faculty

AI-supported patient flow at Kettering General Hospital

Final Report: Technical approach and next steps

November 2021

Faculty Science Ltd, 54 Welbeck Street, London, W1G 9XS | info@faculty.ai
Company Registration No. 08873131 UK | VAT Registration No. GB 202 8674 18

Contents

Abbreviations

ACE	Accelerated Capability Environment
ADT	Admission, Discharge and Transfer
AI	Artificial Intelligence
DTA	Decision To Admit
ED	Emergency Department
GP	Gaussian Process
KGH	Kettering General Hospital NHS Foundation Trust
LoS	Length of Stay
MCTS	Monte Carlo Tree Search
MVP	Minimum Viable Product
ML	Machine Learning
NHS	National Health Service
OPEL	Operations Pressure Escalation Levels
PAS	Patient Administration System
PoC	Proof of Concept
RL	Reinforcement Learning
SMOC	Senior Manager On-Call
UCB	Tree Policy Score
UEC	Urgent and Emergency Care
UI	User Interface

Executive summary

Background and project overview

- The NHSX AI Skunkworks team and the Kettering General Hospital NHS Foundation Trust (KGH) commissioned a 12-week project via the Accelerated Capability Environment (ACE) to test whether Machine Learning (ML) can assist with the 'bed tetris' challenge.
- 'Bed tetris' are two words that have been used to describe the complex process that experienced bed managers are able to seamlessly manage in their heads on a daily basis. That is, the consideration of a sequence of bed moves that often need to be made as a result of one patient's admission into a hospital setting.
- COVID-19 has further complicated bed allocation decisions as there is now an additional set of hospital and patient movement constraints to consider.
- Faculty was engaged by ACE to develop a Proof of Concept (PoC) to demonstrate how artificial intelligence (AI) could be used to enable both site managers and less experienced staff members to allocate and optimise patient movement and safety within the hospital.
- Faculty's objective was to support site managers to achieve the right patient, in the right bed at the right time through a solution comprising four interlocking parts; a virtual hospital environment, a demand predictor, an allocation recommendation model and a user interface (UI).

Technical approach and outcomes

- Faculty carried out a Discovery Phase involving KGH staff interviews to establish the scope of the PoC and gather the KGH patient and hospital layout data. Data analysis, cleaning and engineering was then performed on five years of Patient Administration System (PAS) and two years of Patient Flow system data. Finally, in the third phase of this project we built and validated two ML models, a time series admissions forecasting model and a Monte Carlo Tree Search (MCTS) model.
- We built a highly-performant admissions forecasting model, leveraging a Bayesian modelling approach. Trained on five years of previous KGH admissions data, it helps inform the process of allocating a patient to a bed by providing information on the profile of patients that may be admitted in the future.
- MCTS and Greedy optimisation approaches were simultaneously implemented and validated against each other within a virtual hospital environment reflective of KGH infrastructure. Greedy was then chosen to integrate with the forecasting model and the UI due to its efficiency, tunability and interpretability in an operational setting.
- Results from our experimentation with MCTS suggest that there are various routes that could be explored in a research and development setting to further develop, test and validate this approach.

Next steps and overview

- Our discovery interviews with senior KGH staff members in the site and Senior Manager Oncall (SMOC) teams identified three key bed management challenges or other opportunities for where AI could be used to support hospital, ward and patient level capacity and demand decisions.

- Learnings that emerged conclude that the capabilities of this PoC need to be built upon to further support the identified bed management challenges faced by frontline hospital staff members. We identified a roadmap to turn the PoC into a Minimum Viable Product (MVP) addressing the described challenges.

- Finally, to align with KGH's ambition to become 'the most digital hospital in England', we outline some specific technical and operational issues under two overarching recommendations that would need to be solved in parallel with any future development project phases to support the delivery of an MVP.

01 Introduction

01.1 Background & Context

The COVID-19 pandemic has severely disrupted the National Health Service (NHS) and put incredible pressure on staff that were already struggling pre pandemic to meet many of its care targets. Patients have faced lengthy waits in the Emergency Departments (ED) for a bed , often due to a lack of available bed capacity in the hospital. Effective bed allocation or getting the right patient, in the right bed, at the right time has been proven to reduce Length of Stay (LoS), improve patient outcomes, safety and staff satisfaction.

Admitting and allocating a patient to a bed in an Acute Trust is both a challenging and nuanced process, often referred to as 'bed tetris' due to a number of hospital infrastructure and individual patient constraints that need to be considered or met. Experienced hospital site managers can consider a complex sequence of possible bed moves in their heads and weigh up possible avenues to overcome these capacity challenges but COVID-19 has added another layer of complexity. In addition, when highly experienced staff members are not present, less experienced members of the site team or SMOC have to make these tough decisions without an in-depth background knowledge.

ACE was engaged by the NHSX AI Skunkworks team and senior staff of the KGH and commissioned our expert data science company to build a PoC ML algorithm to assist with the 'bed tetris' challenge. This PoC was aimed at demonstrating capabilities that would enable both site managers and less experienced staff members to allocate and optimise patient movement and safety within the hospital.

01.2 Our Approach

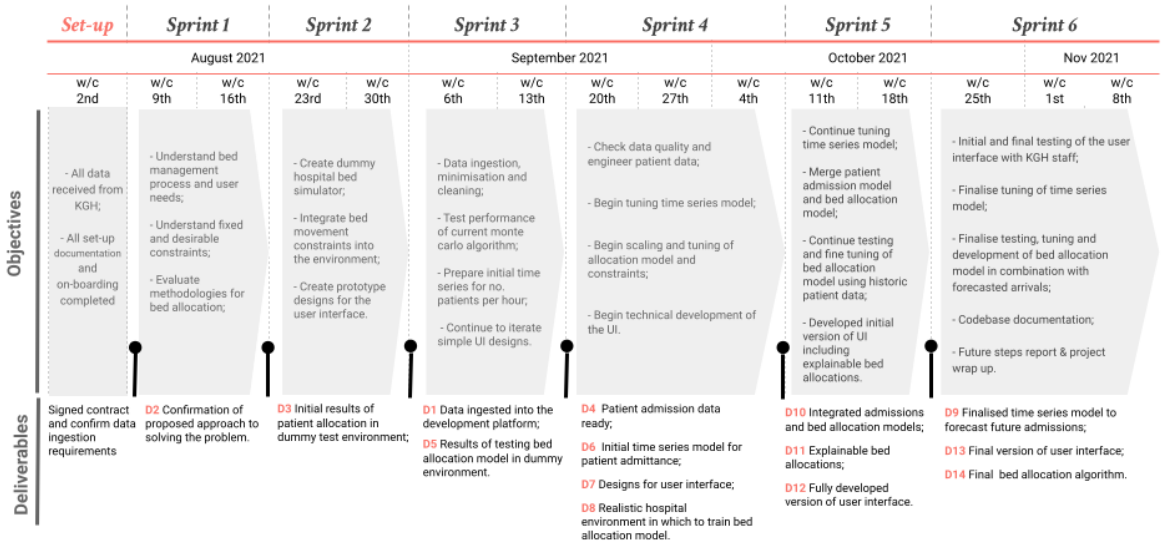
To address this issue, we proposed using innovative AI techniques to build a PoC that would both respect and enhance the expertise of the KGH site team, ensuring that they remain the 'human in the loop'. The overall product focus would ensure that the right patient is in the right bed, receiving the right care, at the right time.

There are four interlocking parts to our PoC bed allocation system:

1. A virtual hospital environment: Capturing information about each ward, bed and the patients allocated to beds.
2. A demand predictor: Forecasting new inpatient arrivals so as to inform bed allocation strategies over a 24-hour window. Aiming to avoid multiple moves as the bed state changes throughout the day.
3. An allocation recommendation model: Using information about the current environment and the predicted bed demand to recommend optimal patient moves that minimise the constraints broken.
4. A user interface: Displaying the suggested moves to KGH staff and why they were made.

01.2.1 Delivery Plan

This project was delivered over the course of 14 weeks made up of six sprints, as outlined by the plan below.



Please note that the project plan above contains the two-week extension of time requested by the Faculty due to delays in receiving the required data for modelling from KGH. This revised plan was agreed by all parties in writing on 4th November 2021.

01.4 Deliverables

The deliverables in this project are detailed below:

- The integrated codebase for the virtual hospital environment, demand series predictor or time series model and the bed allocation model;
- The fully developed version of a simple UI built for the purposes of demonstrating the capabilities of the developed bed allocation model and for user testing to fine tune model accuracy;
- The wireframes for a potential future product build phase, designed around the needs of users and their workflows following the initial user discovery;
- This report; summarising the PoC product that has been developed, the technical approach to each of the systems components, the results and finally a list of recommendations and possible future steps.

02 Overview POC bed allocation tool

02.1 PoC User Discovery

We conducted a number of discovery sessions and exercises across the first two sprints of this project to understand complex bed allocation processes. We interviewed six senior staff members of the KGH operational site team that manage the daily process of admitting and allocating a patient to a hospital bed, and managers that regularly act in the position of SMOC after hours.

02.2 PoC User Discovery Aims

The aims of this discovery phase were as follows:

- To understand more about the complex process of bed management at KGH and specifically the current steps and constraints involved when allocating a patient to a bed;
- To document user stories and define the scope of the initial PoC;
- To engineer the KGH patient and hospital layout data;
- To evaluate the technical AI methodologies that could be employed to support bed allocation.

Section 02.3 describes the scope of the overall PoC tool developed during this commission, the point in the carepath that it supports and the wards, type of patient and constraints included. In section 02.4 we give an overview of the functionalities of the PoC User Interface (UI).

In addition to achieving its original aims, the discovery phase identified key challenges and areas where data science and Machine Learning could be introduced to address pain points. Chapter 04 of this report sets out how KGH can use AI to meet these challenges and achieve the right patient, in the right bed at the right time.

02.3 KGH Bed Allocation PoC Scope

The steps in the KGH patient pathway and where allocation occurs can be illustrated as follows:

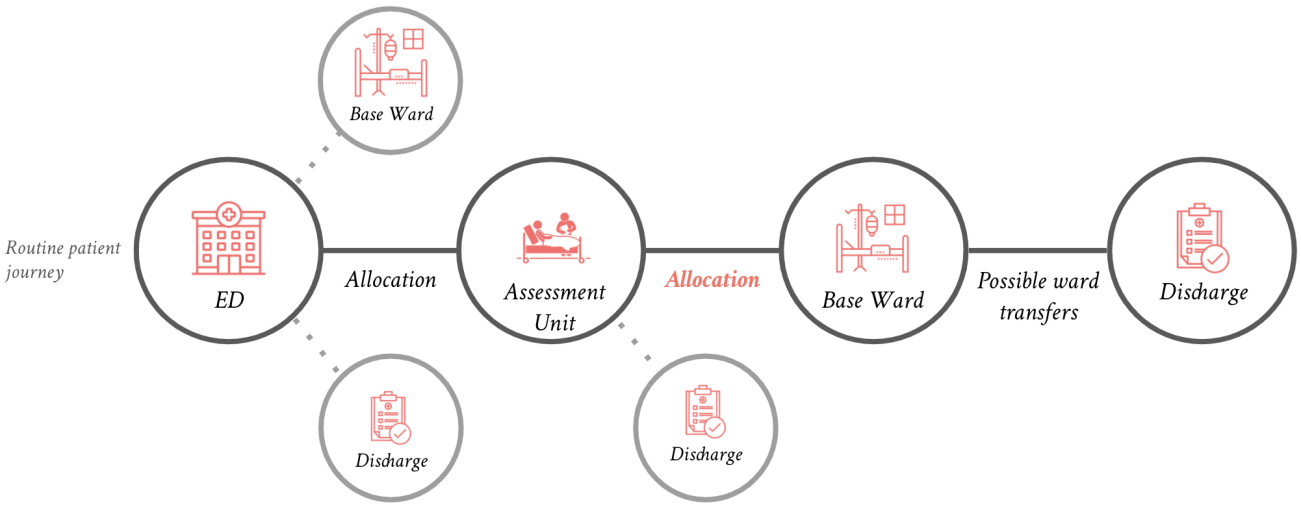


Figure 2.3: Typical flow of adult medicine and surgical patients through KGH showing points of possible bed allocation or discharge

These pathway steps are described in more detail in section 06.1 of the appendix.

