A Web Application for Daily Staff Coordination in Hospital Care

Hampus Adolfsson
Mattias Adolfsson
Daniel Wang
Abstract

A Web Application for Daily Staff Coordination in Hospital Care

Hampus Adolfsson
Mattias Adolfsson
Daniel Wang

Stress and heavy workloads are commonplace for those who work within Swedish healthcare; the issue is exacerbated further by the current shortage of qualified personnel. Therein arises a need for tools to help lessen the burden on the personnel. This report presents a prototype of a web application made for daily staff scheduling, tailored to the needs of the pediatric surgery department at Uppsala University Hospital. The aim of the project was to deliver a more flexible digital solution of daily staff coordination at the department. Two fundamentals of the application were that it should be easy to use and display information clearly with few misunderstandings. The resulting prototype met the usability requirement, other than a few specific usability issues, and improved upon the readability of the contents displayed in comparison to the previous system. To further improve the system, work could be done to increase the efficiency with which tasks can be performed.
Sammanfattning

Contents

1 Introduction 1

2 Background 2

2.1 Stakeholder 3
2.2 The need for flexibility in healthcare 3
2.3 Psychosocial strain among healthcare workers 3
2.4 Current system in place 4
2.5 Similar systems at other hospitals 5
  2.5.1 Medical student in clinical practice at Umeå University Hospital 5
  2.5.2 Assistant nurse at Tierp Health Center 6
  2.5.3 Assistant Nurse at Uppsala University Hospital 6
2.6 Web application 7
  2.6.1 Server-Client Model 7

3 Purpose, aims, and motivation 8

3.1 The importance of intuitive design 9
3.2 Extensible design 9
3.3 Delimitations 10
3.4 Core application features 10
3.5 Ethical concerns 11
3.6 Sustainable development 12

4 Related work 12

4.1 Scheduling and work force management applications 13
4.2 Related means of increasing department effectiveness 13
Due to differences between countries’ healthcare systems, there are not always unambiguous translations with respect to healthcare professions. As such, provided in Table 1 are the Swedish to English translations of healthcare professions used in this report.

<table>
<thead>
<tr>
<th>Swedish</th>
<th>English title used</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Läkare</td>
<td>Doctor</td>
<td><em>Medical Doctor</em> is also used</td>
</tr>
<tr>
<td>Sjuksköterska</td>
<td>Nurse</td>
<td>-</td>
</tr>
<tr>
<td>Undersköterska</td>
<td>Assistant Nurse</td>
<td>-</td>
</tr>
<tr>
<td>Barnmorska</td>
<td>Midwife</td>
<td>Also known as <em>accoucheur</em></td>
</tr>
</tbody>
</table>

Table 1 The Swedish professions that and the English titles used to refer to them.
1 Introduction

With an increasing workload put on the Swedish healthcare system, the demand for more personnel is an ever growing issue [Soc19]. Moreover, stress and heavy workloads are among the most severe psychosocial work environment risk factors in Swedish healthcare [FB13]. Thereby arises a need for tools that can help lessen the burden put on healthcare personnel. A prototype for such a tool was developed in this project in the form of a web application, designed to facilitate the daily staff coordination within a hospital department. Its main purpose is to display a schedule designating to where and during what time staff members are assigned, as well as enabling them to find specific people (for example an anesthesia specialist) to see when they are available. As it is only a prototype, it lacks functionality in the back end, namely saving changes to the schedule to a shared database.

The need for the application came specifically from the department of pediatric surgery at Uppsala University Hospital. The head of the department, Tracey Koo Clavensjö, has several years of experience organizing and coordinating personnel within hospital departments. She was the one who initially expressed the need for a better method of daily organization. Before the project, a physical whiteboard with a grid of tape was used to display the daily schedule. This was used to assign staff to specific rooms and tasks at specific times, for example surgery in room 1 from 12 to 14 pm. It was also, among else, used to keep track of who had called in sick or was at home with a sick child. The application described in this report is aimed at replacing that system, by providing a digital schedule that is quick to use and learn, tailored to the needs of the department, and that is accessible from anywhere, both on mobile and desktop devices.

The application was developed with a focus on being user-friendly and intuitive, meaning that it should be easy to use and provide its information clearly and with few misunderstandings. An intuitive application can be seen as being familiar to the user [Ras94]; to make the application seem familiar, its interface was largely based on the whiteboard that it aimed to replace.

