements to store a dynamic html page which can be rendered which much fewer lines of code. The result is a simple loop which iterates over the received patient data and builds a table with alternative colours for increased readability.

I believe this project could have been completed with far greater finesse, providing other group members involvement. Given the allotted time frame, complexity of the project, and the unavoidable circumstances: I believe it to be a success regardless, and I’m happy to have my name attached to it.

# References

Dayton, D.D. and Barnum, C.B. (2009) The Impact of Agile on User-centered Design: Two Surveys Tell the Story. Society For Technical Communication [online]. 56 (3), pp. 219-234. [Accessed 13 April 2024].

McQuade, M.R.M., Hunter, A.P.H. and Moore, S.M. (2019) Appendix A: How Agile Software Development Came to Enable Adaptable Systems (2019) Acquisition of Software–defined Hardware–based Adaptable Systems [online]. Washington, Dc: Center for Strategic and International Studies, pp. 31-36 [Accessed 13 April 2024].

Royce, W.W.R. (1970) Managing the Development of large Software Systems. ***Ninth International Conference on Software Engineering.***Los Angeles, March 1987. Washington D.C: IEEE Computer Society Press, pp. 328-338

Bell, T.E.B and Thayer, T.A.T. (1976) Software Requirements: Are they really a problem?. ***Second International Conference on Software Engineering.*** San Francisco, CA, October 13-15, 1976. Washington D.C: IEEE Computer Society Press, pp. 61-68

Highsmith, J.H and Fowler, M.F. (2001) ***Manifesto for Agile Software Development***[online], Utah, 11-13 February 2001. The Agile Alliance. Available from: https://agilemanifesto.org/ [Accessed 14 April 2024]

Highsmith, J.H (2001) History. ***The Agile Manifesto*** [blog]. 2001 Available from: https://agilemanifesto.org/history.html [Accessed 14 April 2024]

Electron: Showcase [online] (2024) Available From: https://www.electronjs.org/apps [Accessed 14th April 2024]

Hornblower, C.H. (2024) ***Machine Learning: A Comparison of Two Models*** [online].BSc, University of the West of England. Available from: https://github.com/Sp0xF8/Individual-Project/blob/main/22019799.ipynb [Accessed 14th April 2024]

Data Protection Act [online] (2018) Parliament: House of Commons. UK Parliament. Available from: https://www.legislation.gov.uk/ukpga/2018/12

Pimenidis, E.P., Soomro, K.S., Papaleonidas, A.P., and Psathas, A.P.P (2023) Can Machine Learning support Improvement in Effective Nutrition of Patients in Critical Care Units? In: Iliadis, L.I., Papaleonidas, A.P., Angelov, P.A., and Jayne, C.J. (2023) ***Artificial Neural Networks and Machine Learning*** [online] Switzerland: Springer Nature Switzerland. pp. 137-146. [Accessed 21tst April 2024]