02.3.1 Setting and type of allocation PoC supports

There are two main points between the steps in the journey where a patient can be allocated to a hospital bed.

Admitting and allocating a patient to a ward from an assessment unit was described by allocators as far more difficult compared with an allocation from the ED to an assessment unit. Reasons cited included the following:

- Complexities of having to consider multiple ADTs (Admissions, Discharges and Transfers) when allocating each patient;
- Lack of overall bed capacity and side rooms for patients with infections requiring isolation or immunosuppression;
- COVID-19 constraints adding another layer of complexity to an already long list of constraints;
- Delays in consultant reviewing patient;
- Late discharge decisions;
- Constraint of specific single sex wards and bed bays.

For this reason, we chose the PoC to initially focus on supporting the allocation of patients from an assessment unit to a base ward bed.

“Allocating out of ED I would say is the easy bit. Allocating out and planning ward beds ahead for patients coming out of the assessment units to medical wards is a juggling act

“We've got to factor COVID-19 into our decision making now...it makes your number of moves even more limited

“There is a priority order for side rooms but I think that this bothers me the most going into winter, we don’t have enough

“You can’t break the gender rule unless you are really up against it and that would have to be an executive decision

02.3.2 Scope of virtual PoC hospital environment and demand predictor

In order to assess the performance of the bed allocation algorithm, we needed to simulate the hospital environment in which patients are allocated. This simulated hospital environment needed to represent the layout of KGH and incorporate a number of constraints that reflect the characteristics of bed bays and wards. These characteristics are listed below.

Patient type: Adult patient with medicine or surgical specialty

Bed type: All general beds (the PoC does not differentiate between types of hospital beds)

Ward type: 10 adult medicine and four surgical wards

Demand type: The PoC demand predictor predicts the number of non-elective and elective patients likely to need admission from the emergency department every hour of the day, seven days per week. This demand is then factored into allocation recommendations by considering the future state of the hospital. The forecasts are further described in chapter 03.

Future project phases could focus on breaking attendance forecasts down even further or forecasting predicted LoS and number of expected discharges; both of which are recommended in chapter 04.

We noted during our discussions with users that adult medicine and surgical patients can be moved into continually changing escalation areas. Although the PoC will not enable the user to change the layout of the hospital, support during times of escalation was a clear theme that emerged from our user engagement and is described in chapter 04.

02.3.3 Scope PoC allocation model

An overview of the technical details of the allocation model is in chapter 03 of this report but the types of bed allocation and constraints supported by the PoC are described below.

Allocation type: Bed managers described three types of allocations:

1. **Admit to empty bed -** This type of allocation will be supported by the PoC;
2. **Admit pending discharge -** This type of allocation has been descoped for the purposes of the PoC;
3. **Transfer existing patient:** When assessing the suitability of a patient to transfer a number of metrics and individuals are involved in the decision. Therefore, this type of allocation was also descoped.

Finally, many allocation decisions of different types can be made simultaneously, affecting multiple patients. This is particularly true during times of escalation. The PoC currently supports the allocation of a single patient at any one time. However, the demand predictor considers the numbers of people likely to be admitted in the future when making allocation suggestions.

Constraint type:

Bed allocation decisions increase in complexity depending on the number and type of constraint associated with a specific patient. During the discovery phase KGH staff provided us with a full list of constraints considered when allocating or transferring a patient.

For the purposes of this project, each constraint was described as either fixed or desirable. For example, if the patient has an isolating requiring infection then they must be placed in a side room, this is therefore a fixed constraint. If the patient is admitted under a medicine specialty then they should ideally be placed with other medicine patients in a medicine ward, this is a desirable constraint. Each constraint and its associated penalty (determined by KGH staff) was then encoded into the allocation model.

The hospital and patient based constraints that the PoC supports are in the restrictions can be found in the submodule of the codebase. Some constraints were not included due to a lack of data or their complex nature. However, this tool enables 'human in the loop' decisions by displaying multiple recommendations, other constraints can then be separately considered alongside the tool.

02.4 POC UI Functionalities

The PoC UI was built to enable KGH hospital managers to test the allocation model's capabilities and understand the demand forecasts generated using historical KGH hospital data. The overarching objective was to increase their confidence that ML could be used to support them with the very important and complex task of bed allocation.

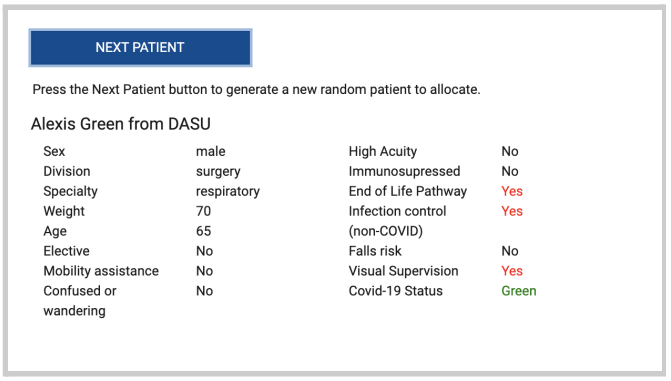
The four functionalities of the PoC tool are described and visually displayed below:

Launching an allocation:

Select the 'Next Patient'. A Patient is randomly generated to represent the patient currently awaiting allocation to a bed.

Information displayed will include:

- Basic demographic information (name, age, sex) of the patient;
- Assessment unit patient is coming from and associated division (medicine or surgery);



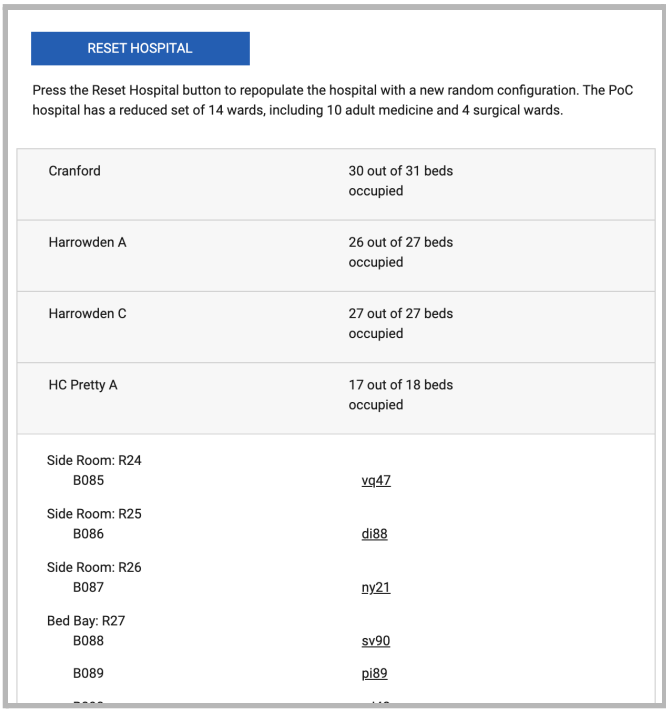
- Specific information related to the patient based constraints are also listed alongside their status;
- Relevant patient characteristics that will affect the patient's bed choice are highlighted in red and their covid status is also colour coded (Green, Amber, Orange) to draw the users attention;
- Each time 'Next Patient' is selected a new patient will be generated for allocation.

Populating the hospital:

The Current state of the hospital is randomly populated with 95% occupancy. The 'Reset Hospital' button allows the user to reconfigure the hospital in a new random state so they can see how changing the available beds will affect the allocation decisions.

A card for each ward is generated showing the number of available beds.

Each ward can be selected for a breakdown of specific bed bay and side room occupancy. Beds that are occupied have a two-letter, two-number patient 'name' next to them. Empty beds appear green and have the word 'available' next to them.

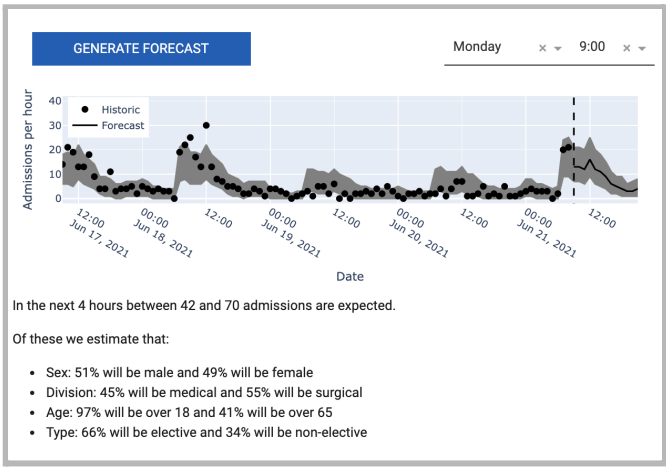


Understanding upcoming patient demand:

By default the forecast is generated from Monday at 9am. A user can select different dates and time using the drop down menus in the top right and select 'Generate Forecast' to recalculate the forecast for the different time period.

The 95% confidence interval of the forecast model is displayed as a grey band and the dashed vertical line signifies the current date and time. To the left of the current time, black dots represent the historic admissions across the past 4 days. For a well performing prediction the majority of those dots should line within the grey region. To the right of the current time, the number of predicted admissions is displayed, with the black line indicating the average forecasted amount and the grey bar again signifying the 95% confidence interval.

Below the plot we state the total range of patients predicted to arrive for the next 4 hours as well as the expected percentages of male/female, medical/surgical, over 18s, over 65s, and



elective/non-elective.

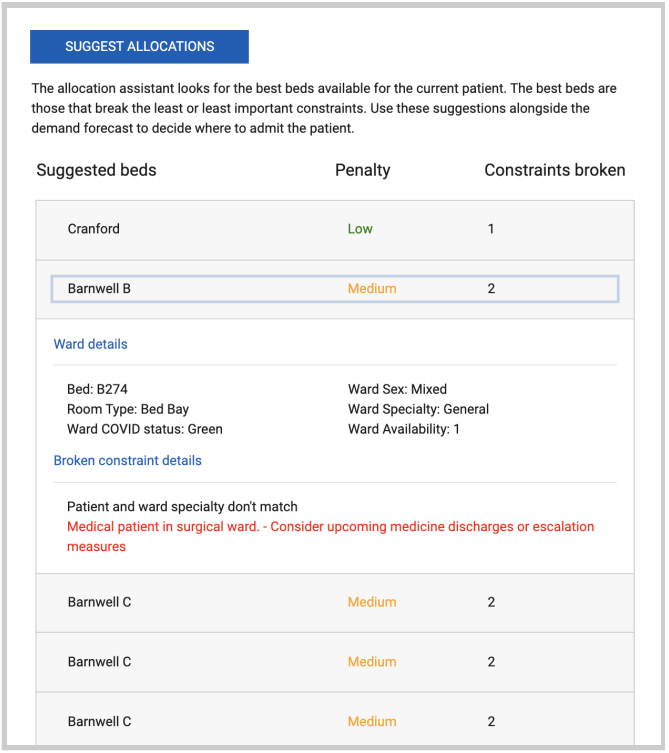
Analysing allocation recommendations:

Select 'Suggest Allocations' to generate a set of allocation recommendations for the current patient. If a user wishes to try a different patient or hospital state they will need to press this button again after refreshing those other components to get the new set of suggestions.