The development of the interface was based mostly on a survey we created as part of this project, asking employees at the department about the previous system and the system we were developing. The survey showed that what employees disliked most about the whiteboard system was its tendency to be messy and difficult to read. It also showed that employees regularly experience problems as a result of changes made to the schedule not being communicated properly. Despite this, the employees expressed a moderate satisfaction with the whiteboard system. The information gathered from the survey was used to make decisions about what functionality was important for the application, as well as what qualities to focus on when developing the interface.
The resulting application is a cross platform web application that can provide information about daily schedules and other employees to staff at a hospital department. During evaluation of the application, we found that almost all basic tasks could be completed with only minor issues, indicating that the application is easy to learn. Additionally, the application improves on the previous scheduling solution with respect to readability. The whiteboard previously used as a schedule had handwritten notes using a whiteboard pen, which were hard to read and left traces and specks if not erased properly.

The evaluation raised some concerns about specific usability issues that were recurrent with all testers. One specific example was a popup showing when the user tried to exit without saving changes; the popup seemed confusing to the users, indicating that the fact that changes were pending, and not immediately saved, was not conveyed properly. There were, however, no large overarching usability issues found, beyond the few specific examples.

We managed to develop an application that meets the specific needs of the pediatric surgery department, and that mitigates some of the major issues with the previous analog system. If a functional back end is developed to support the application, providing the ability to save the schedule to a shared database, it could be of real value to both administrative and non-administrative personnel. It also serves as a base upon which many future extensions could be built.

2 Background

Hospitals and other healthcare facilities are a fundamental part of modern society. Generally, healthcare operations are incredibly complex and have a multitude of components that all need to align in order to function. An operation of such complexity comes with a host of factors all impacting the objective of providing the right care for the right patient at the right time.

In order to develop a system that improves upon the different aspects of the work taking place in hospitals or other healthcare facilities, it is imperative to investigate factors affecting work within healthcare beyond what can be noticed from the perspective of patients. One such factor is the mental health of the personnel working in healthcare. The mental health of nurses, doctors and other personnel working in healthcare is a factor with an impact of significant magnitude affecting not only the well being of the personnel, but consequently also the patients. Information that laid the foundation to the project will be discussed further below along with a presentation of the project’s stakeholder.
2 Background

2.1 Stakeholder

The stakeholder in this project is Tracey Koo Clavensjö, head of the pediatric surgery department at Uppsala University Hospital and also trained ICU (Intensive Care Unit) nurse. She was also head of the burn wounds intensive care department before moving on to pediatric surgery. With several years of experience as the person organizing the personnel at these departments and also her experience as an ICU nurse, we believed she would have very viable input and feedback on the application.

During the development Clavensjö has provided us with requirements of the application and given us important feedback. Additionally she has also provided detailed information in regards to how the pediatric surgery department and others alike, operate.

2.2 The need for flexibility in healthcare

Generally within healthcare there exist numerous specializations that all serve to treat specific conditions exhibited by the human body. Some departments in for instance a hospital require personnel with specific skill sets to treat patients with a specific set of conditions. One such department is the pediatric surgery department. As described by our stakeholder Clavensjö, staff members’ individual schedules can vary drastically between different days. A specific person might be required at a specific location due to their ability to treat a patient’s certain condition or simply because there needs to be an attending staff member to quickly respond should a patients status change.

The task of coordinating staff in a volatile environment in such a way that patients are given the appropriate care at the right time, is given to the head of the department.

2.3 Psychosocial strain among healthcare workers

In 2012 a study was conducted by the Swedish work environment authority as part of a multinational collaboration between countries that are members of The Committee of Senior Labour Inspectors (SLIC) of the European Commission [FB13]. This multinational campaign aimed to investigate psychosocial factors’ effect on personnel within healthcare. A total of 408 inspections were conducted on 218 different workplaces. In 173 inspections psychosocial inadequacies were found. These inadequacies include stress and strenuous relations to colleagues or patients. The study concludes that the investigated psychosocial factors have a definitive negative effect on the personnel, an issue which is of sufficient severity to call for reforms and new demands on work envi-
2 Background

Environments within the healthcare sector.

The negative effects of poor working conditions extend beyond the personnel, as identified by a report published by the U.S. Department of Health and Human Services [HSF+03]. The report is a meta-analysis carried out by the Oregon Health and Science University using articles collected from five different bibliographic databases discussing patient safety and human performance in relation to working conditions. Working conditions were divided into the five categories workflow design, workforce staffing, personal/social issues, physical environment, and organizational factors.