This will populate a prioritised list of five suggested beds. Next to each suggestion will be a penalty and number of constraints broken.

Each of the suggested beds can be selected to see a breakdown of the suggested ward details and broken constraint details.

The Penalty is the sum of penalties across each of the constraints broken by the suggestion (Low <3, 3≤ Medium, 10≤High). If a constraint that is considered unbreakable is violated by the suggestion (e.g., no male patient in female ward) the suggest will have a High penalty, but there will also be a warning to the user, highlighted in red, that encourages them to choose a different suggestion or alternative actions such as escalation measures.



03 Technical Approach

We developed an innovative AI-powered solution with three levels of sophistication. This chapter sets out the PoC technical approach, KGH data sources used, the three technical components and also provides an overview of the integrated PoC system. We describe how the technical approach could be refined in section 03.5.

03.1 Overview of Technical Approach for PoC

1 Discovery: Sprints One - Three

- ➔ **Identifying data:** Our team worked with the KGH team to review and mark all variables required for modelling within two data schemas, PAS and Patient Flow. In addition, information about bed allocation constraints and KGH wards was sought to further the development of the virtual hospital environment.
- ➔ **Data ingestion:** We scheduled multiple meetings with the KGH IT team to ensure that we received only the required data for the modelling activities scoped for this commission. The data was ingested onto Faculty Platform during sprint three of this project, four weeks after it was originally due, a two week project extension was therefore granted.
- ➔ **Defining technical approach**: MCTS was identified at the end of the first sprint as the most appropriate technique for allocating a patient to a bed between MCTS and Reinforcement Learning (RL). A summary of this comparison is provided in Appendix 6.5.

2 Data analysis and engineering: Sprints Three - Four

Given that we received five years of PAS data and two years of complex Patient Flow data, a process of data analysis, cleaning and engineering was undertaken:

1. Data quality, consistency and completeness of each field was assessed;
2. Table and variable data linkage was understood;
3. Long-term changes to the data both pre and post COVID-19 were analysed;
4. Test and training data sets for modelling were engineered.

3 Model build: Sprints Two - Six

- ➔ **Time series model:** One month worth of PAS data was initially explored during the first few weeks of the project, before an additional five years of data was ingested. This enabled the team to quickly build and train an initial time series for the number of patients admitted per hour in parallel with further data analysis activities. The demand predictor is further described in section 03.3.
- ➔ **Bed allocation agent in the virtual hospital environment:** We developed a first allocation algorithm ('greedy model') and tested its performance with dummy patients in a small virtual hospital environment during the first few weeks of the project, described in section 03.4. After data analysis and engineering tasks were completed the focus moved to improving the MCTS performance. Finally, as run time for the MCTS resulted too long to enable user engagement in this PoC, the greedy model was chosen as the solution to be integrated in the final tool.
- ➔ **Model integration:** The demand predictor and the greedy bed allocation model were merged with the UI during sprint five.

03.2 Data Overview

The following table describes the systems and data used to capture relevant information for model training and for the creation of a realistic hospital environment to train the bed allocation algorithm:

	System/Data	Description	Amount and type	Use
1	**System C Medway PAS System** (Patient Administration System)	PAS is used across the patient pathway at KGH to register, admit, transfer and discharge patients from hospital beds.	Five years (Jul 2016 - Jul 2021) Type: ● Demographic (Age, Sex) ● Specialty ● Inpatient location ● Ward information ● Number of beds	We requested specific variables from the PAS data schema for the forecasting of admissions and for the bed allocation model. We used this data to ensure that the allocation model was trained on real data.
2	**Patient Flow System**	Patient Flow is used to record relevant patient alerts and information about a patient's individual needs. It has a hospital, ward and patient level view.	One - two years (this system was only recently installed. ● Individual status/alerts relevant to constraints e.g. COVID-19 status	This information was used to encode the spread of constraints information into the patient generator to determine the likelihood of something occurring.
3	**Constraints and penalties spreadsheet**	The constraints were listed by operational managers and provided to us at the start of the project. We later asked for the penalties against these constraints to acknowledge weighting.	● 23 constraints were encoded ● 17 were deemed unsuitable for encoding at this stage due to a lack of digitised information or complexity e.g. patient ward preference.	The constraints were encoded into the allocation model. This was to ensure that the allocation model considered them when making a recommendation. Unmet constraints are displayed on the UI.
4	**Operational KGH site team spreadsheets**	KGH operational managers currently use spreadsheets to record hospital capacity information e.g. numbers of side rooms.	Three spreadsheets containing: ● Bed count ● Ward names ● Side room count As of July 2021.	We used some of this information to create the virtual hospital environment as we wanted this to be as accurate as possible.

03.3 Demand Predictor

In order to inform the bed allocation process we developed a demand predictor to forecast the number of patients due to be admitted into the hospital. We describe this forecasting model in detail in the subsequent sections.

03.3.1 Bayesian modelling approach

We adopted a Bayesian modelling approach using numpryo and jax. The fundamental idea behind this approach is that the model itself is considered to be a statistical object. In particular, the model parameters that define the model are interpreted as being drawn from a distribution defined using Bayes' Theorem (equation 3.3.1). This states that given some data *D* and some model with parameters θ, the probability distribution of these parameters given the data is defined by the following expression:

$$p(\theta|D) \propto p(D|\theta)p(\theta) \quad [3.3.1]$$

Where $p(D|\theta)$ is known as the likelihood function and is the functional form of the model that we fit to the data. The term $p(\theta)$ is the prior distribution of the parameters and informs a 'best guess' of the parameter space of the model, and encodes pre-existing knowledge about the problem. For example, if we know that admissions can't have negative values, we would choose a prior that prevented this. $p(\theta|D)$ is then referred to as the posterior distribution and represents the true distribution of arriving patients given the data that we have previously seen and the domain information we have encoded in the prior.

The output to the model is a full posterior distribution for the predicted number of admissions rather than a single number. This allows us to draw samples for the predicted number of admissions and generate confidence intervals over our estimates.

03.3.2 Combining the components of the admissions model

. The admissions forecast model consisted of four components which account for:

- The long term trend: to account for variations such as the current rise in admissions after the downturn during COVID-19 lockdowns;
- The day of the week: to account for variations connected with specific days in the week, e.g. admissions are typically lower on the weekends;
- The hour of the day: to account for variations connected with the time of day, e.g. fewer patients are admitted at night than during the day;
- Whether it is a bank holiday: to account for variations that are due to these specific dates, e.g. fewer patients are admitted on bank holidays.

We captured the long term trend in the historic admissions data using a Gaussian Process (GP), inspired by previous work of Vehtari et al., as summarised in this blog post. A GP does not constrain the model to take on any particular form. Instead, it returns a distribution over the functions which are consistent with the observed data, in this case, the historic admissions timeseries. In practice, exact GPs can be inefficient to calculate, we therefore utilised the Hilbert Space approximation based on this numpyro tutorial to ensure tractable runtimes. The other short-term trends were then incorporated into the model as linear addition, resulting the the following model:

$$y \sim NB(f, \sigma) \quad [3.3.2]$$

Where y is what we are trying to predict, i.e. future patient admission numbers, which is drawn from a negative binomial distribution with parameters f and σ. The parameter σ is drawn from a half normal distribution with standard deviation = 0.5 and f is given by:

$$f = intercept + f1 + \beta_{day\ of\ week} + \beta_{hour\ of\ day} \quad [3.3.3]$$

The intercept is taken from a normal distribution (mean = 2, standard deviation = 1) and f1 is the Guassian process used to define the long term trend. In the equation above, $\beta_{day\ of\ week}$ is the effect due to the day of the week, with the parameters for Tuesday to Sunday being drawn from a normal distribution with mean = 0 and standard deviation = 1. The parameter for Monday is then set to the sum of the parameters for Tuesday to Sunday, multiplied by -1. The effects of each day of the week then sum to 0, representing the deviation from the average number of admissions per day. The parameter $\beta_{hour\ of\ day}$ is similar, but for hours of the day, where the parameter for hour 23 is the sum of the parameters for hours 0 to 22 multiplied by -1. The bank holiday effect is combined into the day of the week effect, where bank holidays are assumed to have similar admission levels to Sundays.

03.3.3 Training the admissions model

We trained the model on previous admissions data, aggregated to show the total number of admissions (medical and surgical) every hour for the past 120 days. We chose this as our training period because it allows enough time for both the short and long term trends to show up within the data, whilst reducing the effect of the abnormalities in admission caused by COVID-19. Once the model has been trained, we can then use it to predict the number of patients that will be admitted each hour. Rather than a single number, the demand predictor produces a posterior distribution, meaning we can draw samples for the predicted number of patient admissions. We can also use the posterior to generate confidence intervals for our model. Here we use the 95% confidence interval, which means that 95 times out of 100 we expect the patient admission numbers to fall within the range provided.

03.3.4 Generating admissions model outputs

The two plots in Figure 3.1 show the short term (top) and long term (bottom) trends in historic admissions data provided to us by KGH as blue points, along with the output from the demand predictor as a grey band. The width from the bottom to the top of the grey bands represents the 95% confidence interval and shows the range in values we might expect for patient admissions.

In the top plot we can see the daily and weekly pattern in hospital admissions, with more patients coming in during the day and on weekdays than during the night and on weekends respectively. The bottom plot shows the long term trend, with the number of patients being admitted gradually rising with a bump in admissions around November 2020.

03.3.5 Validating demand predictor performance

In order to check the performance of the demand predictor, we need to validate the predictions against historic admissions data. We can do this by training the model on historic patient admissions data, for example on the 120 days between 01/01/2021 and 30/04/2021. We then generate a forecast for the next 7 days, in this example between 01/05/2021 and 07/05/2021, and compare it to historic data from that time period. We can look at where the admitted number of patients for each hour lies compared to different confidence intervals and check that the correct amount of historic data lies within each confidence interval, for example 50% of the actual patients that arrived should fall within the 50% confidence interval predicted by the model.

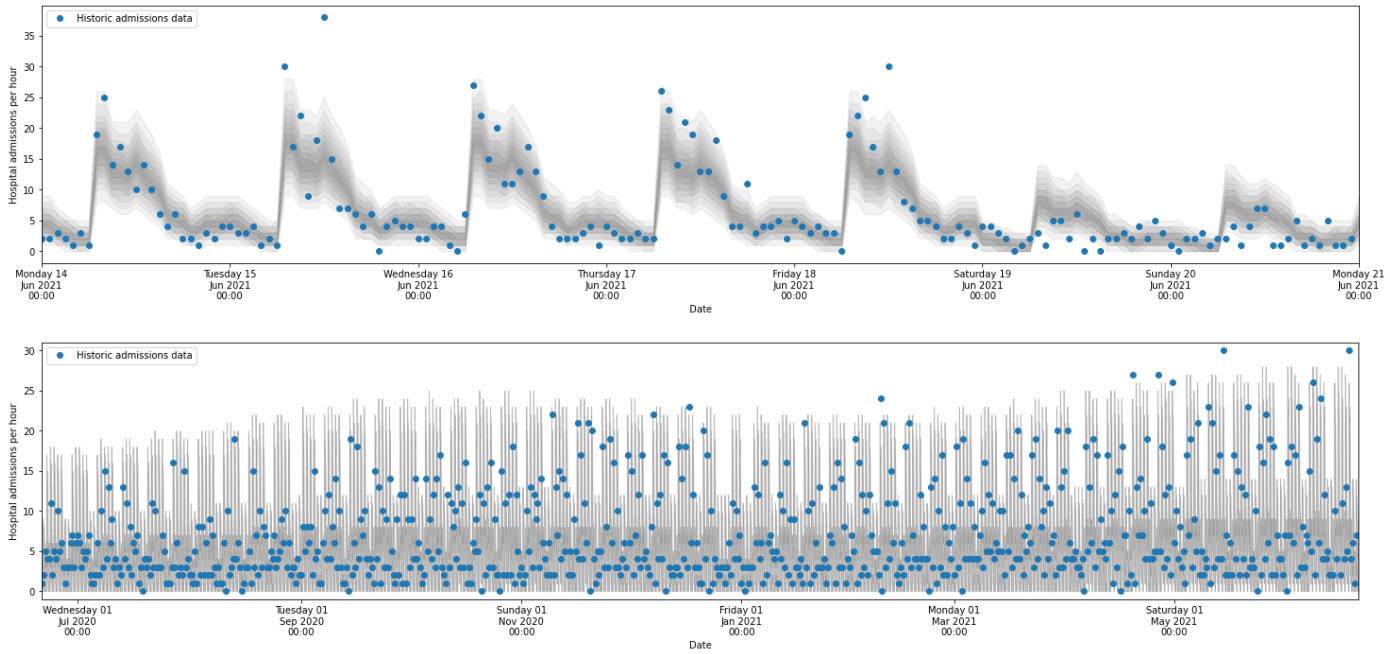


Figure 3.1: True historic admission data from KGH alongside forecasting results. Top: short term, bottom: long term. Historic data is shown with the blue points, the output of the demand predictor is shown in grey. The width of the grey band represents the 95% confidence interval and shows the range in values we might expect for patient admissions.

We chose 20 random dates during the period that we have data for (01/07/2016 - 01/07/2021) to perform validation on. For each of these 20 dates, we checked that there was sufficient data before that date to train the model (120 days) and sufficient data after that date to validate the model (7 days). We also checked for each of the dates the training and validation periods did not overlap with the start of the COVID-19 pandemic (defined for this purpose as 20/03/2020). The reason that we did not extensively test this model's performance over the pandemic period is that this model was not designed to predict the sudden drop in planned admissions and rise of COVID-19 admissions caused by the pandemic. For modelling admissions during COVID-19 a different functional form is recommended, such as the epidemiologically inspired approach adopted in the Early Warning System (NHS and Faculty) .

Figure 3.2 shows 6 examples of validation plots on different dates, with the date for each plot given in the title and shown in the plot as a black dashed line. In each of these plots, the blue points show a subset of the 120 days worth of training data before each date, and the grey points are 7 days worth of validation data after each date, with the training and validation data both having come from the data provided to us by KGH. The coral bands then show the 95% confidence intervals for the model. The day of the week and hour of the day effects are evident within these plots, as you can see the drop in admissions on weekends and during the night. The effect of the long term trend is also evident, as the forecast reflects the drop in admissions on 19/07/2020 compared to 10/03/2020.

To get a measure of how well our model is performing, we can generate plots to show where the validation data points land relative to the confidence intervals from the forecast, such as the one in Figure 3.3. For example, if a validation data point was right in the centre of the forecast band it would be placed on percentile 50 in this plot. The x axis then shows the time in hours between the validation data point and the date the validation was performed on. For a perfect model, 50% of the points would lie within the darkest coral band, 80% within the next band and then 95% within the lightest band. Figure 3.4 shows the results of this test - 48%, 78% and 94% of the validation points fall within the 50%, 80% and 95% confidence intervals respectively. In summary, the results of these tests show that our demand predictor is incredibly accurate and provides the users with a reliable indication of both incoming patient demand, and the variability within it.

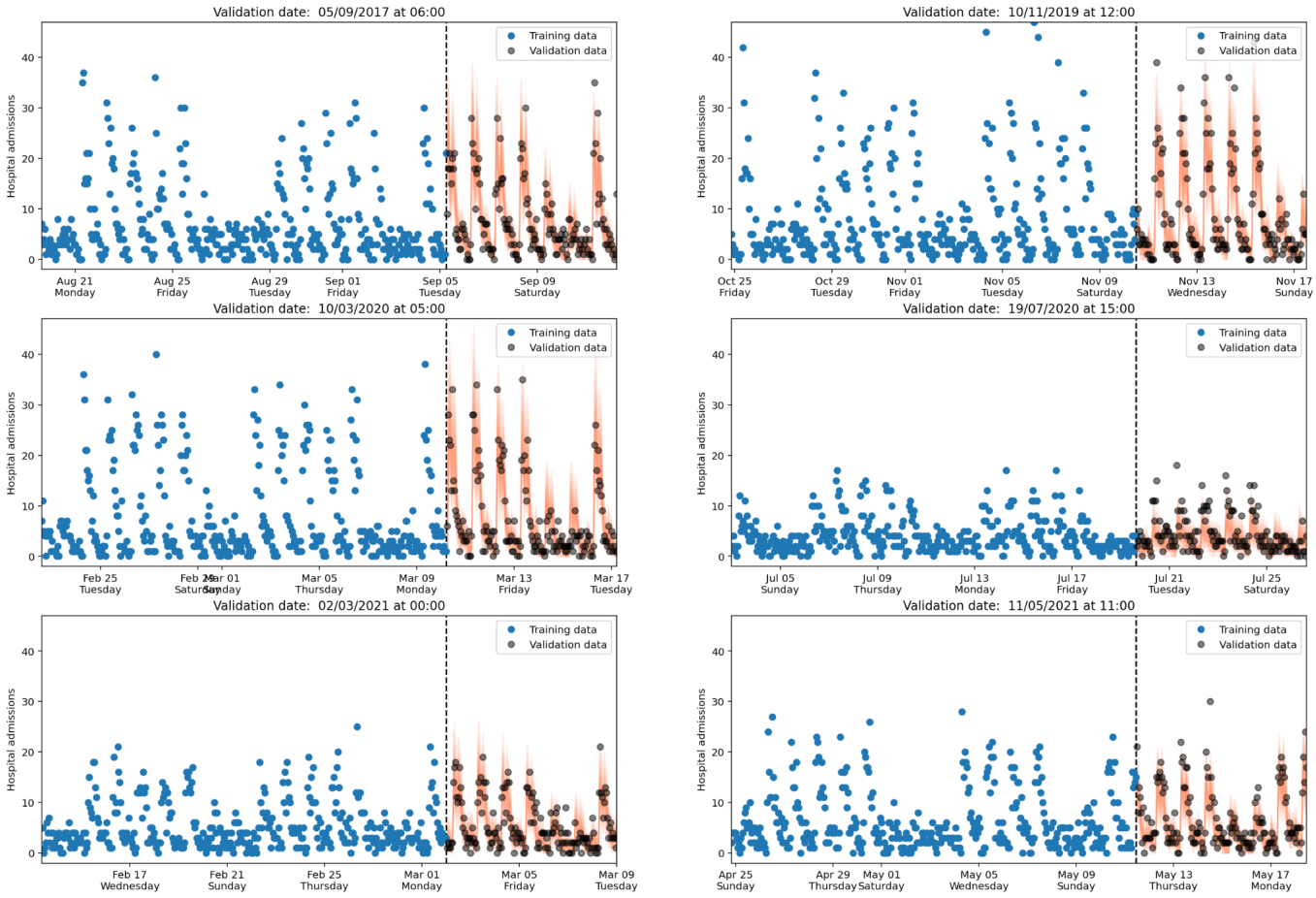


Figure 3.2: Model validation on 6 randomly selected dates during the period we have data for.

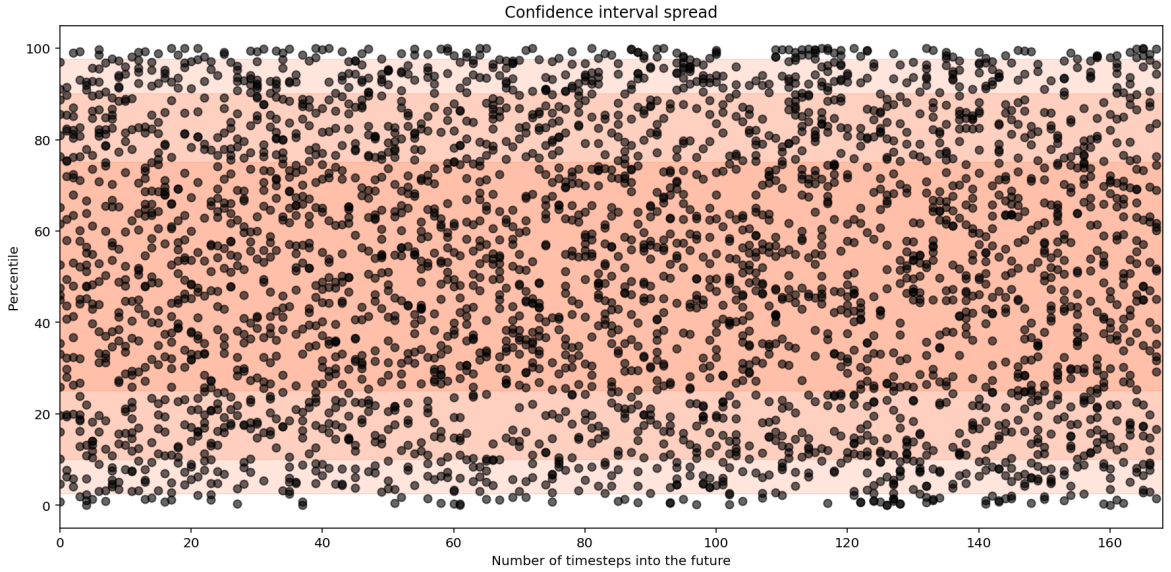


Figure 3.3: Comparison of historic data to confidence intervals from the forecast. From darkest to lightest, the coral bands show the 50%, 80% and 95% confidence intervals.

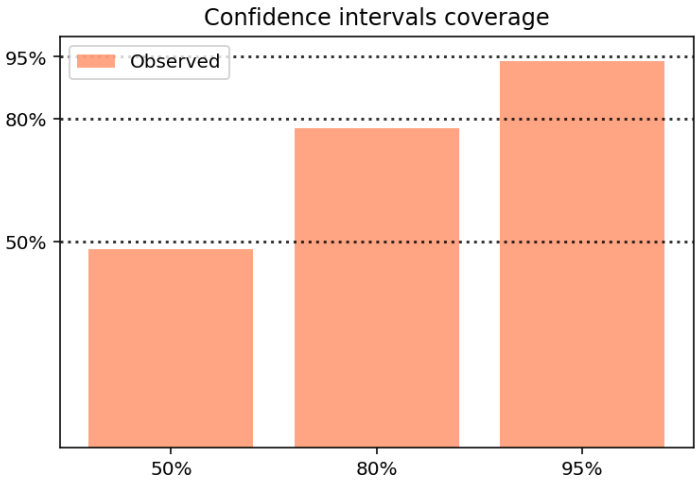


Figure 3.4: Comparison of historic data to confidence intervals from the forecast.In this example, 48%, 78% and 94% of the validation points fall within the 50%, 80% and 95% confidence intervals respectively.