Results of the study showed evidence linking rates of medical errors and incidence of patient outcomes related to patient safety to the five aforementioned working condition factors. Possible patient outcomes were, among others, infections acquired in hospitals, pressure ulcers, and patient falls.

The report concludes that an investment in improving these working conditions is required to improve patient safety. Areas most likely to see a successful improvement in patient safety as a result of such an effort in improving said working conditions include:

- Avoidance of distractions and interruptions.
- Improvements on the exchange of information between staff and other hospitals.
- Giving high-risk and complex procedures to physicians who conduct those frequently.
- Changes in the staffing of nurses.

2.4 Current system in place

Currently the pediatric surgery department at Uppsala University Hospital coordinates their staff through daily meetings with the help of a whiteboard, a sketch of which can be seen in Figure 1.

All information is organized on the whiteboard using a grid of tape which creates different cells. There is a magnet with each respective staff member’s name on it. The magnets are then placed in cells on the whiteboard to signify what person has what role in which operating room, and sometimes additional notes are added with a whiteboard pen. The employees are divided into two work groups, surgery (OP) and anesthesia (AN), and each work group has its own cell for each room. Other important information on the whiteboard includes lunch and coffee break times, absent staff members, and information about other departments with which they cooperate.
2 Background

Figure 1 A sketch of a part of the whiteboard schedule in the department

2.5 Similar systems at other hospitals

The pediatric surgery department at Uppsala University Hospital is not the only hospital nor department of, that requires staff coordination. We have held interviews with three medical students at different stages of their education and who thus have held different professions or roles within healthcare. In each interview the medical students laid out a description of the systems for staff coordination at their respective workplaces. The people interviewed have chosen to remain anonymous and will therefore not be named in the respective interviews written below.

2.5.1 Medical student in clinical practice at Umeå University Hospital

In an interview with a medical student studying their fifth semester who also works as a care assistant [WA19a], the medical student explained that during their practice at the medicinal intensive care unit department (Swedish: Medicinsk akutvårdsavdelning) of Umeå University Hospital, the daily staff coordination was for the majority of the work week done verbally. The responsible staff member for the organization of staff would in conversation convey each employee’s respective schedule.

The interviewed medical student explains that during periods of higher than average workload where the number of patients were higher than average or patients had much more complex conditions, the need for organization was notably more important. An increasing demand for organization arose from the fact that an employee might be required to perform several different procedures on more than one patient. Thus the employee would be required to be present at different locations during their work shift. During
these times each employee’s schedule was sent to them as a PDF file via e-mail.

According to the student, periods of higher workload were often anticipated and prepared for. The student noted that if such a situation was to occur without forewarning, it would be difficult to properly coordinate the staff.

2.5.2 Assistant nurse at Tierp Health Center

Another interviewed medical student was studying their sixth semester and works as an assistant nurse at Tierp Health Center [WA19b]. The entire department’s schedule is available as a PDF file through their booking system. The schedule includes all patient appointments, their respective personal information, and reason for their appointment. However the schedule does not include any information in regards to what and where different staff members need to do and be.

In practice the schedule is shared between the employees and treated more as list of tasks that needed to be completed. The administrative staff member prints out a copy of the schedule on paper and through verbal communication employees assigns themselves to a task. When an employee has completed a task it is marked as such on the list. The main reason for printing out the schedule was because the computers at the department had to be used for other administrative tasks.

The overall opinion of system amongst the employees was that it was simple and suited their needs. There were a few minor complaints, mainly concerning miscommunications and difficulty in making changes that would change the staff’s workflow. Sometimes employees would forget to mark a task as finished, leading to other employees taking on the already finished task. The medical student noted that while these flaws are unlikely to have serious implications, they do waste valuable time that could be spent productively. Another concern raised was the possibility to leak sensitive patient information; the schedule was according to the interviewee sometimes handled carelessly and in reach of people who should not have access to it.

2.5.3 Assistant Nurse at Uppsala University Hospital

The third interviewee is studying their eleventh semester and has worked as an assistant nurse at Uppsala University Hospital’s maternity ward[WA19c]. The maternity ward utilized a whiteboard for their daily staff coordination. It was almost identical to the system used at the pediatric surgery department described in section 2.4. The two major additions were inclusion of patient data and additional work groups beyond the two used in the pediatric surgery department (anesthesia and surgery). The maternity
ward’s whiteboard also showed the patient admitted in each room along with related information to said patient. Instead of two columns for surgery and anesthesia staff, the maternity ward had columns for assistant nurses and midwives.