03.3.6 Sampling patient characteristics

The demand predictor helps to inform the process of allocating a patient to a bed by providing information on patients that may be admitted in the future. In order to be as informative as possible, it would be useful for the demand predictor to predict not just how many patients will be admitted to hospital, but their attributes, such as their sex, age, what division and specialty they will admitted into, whether they are elective, and their COVID status. An example of where this is useful could be when a bed manager is trying to allocate a bed to a male patient, but knows a lot of female patients are due in the next few hours. Instead of allocating this male patient to an empty male bed bay, they could then instead consider flipping the gender of this bay to female to accommodate this future demand, and find the male patient a bed elsewhere.

However, modelling the joint distribution of all of the patient attributes and how they vary by time soon becomes intractable due to the number of possible combinations. It was not feasible within the timelines of this PoC project to model each of these attributes, but we still wanted to be able to provide this information along with the projected numbers from the demand predictor. To get around this problem, we sampled patients from historical admissions data. From PAS and Patient Flow, we built a dataset of patients who had been admitted, their attributes, along with the day of the week and the hour of the day they were admitted.

Take the example of a forecast starting from a Saturday at 11am. The demand predictor provides us with the full posterior distribution for how many patients are expected in the next 4 hours. Let's assume we take one sample from this distribution which tells us that in the next hour we expect 6 patients, in the hour after that we expect 8 patients, etc. For the first hour, we then sample 6 patients from our dataset who were admitted on a Saturday in the hour starting at 11am. For the next hour, we then sample 8 patients from our dataset who were admitted on a Saturday in the hour starting at 12pm. We then repeat this 2 more times and end up with a series of realistic patient admissions for the next 4 hours. This process of sampling the posterior distribution and then sampling patients is repeated 1000 times, so we end up with 1000 series of realistic patient admissions for the next 4 hours. We can then look over all of those patients and find the percentage male/female, medical/surgical, etc., which is what we display to the bed manager on the UI.

Relevant code demand predictor

The code for the demand predictor is contained within the `forecasting` submodule. Please note this code will not be able to be run in absence of the historic admissions data from KGH used to train the model. Guidance on how to validate the model is provided in the pdf version of the notebook `model_validation.pdf` and guidance on how to train the model is provided in the pdf version of the notebook `train_predict.pdf`.

Code location	Description
`src/forecasting/time_series_model.py`	Functions to define model components
`src/forecasting/forecast.py`	Forecast class
`src/forecasting/patient_sampler.py`	Patient sampler class to provide samples to MCTS allocator

03.4 Bed Allocation Agent in the Virtual Hospital Environment

We first created a virtual hospital environment as the test bed upon which to develop a bed allocation framework. Using this environment we subsequently explored two approaches to the bed allocation agent:

- Greedy optimisation
- Monte Carlo Tree Search (MCTS)

Each of these three components are described in detail in the following subsections.

03.4.1 Virtual hospital

The virtual hospital environment comprises several components that can be tailored to mimic any arbitrary hospital structure, allowing the user to test the agent at different scales. It is based on a tree-like structure using the anytree library to define the hierarchical structure of a hospital.

The user can build virtual hospitals containing the desired number of wards, rooms and beds. In addition, the virtual hospital encodes the allocation restrictions and associated penalties that apply to the hospital, as well as the data structure for patients. We have encoded different types of ward (medical, surgical) and rooms (bed bays, side rooms) to cater for a broad range of allocation rules. In addition, certain restrictions apply specifically to a patient and are thus contained within the patient class (e.g. if a patient requires a sideroom).

When developing the algorithms for this commission, we utilised KGH estates information, and learnings from user interviews to define the PoC hospital. This included 14 wards (10 adult medicine and 4 adult surgical) with a total 303 beds. This is less than the more than 550 bed base of KGH, as for the purposes of the PoC we excluded paediatric and maternity beds. For further details of the allocation constraints please refer to the restrictions submodule of the codebase.

Relevant code virtual hospital

The code for the virtual hospital is contained within the `hospital` submodule. Guidance on how to create a virtual hospital is provided in the notebook 1.Virtual_Hospital_Environment.ipynb.

Code location	Description
`src/hospital/building/`	Hospital structures: wards, rooms etc.
`src/hospital/equipment/`	Bed class
`src/hospital/restrictions/`	Restriction classes
`src/hospital/people.py`	Patient data class
`src/hospital/data.py`	Set of Enum classes used to validate patient and ward characteristics.
`src/hospital/exceptions.py`	Set of custom exceptions that are raised throughout the submodule.

03.4.2 Greedy optimisation

Greedy optimisation finds the best allocation currently available given the state of the hospital and attributes of the patient. Consider the following example:

- A female adult medical patient has arrived at the hospital and there are only two available beds. The first is within a bed bay of the male medical ward, the second is within a bed bay of a female surgical ward.
- According to user research the penalty for assigning a patient to ward with incorrect sex is 10 whilst the penalty for assigning a medical patient to a surgical ward is 3.
- The greedy algorithm would choose the second bed, finding the solution that incurs the lowest penalty.

In a more complicated scenario, where multiple penalties of varying cost may apply, it will optimise for the lowest aggregated penalty, which is a balance between the number of constraints broken and the magnitude of their costs i.e., breaking 1 constraint at a cost of 10 is worse than breaking 2 with a total cost of 7.

This approach is called 'greedy' as it performs a brute force search across all available options. This may be slow if the size of your search space (e.g. number of possible choices) is very large. Therefore, we implemented an intermediate step that finds equivalent beds and reduces these to a set of representative beds. This method reduces the search space of the 303 beds in the PoC hospital to 67 beds, the runtime of this approach is discussed in the results section.

During development we have run multiple experiments with the greedy allocation algorithm on the PoC hospital. The primary aim of these experiments was to determine if this approach is operationally feasible in terms of compute-time. We ran the greedy allocator several times, for randomly generated patients and initialised hospital states. Figure 3.5 shows how the runtime in seconds scales with the number of empty beds. We can see that even with the hospital completely empty the greedy allocator can return the optimal bed suggestion within 0.5 seconds, which is well within the operational allowance of 1-5 minutes stipulated by KGH.

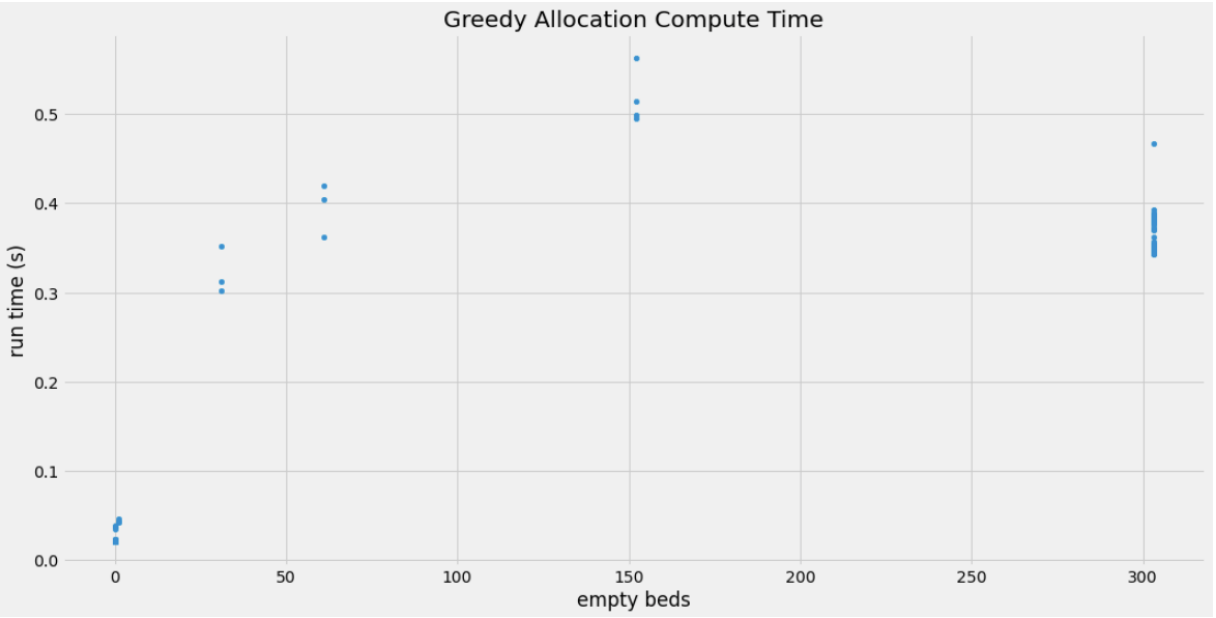


Figure 3.5: Runtime of the greedy allocator.

Relevant code greedy optimisation

The code for the greedy optimisation is contained within the `agent` submodule. Guidance on how to use the agent is provided in the notebook 2.Greedy_Allocation.ipynb. It is also plugged into the UI api and can be interacted with using the UI.

src/agent/policy/ contains all of the functions that perform greedy optimisation, including the functions to reduce the search space to a set of equivalent beds.

03.4.3 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a popular optimisation algorithm often applied to identify the best decision given a complex problem. For example, AlphaGo and AlphaZero projects by Deepmind that utilise MCTS to win board games against grand masters. A full review of MCTS is available in Browne et al., 2012, and has been the basis of the below work.

Broadly speaking MCTS randomly explores a decision space, building a tree of connected moves and their associated rewards. Over multiple iterations the MCTS determines which decisions are most likely to result in a positive outcome. Ultimately the best decision is the one that provides the best long term reward, the definition of which depends on the specific domain. For example, when creating a MCTS agent to play a board game, the long term reward would be a 0 or 1 depending on whether you won or lost the game after making the current move.

In the context of bed allocation we do not have a natural end state. Therefore the long term reward is determined as the total reward incurred after N time has passed according to the equation 3.4.1. Here R_n represents a reward associated with the state of the hospital at a given time step, $\gamma \epsilon [0,\ 1]$ is the discount factor. The reward associated with a hospital state is 1 - total penalties incurred. The first term in the equation, R_1 , is the immediate reward associated with the current allocation i.e., the greedy allocation score, and the subsequent terms are rewards associated with future states, where the discount factor weighs the relative importance of these future states against the current state.

$$R = R_1 + \gamma R_2 + \gamma^2 R_3 \ldots + \gamma^N R_N \quad (3.4.1)$$

Below we describe the four stages of the MCTS algorithm as they are specifically implemented for the bed allocation agent. The implementation utilises the anytree library to build a tree structure, where each node represents a specific state of the hospital and each level of the tree represents a time step. Time steps are incremented in hours and connected to the number of forecasted admissions arriving each hour. The input to the treesearch is a queue of patients arriving at each time step, with the current patient to be allocated (t=0) as the first entry in this queue, and the current state of the hospital as the root node to search from.

1. **Selection**
   Starting at the root node a child node is selected. The root node represents the current state of the hospital at time t=0 where the state is defined by the patients that are currently occupying beds and the set of empty beds.

   In the first iteration of the tree search, the algorithm selects the root node and moves to the expansion step. In subsequent iterations it will traverse from the root and choose one of the child nodes according to the *tree policy*. The *tree policy* is the UCB score (equation 3.4.2), in which $\overline{R}$ is the mean reward from visiting that node, N_p is the number of times its parent node was visited and N_i is the number of times the node itself has been visited. The first term encourages the algorithm to select nodes that have previously resulted in good outcomes, while the second encourages the algorithm to explore options that it hasn't visited as often.

$$UCB = \overline{R} + \sqrt{\frac{2\,log(N_p)}{N_i}} \quad (3.4.2)$$

2. **Expansion**
   Once a node is selected, a child node is attached to represent each possible decision state for that time step.

   For example, we have a hospital with 4 beds, 2 are occupied and 2 are available. We are currently trying to allocate a patient P1. As there are two possible decisions, two nodes can be attached that represent 1) allocating P1 to the first available bed and 2) allocation P1 to the second available bed.

   As we progress through the tree search, we may encounter time steps where multiple patients have arrived. In such cases, a node is expanded for each possible combination of patients to available beds.

3. **Simulation**
   From one of the attached children we then simulate a future. The simulation stage involves the following steps:
   a. Each patient currently within the hospital has a length of stay attribute, and an expected length of stay attribute. At the start of the simulation step, the length of stay counters are incremented by one.
   b. Then a discharge model is applied to discharge existing patients. The probability of being discharged increases according to proximity to your expected length of stay.
   c. The patients arriving in the given time-step are then assigned to beds according to the *default policy*. The default policy is a random assignment of patients to beds to available beds.

4. **Backpropagation**
   The total penalty of the hospital is calculated after the simulation step. This is the sum of all penalties for each broken restriction within the hospital. We then backpropagate this score up the tree to distribute the outcome across all decisions along the currently explored path.

This stage updates the UCB score (tree policy score) and visit count for each node that was traversed along the current decision pathway. If the result of the simulation was good, the UCB scores of each node will have increased, making it more likely that future iterations of the tree search will select these nodes again. The reverse is also true. In this manner MCTS is more tractable than a completely random search of the possible decision pathways as it more frequently visits the most promising options during the selection stage.

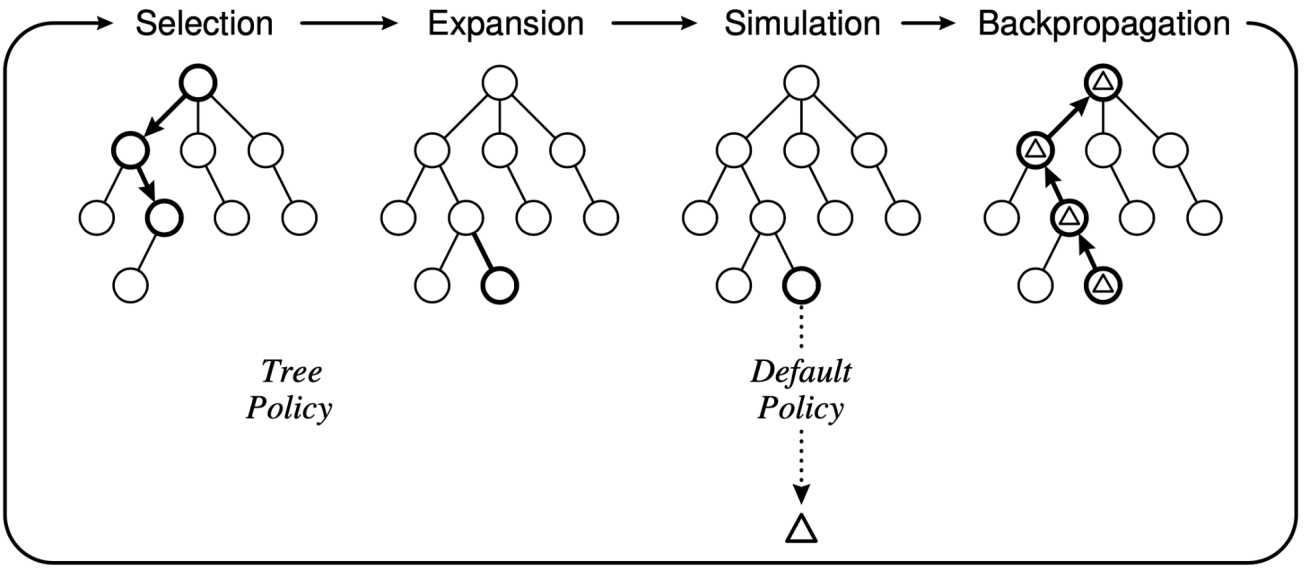


Figure 3.6: The four stages of MCTS algorithm, Brown et al (2012)

The above procedure is repeated multiple times until a maximum number of iterations have been reached. At this point the tree object is returned and the best child node of the root is selected as the optimal allocation for the current patient. There are several potential strategies for determining what the best node is. In the current implementation we choose the node that has the highest visit count. Alternative approaches such as choosing the node with the highest UCB score or some balance of the two, and how this affects outcomes, remain to be explored in future work.

Limitations

In the above implementation we take a single sample from the demand forecast and use this as a fixed version of the future. This means that the future within the treesearch is deterministic and significantly reduces the search space, and branching of the tree, allowing the algorithm to find a recommendation in a more tractable timeframe. However, a single sample from the forecast represents one of the possible futures. To truly capture the variability of incoming patients, we envisage a strategy where multiple simultaneous tree searches are implemented, each using a separate sample from the forecasted admissions. These could be run in parallel to increase runtime efficiency, and the final suggested allocation would be the bed that has the average highest ranking across the ensemble of tree searches. The efficacy of this strategy and alternative approaches to dealing with non deterministic search spaces remain to be explored.

Despite fixing the set of arrivals within a tree-search we can still experience an intractable amount of branching that makes the current implementation of MCTS unsuitable for operational use. For example, if there are just 4 empty beds in the hospital, and 9 arriving patients within a time step, the tree expands 840 possible permutations of patients to beds. With multiple time steps into the future this can compound and result in either memory issues or extremely long compute times. Figure 3.7 indicates the types of runtime we have encountered during testing using the 4-core, 16GB CPU. The x-axis shows the number of empty beds, and the y-axis is the compute time in minutes. In these experiments we limited the maximum number of arrivals to 4 per hour and only forecast ahead for 4 hours (time steps) into the future, even with these limits the runtime quickly rises to 30 minutes with just 8 beds, and over two hours with 16 beds, beyond that we ran into memory allocation issues. Further work is needed to explore engineering strategies that can make MCTS more operationally feasible.

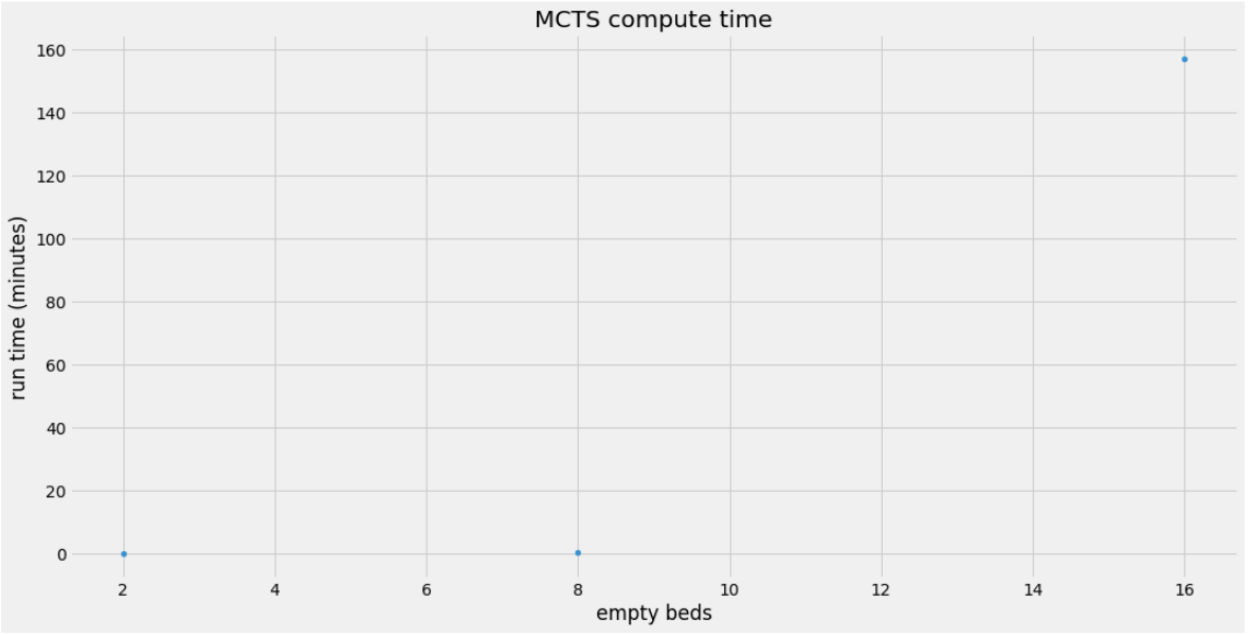


Figure 3.7: Runtime of the MCTS allocator.

Relevant code MCTS:
The code for the MCTS is contained within the `agent` submodule. Guidance on how to use the agent is provided in the notebook 3.MCTS_Allocation.ipynb.

Code location	Description
`src/agent/mcts.py`	Node class for MCTS algorithm
`scr/agent/simulation.py`	Simulator class for the simulation step
`src/agent/policy.py`	Random allocation (default) policy used during simulation
`src/agent/utils.py`	several helper functions that are used throughout the code base.
`src/forecasting/patient_sampler.py`	Patient_Sampler class which provides the forecasted arrivals to the MCTS.

03.4.4 Comparison of Greedy and MCTS

Table 03.4 below summaries the key differences between MCTS and greedy allocation with regards to several important aspects.

Feature	MCTS	Greedy
Predictive power	Able to incorporate knowledge about the future to return the best long term allocation suggestions. The efficacy of this approach is contingent on the accuracy of the forecasting model that simulates the future.	None - can only provide the best solution given the current state of the hospital. The user is required to leverage information from the forecast and their own expertise to determine the best long term option.
Explainability	Currently only explains the immediate set of restrictions violated by an allocation. Significant further development would be required to surface how the forecast has affected the decision. Whilst theoretically plausible, such additions would not be computationally tractable within the required timeframes. Instead, users may be required to leverage information surfaced from the forecast and training will be necessary to ensure they are aware of any associated limitations.	Fully explainable, and easy to interpret as it encodes the set of rules that end-users are already familiar with.
Overall computational time and cost	In this first implementation it takes too long time to compute for real-time decision making and the overall cost would be too high in a hospital-based environment.	Is quick to compute and could return a solution within operational timeframes, even on a standard machine.
Retraining/Tuning	Other than the demand predictor component, retraining is not required for this method. If the layout of the hospital were to change, the virtual hospital component can be modified and MCTS can run in the new layout with low overhead. Tuning the decision outcomes is more challenging and further work is needed to validate model decisions with end-users and determine how hyperparameters like the discount factor affect the efficacy of decisions.	Retraining is not required for this method. If the layout of the hospital were to change, the virtual hospital component can be modified and MCTS can run in the new layout with low overhead. Tuning the decision outcomes is more straightforward as it requires collaborating with the end-user to define the magnitude of penalties. As with the MCTS option, the demand predictor will require a continuous training pipeline. However unlike with MCTS, the efficacy of allocation suggestions are not contingent on this component.

3.5 Recommendations for further model development

Given its efficiency, tunability, and interpretability, we would recommend early bed allocation products utilise greedy optimisation. MCTS shows some promise in this domain despite the identified limitations and we recommend further development time in this area. Overall, we recommend the following areas for future exploration:

- ➔ **Refining the technical approach:** Exploring the literature to find implementation improvements to MCTS that could improve runtime efficiency. These may include engineering multithreading or other parallelisation methods around the simulation step of the algorithm. We would also recommend revisiting alternative optimisation algorithms such as the mixed-integer model proposed by Berg et al.