The medical student explained that the system met the requirements for their work. None of its flaws has had any serious consequences, but sometimes caused inconvenience. Information on the whiteboard was mainly written using whiteboard markers and tended to not look structured according to the interviewee, to the point that information could go unnoticed. Another drawback was the limitation of a single whiteboard which only made it possible to view one day at a time.

One potential problem with the system raised by the interviewee was the risk of leaking patient data. The whiteboard which displayed confidential patient information was accessible to anyone and could potentially be viewed or altered by an unauthorized person. While this had not yet been an issue at the department the interviewee emphasized the relatively high possibility of it.

2.6 Web application

One way to deliver a digital application to the end user is through a web application: an application served through a web page. These run in web browsers — for example Google Chrome — which can be found in a multitude of systems across several platforms. Because this wide availability of software capable of running them, web applications can run on many different platforms with minimal platform-specific code.

2.6.1 Server-Client Model

The server-client model is a commonly used structure for network applications. The server-client model generally has one participating member in the system defined as the server, which is the provider of the service or other resources. Other members called clients make requests for resources from the server. For example, a server can provide a web page by responding with HTML, CSS and JS to clients making requests for it.

In order for the clients to acquire services and/or resources they have to establish communication with the server. Servers get incoming requests and once such a request has been responded to, a session is established and data can be exchanged. Sometime this session might just be the server sending some data (for example a file) after which the exchange is done. In more complex applications, however, this could be a logged in user with certain preferences and administrative privileges.
3 Purpose, aims, and motivation

The purpose of this project is to provide an improved way of day-to-day organization within a hospital department. The aim is to provide this in the form of a digital application that can, among else:

- **Display** the staff schedule (for example: show their assigned room).
- Enable administrative personnel to **assign** tasks and rooms.
- **Find** specific people and when they are available (for example: an anesthesia specialist).

This stems from a current need for such an application in the pediatric surgery department at Uppsala University Hospital. As was discussed in Section 2.4, today a physical whiteboard is used to organize, a sketch of which can be seen in an earlier section in Figure 1.

According to the head of the pediatric surgery department, our stakeholder, their current solution is limiting and requires a significant amount of time and effort to use and maintain. With the whiteboard system, it is hard for everyone to keep track of each other and where they are supposed to be without repeatedly physically walking to this whiteboard. Moreover, making a change to the whiteboard schedule would require a person to physically erase the notes, move the magnets and write down new notes. Additionally, the change has to be manually communicated to everyone. According to Clavensjö this usually means directly calling the employee(s) that are affected by the change.

With only two whiteboards at their disposal only two days can be planned for at the same time. During times where the influx of patients is larger than normal it is important to be able to plan further ahead due to the larger constraint put on their resources. Currently, should planning exceed two days be required, the schedules past two days have to be written down on paper.

A survey we conducted with the personnel of the department pointed to some additional problems with the whiteboard, pertaining chiefly to its readability and to difficulties in making sure the personnel stays up to date with the information on it. The results of the questionnaire are described in detail in Section 8, and the entire questionnaire can be found in Appendix A.

The aim is to reduce these problems, meaning that the application should be able to effectively deliver the schedule to employees wherever they are, in a way that is clear and easily readable.
The benefit of the application can be generalized to almost all hospital departments. All Swedish county councils (Swedish: landsting) report a shortage of qualified personnel within multiple health care professions [Soc19]. In addition, a report from the Swedish Work Environment Authority indicates that stress and heavy workloads are among the most severe psychosocial work environment risks for health care professionals [FB13]. This highlights the need for reducing time spent coordinating and organizing within hospitals, thus achieving more time efficient care. That is the goal of this application.

Although there are general applications available today for organizing tasks and otherwise work (e.g. calendars, work force management apps), a more specialized hospital-focused application can provide functionality especially made for healthcare operations. For example, the application could provide an overview of the schedule for the day, showing all the rooms of that specific department and their respective assignees. Additionally the application could provide a searching function for staff, finding pager-numbers and sorting on categories based on hospital-specific professions.

To increase the flexibility and effectiveness of the application, it should work on smartphones and desktop computers. This enables it to be used on-the-go, wherever the user might be at that time, which helps solve the current problem with the schedule information only being accessibly in one physical location. The hospital department uses a TV-screen to display other scheduling information, and the same screen could be used to display this schedule as well. The department does not use tablets, and thus it is not necessary to develop the system for those devices.

### 3.1 The importance of intuitive design

The focus of the application is to be user-friendly and intuitive. In other words, it should be easy to quickly understand and use it, as well as leaving limited room for misunderstandings. The reason is that the intended users (hospital staff) may be in a high-stress environment, where speed and correctness are important factors. We can also not assume that the intended users have good technical skills. Moreover, because the main goal of the project is to provide a more efficient coordination method, we will naturally focus on providing an intuitive interface without unnecessary distractions. This means that a big part of the project is designing the interface to meet these goals.

### 3.2 Extensible design

The overarching purpose is not just to create an application for a specific department, but to enable better organizing within (almost) any hospital department. The application
should be extensible and be able to be customized for a specific department’s needs. This includes being able to add their respective rooms and locations, their staff, and their relevant professions.

3.3 Delimitations

Several delimitations were made to limit the scope of the project. Most significantly, the system is not a fully functional application, but rather a fully functional user interface and client-side code base that works with mock data from a bare-bones server. Developing the server portion of the application is a time-consuming and sufficiently separate task that it was deemed not to be worth doing for this project. Instead, we opted for creating a mock server, to be able to display the principles of how our client-side code might communicate with a server without the cost of developing that server fully. As a consequence of this delimitation, this report focuses mostly on the client side code and user interface, with some information about the mock server included where relevant.

The ideal application would include an account based communication system with varying administration rights. Within the communication system the user would have access to voice calls and text messages, through which the users could for example request changes to the schedule and the administrating personnel would be able to respond to such a request. A communication system would also imply functionality of notifications for messages, or could also be used for notifying of schedule changes. Another useful but complex functionality would be the integration of patient data. However, this functionality would come with a host of difficulties and drastically increase the complexity, as things like real-time communication, native device features and handling of sensitive data would have to be considered.

3.4 Core application features

To fulfill the purpose set out for this application, certain core functionalities are mandatory for implementation. There are two main features of the application.

The first feature is the ability to view and edit the schedule for the whole department and each employee’s individual schedule. An employee can be scheduled to be present at a certain location from and to a specific time. They can also be scheduled to have an overarching responsibility, such as nurse in charge (as seen in Figure 1, where nurse is abbreviated by SSK).

The second core piece of functionality of the application is a page dedicated to dis-
playing information about each employee. All employees at the department are listed after their profession and through interaction additional information may be acquired. The user should be able to find a specific employee, either by name or profession, and display their information. Such information includes, profession, mobile phone number and the employee’s personal schedule. The schedule in particular can be useful to find where a specific person is and/or when they are available.

As with the schedules, the list of employees can also be edited. All information related to the employee can be changed and new employees can be added or removed from the list at the behest of administrative personnel.

### 3.5 Ethical concerns

As discussed in Section 3.3, our application only handles mock data. However, the next logical step of making it ready for real-world use would be to implement a database-backed server with real data. As such, it is relevant even to this project to consider the ethical concerns of that. Our application does not contain any patient data, both for legal and project complexity reasons as discussed in Section 3.3. However, should a back end with real data be implemented, employee information would be prevalent throughout. Examples of such delicate data includes scheduling, absence (sick leave, care of sick child), and contact information. This should be protected from unauthorized access, so as to not expose it to third parties without the employees’ consent. For example, an authorization protocol could be implemented, limiting the access to the scheduling data to those with administration privileges.

Another ethical concern with a back end with real data is the possibility of saving a large history of scheduling and absence data. While the scheduling data today is available to all employees just by observing the whiteboard, providing it in a digital format lends itself much more to data collection and data analysis. This could lead to a situation where the data is used to for example analyze patterns in the scheduling or employee behaviour. However, it is hard to control access to the scheduling data, since the application needs it to display it. One possible measure could be to use a login system, where employees must have an account to see the schedule. This in turn opens possibilities to log and analyze each employees’ schedule access patterns, which possibly is not desirable from an ethical standpoint. A complementary solution would be to not allow access to scheduling data beyond a certain time period, and instead only provide anonymized and very general statistics of the time period.