- ➔ **Access to other systems and data:** Any algorithmic approach is ultimately limited by the level of information that it has access to. Even with greedy allocation, there are important patient or building level characteristics that are not currently digitised. Exploring other systems e.g. vitals, Care Flow connect may help bridge these gaps but KGH will need to ensure they are capturing all the necessary information.

- ➔ **Additional model and feature development time:** For this PoC we limited the scope to direct admissions from assessment units into wards. However, there are several other modes that can be realistically explored by an allocation agent, such as moving existing patients to different beds, escalation measures etc. More development time is needed to incorporate these added levels of complexity to the model and crucially, several of these aspects are contingent on robust (preferably live) data feeds that capture the state of the hospital.

- ➔ **Further user testing and validation:** Given that the aim of an allocation agent is to ultimately assist human users, further user testing is essential to understand what an operationalised product would look like. In addition, a proper validation framework needs to be developed that can quantify the performance of the allocation agent. This will require fully connected, and fully informed data flows that can be used to track the state of the hospital at the given time, and eventually measure the impact of a specific allocation decision. In our limited time with the PAS and Patient Flow datasets we identified that tracking the state of the hospital is theoretically possible, but currently has insufficient information to allow proper validation. For example, patient acuity information, or number of previous moves are not currently available, discharge information is also unreliable. Significant work also needs to be done to determine the appropriate KPI for such a validation process.

- ➔ **Deployment pipelines:** Due to lack of live data feeds the Bayesian demand predictor is being utilised as a pre-trained object. A deployed version of the tool will require a MLOps pipeline that is able to retrain the model regularly to provide the relevant forecast for the day/shift. In addition, a final model may want to incorporate additional information such as leading indicators (111 call volume, mobility data, covid data) to improve the robustness of the forecasts in the face of unpredictable events.

The next chapter details the outcomes from the thematic analysis performed during the discovery period and suggested next steps to ensure that the value from adoption of AI to all areas of bed management can be realised across the NHS.

04 Next steps

04.1 Overview

Optimising the use of existing available hospital resources and bed capacity has been a long existent challenge for all acute NHS hospital Trusts. NHS overnight available bed levels were steadily decreasing for the past decade pre-pandemic and were already lower than all comparative European countries. The NHS long-term plan is focused on reducing pressure on acute care by improving the connections between inpatient and outpatient settings. But when inpatient care is required, placing the right patient, in the right bed, at the right time has been proven to reduce length of stay, improve patient experience and overall quality of care.

COVID-19 has put unparalleled pressure on Trusts such as KGH. Capacity has had to be organised in new ways and optimising the placement of patients to meet their individual care needs has become increasingly difficult. We were engaged to build a PoC to demonstrate how AI could be used to support site managers to allocate patients to beds by taking into account a large number of hospital and patient based constraints. This report has detailed the overall scope, technical approach and results of the PoC but our initial discovery conversations with senior KGH managers identified other challenges and opportunities where AI could be used to improve operational workflows.

This chapter describes how cutting edge data science and AI techniques could be applied to solve three key challenges identified during our discovery to bring immediate operational value to KGH and the wider NHS.

04.2 Key challenges identified

We performed a series of discovery interviews with six senior KGH hospital managers that informed both the development of the PoC and our understanding of the wider context of bed management - the process of optimising the admission, transfer and discharge or flow of patients through the hospital. Thematic analysis was then used to generate three common themes at three different levels of the trust from interview recordings. These are outlined below.

Level	Challenge and Impact	User quotes
Hospital	**Challenge:** Disparate view of the Trusts total capacity and demand. ➔ Site and other hospital managers spend valuable time using multiple systems to generate manual reports up to six times per day in an effort to understand the Trust's current position. **Impact:** Staff under immense pressure have to quickly make capacity and escalation decisions using out of date information. This unintentionally compromises the safety of staff and patients.	*" Our biggest challenge is not having a central pulse of Patient Flow information. I still populate a manual spreadsheet to give me an overall Trust position* *" We need to have the whole picture to be able to hold our nerve. If you can't do anything for 11 hours you shouldn't just rush something at the end because chances are you'll get it wrong*

Ward	**Challenge:** Limited ability to see the impact of escalation and reconfiguration decisions ➔ Pre COVID-19 the bed base was a lot more static but now KGH is regularly at higher Operations Pressure Escalation Levels (OPEL). Wards and bed bays have to be reconfigured routinely to cope with surges in demand and resultant reduction in capacity. **Impact:** Heavy reliance on experienced staff members to foresee the impact of their escalation decisions during key capacity meetings. Experienced staff are not always available, leaving less experienced members to weigh up difficult decisions and execute.	*" When and how should we best utilise our escalation areas?* *" If it's bigger decisions, like a whole bay flipping from male to female, the impact of that will be discussed in bed meetings* *" If you don't have enough capacity then following the site meeting you have to formulate a plan to find it. Only then can you handover to the evening shift and feel comfortable that there are various routes to explore*
Patient	**Challenge:** Highly complex combination of fixed and desirable constraints to be considered for optimal allocation ➔ Experienced staff members currently balance fixed and desirable constraints in their heads. Less experienced staff members struggle to consider the current and future status of the hospital. COVID-19 further complicates ADTs. **Impact:** Blocks consistently occur in ED and Assessment units in part due to the complexities involved in considering where multiple patients can be placed, moved or discharged. Pressure falls on the shoulders of a small number of more experienced staff to make these difficult decisions.	*" We've got to factor COVID-19 into our decision making now...it makes your number of moves even more limited* *" I know that this is sometimes referred to as Tetris but I think of this as more like chess. It is not anticipating the next move, it is about anticipating the next 4 or 5 and takes skill and experience*

04.3 Bed management product vision

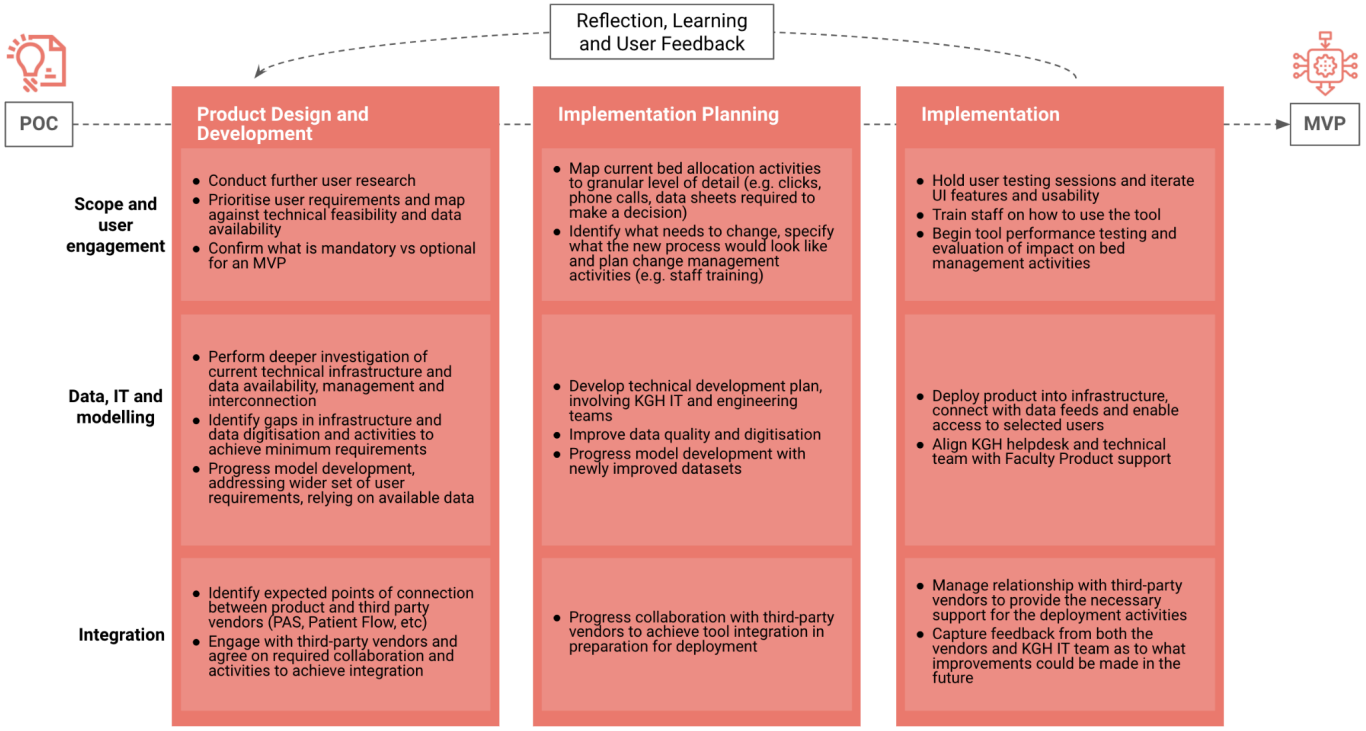
Our PoC has explored ML/optimisation capabilities to address some of the challenges highlighted above. Specifically, our solution focused on showcasing the value of providing a comprehensive view of the hospital current capacity and expected demand (the 'Hospital' challenge), and on incorporating known constraints in the calculation of optimised bed allocations (the 'Patient' challenge). What we've built has been useful to highlight the value and potential of supporting operations in a hospital setting with AI, in a way that should integrate seamlessly into current practice and lead to improved services and staff and patients' wellbeing.

The capabilities of the PoC need to be built upon to further support allocation decisions for individual patients and to optimise complex flow decisions at a patient, ward and hospital level. This would result in a comprehensive tool that would address all the challenges we have identified in our engagement with users. The core AI elements of this tool are described below, with an indication of what has already been achieved in this PoC.

Challenge	Solution	Progress in PoC	Next steps
Hospital level Disparate view of the Trusts total capacity and demand	Use data aggregation to visualise current capacity and ML forecasts to predict hospital attendances, admissions, length of stay and discharge in the same User Interface to provide 'helicopter view'.	Developed forecasting of admissions and view of PoC hospital containing 14 wards.	Plug into PAS and Patient flow to get current hospital state and then use forecasting to incorporate other variables to have end to end pathway and overall helicopter view MLOps deployment pipelines for retraining the predictive models.
Ward level Limited ability to see the impact of escalation and reconfiguration decisions	Use scenario modelling to plan, understand, escalate and reconfigure available capacity by assessing impact of changes in hospital set up (e.g. increasing beds, changing bays) and demand (e.g. expected peak because of flu season)	Not included in the scope of the PoC	Use aggregated dataset with patient and hospital information to feed into the combination of forecasting algorithms and models to simulate various factors that impact demand and bed availability (e.g. flu, length of stay, staffing). Develop user-facing feature to enable interaction with the tool and visualisation of impact of inputted metrics and changes (e.g. planned changes in hospital layout or number of staff)
Patient level Highly complex combination of fixed and desirable constraints to be considered for optimal allocation	Develop allocation algorithms to support ADT's overall. - Admission: Support direct admit decisions from ED and all specialties - Discharge: View pending discharges and Reserve patients against chosen beds - Transfers: Consider multiple transfers of patients as a result of one allocation 'bed tetris' component	Developed MCTS and greedy algorithms that allocate patients from ED to a number of specialities.	Expand the tool to support admission to all specialities and to consider discharges and transfers, focussing on greedy allocation first.