Another concern of the current application is the lack of support for unconventional usage methods, such as with a screen reader or without a mouse. This is typically not a
problem; any employee working at the department can be expected to have a sufficiently functional body to be able to use the application with a touchscreen or a keyboard and mouse, as they would otherwise be unable to perform basic healthcare tasks. Some aspects of accessibility are relevant however, such as the ability to use the application with reasonably poor eyesight or with limited experience with technology. Because we use red, green and blue to color code for the different professions within the department (exactly like the hospital does, internally), people with red-green color blindness could have a difficult time differentiating between doctors and assistant nurses. This could be helped by explicitly writing out employees’ profession where it is important.

With the schedule being available anywhere and any time, there is a possibility that the employees could feel pressured into keeping up with the schedule even when they are not at work, thus doing unpaid work outside work hours. For instance, the management could argue that employees need to be aware ahead of time if a staff shortage is likely to occur; they could also suggest that when employees come into work they need to know what the day’s schedule looks like.

3.6 Sustainable development

Good health and well-being is imperative to allow us to reach our full potential as human beings and as a society. In turn, this requires a well-functioning healthcare system. However, as was established in Section 2.3, there are significant psychosocial inadequacies in today’s healthcare scene, leading to negative effects for both personnel and patients. To rectify part of these problems, our application can be used to improve daily staff coordination. This would serve to ease the strain on staff and subsequently improve the healthcare. Improving healthcare quality is an important step in ensuring the third goal as set out by the United Nations. It states that we should strive to “Ensure healthy lives and promote well-being for all at all ages” [Uni15]. Naturally, an improvement to healthcare — which our application should be able to provide — will help to ensure healthy lives and promote well-being.

4 Related work

The are two main types of related work: digital applications similar to the one we developed, and academic research into how to achieve an effective and well-functioning healthcare. The digital applications can be of interest to draw inspiration in architectural decision, as well as in interface design. Conversely, the research into effective healthcare can provide valuable information about which features of the application are

12
important.

4.1 Scheduling and work force management applications

The application developed in this project is partly a scheduling and work force manage-
ment application. Several of these type of applications exist today, although many are
mostly focused on time-reporting and subsequent salary calculations (for hourly-paid
employees). One example of this is *Timeplan* [Exs19], which is an application devel-
oped by Exsens AB focused on time-reporting. It includes functionality for scheduling
and is a multi-platform application (just like ours). It has functionality for calculating
cost of staff, making sure schedules follow local regulations and employment contracts.
It is, however, mostly made for the management to keep track of employees, not for
employees themselves to coordinate with others. This is where our application could
prove useful.

An example of a work force management application in a more academic context is a
Maros’ and Norberg’s *Employee administration* system [MN05], which was developed
as part of their Bachelor’s thesis at Ingenjörshögskolan in Jönköping. This is a system
that provides delegation of short-term assignments to employees based on incoming re-
qusts from clients. It also provides a way to report progress and time spent working.
The work is almost 15 years old, which means that hardware such as phones and soft-
ware frameworks have changed since then. However, it is still interesting to note that
is similar to our work. This is both in the sense that it focuses on providing employees
with tasks and schedules, and that it is a web-based multi platform system. Even 15
years ago, web-based systems were used for their multi platform capabilities. It also
shows that work force management applications are nothing new, although it seems to
be lacking in use within the pediatric surgery department at Uppsala University Hospi-
tal.

4.2 Related means of increasing department effectiveness

There already exists some research on increasing the effectiveness of hospital depart-
ments, a significant portion of which relates to scheduling and organization. However,
existing research tends to focus on scheduling algorithms and ways to plan for more ef-
efective utilization of resources. On the other hand, the work presented in this report was
concerned with how a given schedule and organizational information can be best com-
municated to the personnel, regardless of how said schedule was devised. The articles
most relevant to us in this field were:
Software Tool for Operating Room Scheduling in a Spanish Hospital Department by Clavel et al. [CBMA18] presents a software tool for aiding in scheduling the surgical department of a Spanish hospital. The main focus of the paper is an algorithm for scheduling surgical procedures in a way that gives an efficient use of resources, but it also emphasizes the usability of the tool, and its interface was developed with feedback from the doctors meant to use it. The authors identify the need to make the interface clear, efficient and easy to use, but did not evaluate its usability. Unlike our application, this tool includes patient data, so that specific patients and procedures are scheduled, rather than individual staff members. The schedule for each operating room is presented as a list of procedures scheduled for each day, along with their start time and expected end time.