04.4 Proof of Concept to Minimum Viable Product

To align with KGH's and ambition to become 'the most digital hospital group in England', we've identified a roadmap to bring the result of this PoC to a Minimum Viable Product (MVP). It's based on the learnings we've collected in this commission and informed by our experience in developing operational tools, from concept to live utilisation in practice. Activities are split across three main areas of focus ('Scope and user engagement', 'Data, IT and modelling', 'Integration'), and progressed in stages, as shown in the diagram below.



04.5 MVP product expansion

The many challenges highlighted above are by no means specific to KGH. They can be expected to apply in varying extents to all trusts in the NHS. Even more so during the current times, when the COVID-19 pandemic adds pressure on healthcare services and increases the complexity of patient management activities.

The capabilities that have been developed in this commission are only partly reliant on specific features of the KGH (e.g. data structure, hospital layout, constraints and historical patient admissions), and could be quickly adapted to suit any other trust in the UK, given the right amount of user engagement, data availability and development time.
The learnings captured in this report, as well as the roadmap to MVP, are equally valuable and applicable to other trusts, and could be utilised as a starting point to assess the level of preparedness for embarking on the development and utlisation of this tool.

In summary, there is a huge potential to scale and productise the work delivered in this PoC, bringing streamlined, simplified and faster processes to every bed manager in the NHS, and better care to every patient.

05 Recommendations

Finally, throughout the delivery of this commission, we have identified a number of technical and operational challenges that would have to be addressed to support the successful delivery of the bed management product vision. Interim solutions were identified to enable the completion of the PoC, but dedicated and more comprehensive activities would be required to solve the root causes of these issues, resulting in long-term benefit to KGH. We have identified two overarching recommendations and below these we have listed specific recommendations with reporting quotes from KGH staff.

Recommendation 1 – *Data collection and existing system infrastructure*

Data used to make allocation and capacity and demand decisions is disparate and scattered across many hospital electronic and paper-based systems. Individuals and multidisciplinary teams currently make bed management decisions at the whole hospital, ward and individual patient level either without access to all available information or after a painful process of attempting to extract it. We have made seven initial recommendations to help solve this issue:

	Recommendations:	**KGH staff quote:**
1	There should be a live view of the hospital status (current and forecasted capacity and demand) to support the site team and multidisciplinary bed management meetings	*" Our biggest challenge is not having a central pulse of Patient Flow information. I still populate a manual spreadsheet to give me an overall Trust position*
2	All wards at KGH should be supported by Patient Flow and all estates information including escalation areas should be available within Medway to support escalation measures and allocation decisions.	*" ITU, discharge and paediatrics are not on Patient Flow so we have missing COVID-19 numbers in the data. We also don't have our full bed base in Patient Flow*
3	All metrics and patient specific information associated with safe admission, transfer and discharge decisions should be digitised to ensure vital information is not missed when allocating	*" We don't a have a count of the number of bed moves built in - this is being worked on and is likely to go into patient flow but this would really help during allocation*
4	KGH ED targets and key flags in the allocation process e.g. consultant review status and prioritisation of side rooms should be easily accessible against patients requiring bed admission, transfer or discharge to enable timely decisions to be made	*" We have numerous data interfaces - I have my front door picture - who is on site and who has started, who is on their way and I have a clock beside this as we have a 1 hour target to get them offloaded*
5	Pending decisions should be able to be lodged against systems e.g. holding a patient against an appropriate bed of a known discharge to avoid multiple patients being allocated to the same bed	*" Sometimes we know the patient is in transit or the room is being cleaned but we don't want to make it look free...it's there but not really there*

6	Live ward staffing information should be easily available to assist with redistribution of staff and to align patients with specialties where possible	*" When we make escalation decisions we need to know staff levels as well*
7	There should be a system that provides an accurate regional position of capacity and demand to enable rerouting decisions to be made	*" We use X app sometimes that gives us a picture on what is happening in the county but some Trusts don't accurately report on it so we use it less now*

Recommendation 2 – *User engagement and adoption*

Any product or solution should be designed around the needs of its users, in this case the KGH site and SMOC team. Not everyone feels comfortable using existing systems and not all members of the team feel confident to make difficult capacity and demand decisions. Before a new product is integrated to support human-in-the loop decisions we recommend the following:

	Recommendations:	**KGH staff quote:**
8	All staff should have adequate training in new systems to increase overall confidence and adoption and ensure that each system is used to accurately record any capacity and demand decisions	*" Not all of the SMOCs are okay with Patient Flow as this is a new thing*
9	KGH staff (particularly less experienced site managers and SMOCs) should have tooling to support the planning phase of short or long term capacity decisions to enable the weighing up of different scenarios	*" Our role is to look at the options on the table and go back to the problem and think about it as a removed view - have you thought about this or considered this. We support the decision making process*
10	KGH should continue to review what is currently working well and what could be improved within complex bed management workflows. We spoke to six members of the highly skilled site and SMOC team. However, other members of the site team and approximately 30 people in the SMOC team should be consulted during a further product discovery phase to ensure that a user centric product solution is designed to fit and support their operational workflows.	*" The site team are a fab team, really, really great and we (SMOCs) are just there to support that escalation route*

Acknowledgements

We want to acknowledge the support, time and immense contribution that the KGH site and SMOC team made to the success of this work so far, and the ACE and NHSX skunkworks team for their technical guidance and oversight.

06 Appendices

6.1 KGH Bed Management Care Path Steps - An Overview

Phase	Steps
Emergency department (ED)	**1.** Arrival: ● Walk-in: Patients that self-present to the emergency department (ED) have their information taken by reception staff and entered into the Patient Administration System (PAS) ● Ambulance: Handover is managed by a member of the nursing team. Staff aim to offload the patient and commence observations within 15 minutes of arrival. **2.** ED Triaging: Patients are streamed into the waiting room or the minors and major sections of ED depending on their clinical presentation. **3.** Initial assessment: Once allocated to an area repeat observations and initial treatments begin. This step can be skipped in favour of a 'see and treat' approach in which patients are immediately seen by senior doctors when staffing allows. After initial assessment the patient is typically discharged, admitted directly to a base ward* or referred to the admitting team for assessment. *The PoC will not be supporting direct admits to a base ward as these beds are allocated directly by the clinical operations team.
Decision to admit (DTA)	If the recommendation is to admit, then the following steps are carried out by the ED KGH site teams: **1.** Decision to admit (DTA) is input into PAS. **2.** The relevant specialist team is notified of the DTA and is given 1 hour to challenge this decision. If they do not attend the admitting team has the right to admit under the new urgent emergency care (UEC) standards. **3.** The site team gathers more specific clinical information from verbal conversations, physical notes and multiple systems to assess each patient's individual needs. The main aim of this process is to determine the particular patient and hospital based constraints that need to be considered before allocation can safely occur. **4.** Most patients admitted through ED fall under the speciality of medicine or surgery and are therefore standardly admitted by the allocator to an assessment unit.
Assessment unit allocation or discharge	There is one surgical and two medicine assessment units at KGH. Patients are usually assessed and treated here before being discharged, referred to an outpatient specialist service or admitted to a base ward bed under a specific specialty. Patients typically stay in the assessment unit for less than 72 hours but irrespective of the decision to admit, all patients are required to be assessed by a specialist consultant before they can leave an assessment unit. If the patient is being referred on for admission then the coordinators of each assessment unit will liaise with base ward staff and the hospital site team to allocate the patient appropriately. **1.** Each patient's individual constraints are considered before they are either allocated to an appropriate empty bed or held against a bed where a patient is

	likely to be discharged soon. If the most suitable bed is one that is already occupied then a decision to transfer an existing patient may occur. **2.** If there are no suitable beds available in a base ward then escalation measures* may take place, for example: a. Flipping a ward bay from male to female or visa versa - patients in an existing single sex bay are moved around the hospital or discharged to enable patients of a particular sex in the assessment unit to move into that bay b. Opening an escalation area e.g. annexe to a ward will occur when all other options have been exhausted *The PoC does not support escalation measures
Base ward stay, possible transfers and discharge	**Base ward stay:** Patients will remain as an inpatient on a baseward until they can be safely discharged. The length of stay of each patient can vary depending on clinical condition and treatment regime. **Base ward transfer:** Patients may be moved to another base ward if their condition improves or deteriorates. The site team tries to keep the total number of moves to three or less during an inpatient stay (not including the move from assessment unit to base ward) as this has been proven to reduce LoS and improve patient outcomes. **Discharge:** Patients are typically moved to the KGH discharge lounge on the day of discharge so that the clinical team can complete the required discharge checklist and outpatient care plan. This ensures that the base ward bed is available as soon as possible for patients being transferred for inpatient care from ED or an assessment unit.

6.2 Comparison of MCTS and RL

Metric	Description	Deep Q RL	MCTS
Overview of algorithm	Brief description of how the algorithm makes decisions	Reinforcement Learning is a branch of machine learning which concerns the way an agent should take actions in an environment in such a way that maximises some reward or minimises some penalty over time. These agents are trained by simulating the agent in the environment, allowing it to explore the way its actions result in reward or penalty in different environment states and gradually learning to take actions which lead to the most reward. Q-learning is a type of RL that is 'model-free' which means that the agent does not need and does not try to learn a model of the environment (which means trying to predict state transitions and rewards/penalties).	A search algorithm that tries to find the most promising next decision in a sequential decision problem. Does this by searching the tree of possible decision sequences by randomly sampling from it in a way which balances exploring the tree with exploiting promising directions already discovered within it. Useful when the search space is too big to exhaustively search (so useful for chess, not useful for tic-tac-toe)
Input data	What input data is required for an algorithm to make decisions. What is the required structure of the data?	Same size input for every single patient, potentially leads to lots of zeros in empty beds.	
Ease of training	Does the algorithm need to be trained? How easy is it to train? How confident are we that the algorithm will learn?	Catastrophic forgetting? Harder to train	No training required. The tree search will need to be run in each new state to provide a fresh set of recommendations.
Training time	How long does it take to train?	Long time to train, expensive?	N/A
Expected performance	Do we expect to find the optimal move with this algorithm every time?	Could outperform, but dependent on training time, might not be robust to rare situations	No guarantee the recommendations will be optimal, however the longer the search is allowed to continue, the more optimal the recommendations will become. Performance would be limited by our ability to 'look ahead'. Can build in warning if score is too high for suggested move

Explainability	How easy is it to explain the decision? What level of explanation can we get from this algorithm?	Could combine with LIME to explain in terms of input features (descoped)	Explainability would be in terms of the constraints that are broken + explainability of whatever model we are using to forecast admissions
Time to prediction	Time to prediction	Prediction would be very fast	Better solutions will be found if allowed to search for longer, but can theoretically work on shorter timescales, there are also strategies that can be adopted to reduce the search space.
Customisability	Does the algorithm allow the layout of the hospital to be changed? Can a ward be opened or closed easily?	Have a flag for whether the bed is open or closed. Can't just add a new ward. Can't change penalties without retraining.	This can be easily accommodated.
Robustness	Will the algorithm be able to handle rare situations?	Less likely to be able to handle rare situations as would have to remember it had seen it before in training	This should be fine since the search algo runs fresh in each situation. The rarity of a situation won't affect the search.
Computational feasibility	What compute power will be required to make suggestions?	smaller than MCTS	Expensive, but if running on platform perhaps not important
Provision of options	Will the algorithm be able to provide options for different bed allocation strategies, or just one	Output is probability of which action to take, so can provide multiple options	Yes, once the algorithm is run it can 'score' each possible decision by how promising it is. Usually the most promising one is chosen, but in principle the top N most promising could be surfaced to a user for example.