Att mäta och planera för en välfungerande sjukvård (english: To measure and plan for a well-functioning healthcare) [PE14] is a report by The Swedish Association of Local Authorities and Regions (SKL). In the report, the authors discuss the importance of and challenges faced in organizing Swedish health care. Several ways of improving efficiency are proposed centered in part around predicting resource demands and planning to handle those demands, and in part around ways to follow up and evaluate the results of such planning. The relevance of the report comes mostly from the fact that it concerns itself specifically with the Swedish healthcare system, which is where the application in this report is targeted. The report is also important for establishing the value in and relevance of developing effective organization within Swedish hospitals.

5 Method

The following section describes the methods used to develop the application. We outline the ways in which we collected user data, and then describe the process by which we developed an interface from that user data. We chose to implement the interface in a web application, and the reasons for and specifics of this choice are also described.

5.1 Obtaining use case information

To be able to make informed decisions with respect to application design, we needed opinions from the intended users. This included opinions on both the previous system as well as requirements and potential use cases of the new system. To that end, we sent out a questionnaire. This method was chosen as a complement to one-to-one interviews as interviews are generally more time consuming and we could not guarantee to be able schedule enough interviews to acquire good insight of the subjects we are investigating.
Questionnaires in contrast to interviews are very likely to reach more people with little time investment from either side. To improve the quality of the collected data as well as the response rate, the questionnaire was created according to the principles outlined by Dawson [Daw15, p. 29]. Questions are categorized as either close-ended, having a distinct set of possible answers, or open-ended, allowing free-form answers.

According to Dawson, questionnaires should start out with closed, general, and factual questions, and then progress into more open, specific, and abstract questions. Section 8 describes how we designed the questionnaire questions according to this principle.

### 5.2 Conducting interviews

To learn more about systems used at other hospitals or departments for staff coordination (see Section 2.5), interviews were conducted. When conducting interviews it is important to obtain data with a high density of details with a low risk of being misinterpreted by us later. The methods used were based around Dawson [Daw15, p. 26-27].

#### 5.2.1 Selecting interviewees

According to Dawson the selection of interviewees has an important role in obtaining useful data in interviews. In cases where the ideal participants may not be available, the interviewer needs to consider alternatives that might provide the same information [Daw15, p. 28].

In our case the ideal candidates would be those working full time within healthcare at another hospital or department. We were not able schedule interviews with people working full time and instead sought medical students. The selected medical students had all either worked as assistant nurses or completed their clinical practice. While the candidates may not have as much experience with their respective systems, they would still be able to provide information of how the system functioned and how it was used in practice.

#### 5.2.2 Types of questions to ask

Depending on the questions asked in an interview Dawson generally places interviews in two different categories, Structured and Unstructured. Structured interviews are conducted with a series of pre-made questions, while unstructured interviews are centered
around open-ended questions and are in comparison less formal. The purpose of an unstructured interview is to give the interviewee room to provide extended answers without the constraints that pre-made questions might impose [Daw15, p. 27]. Since the objective of the interview was to acquire information about a staff coordination system that we did not have any knowledge of, we decided to conduct an unstructured interview with open ended questions. Having the interviewee tell us about their system was decided to be more useful than to ask specifics about the system, since we would not know what to ask about.

5.3 The application design process used

When designing the application, by and large the process used was based around the three phases described by Arvola in Interaktionsdesign och UX [Arv14, p. 7-8]:

- **Conceptual design** (Swedish: konceptfas). Examine what needs to be done and why. This includes interviewing the potential users and understanding how the application will be used.

- **Physical phase** (Swedish: bearbetningsfas). Make rough sketches of UI elements and preliminary layouts. Focus on how elements should be organized with respect to each other. Determine main features of the application.

- **Detail phase** (Swedish: detaljeringsfas). Finalize details of the application. This includes both UI aspects (e.g. color, position, size) as well as feature aspects (e.g. what to include in the final product).

This process initially tries to answer what needs to be done by understanding the problem and the situation in which the application will be used. This means analyzing the obtained user data in ways to make it more manageable, such as creating personas. We obtained this information by interviewing the stakeholder (Tracey Koo Clavensjö) about the current situation in the department, as well as from the questionnaire sent out to the department personnel. Additionally, one of us has experience working at the hospital in a similar department, which provided valuable first-hand knowledge of the day-to-day activities. The design process described by Arvola provided a structured and systematic way of designing a usable interface in a way that is based in user data. There exists other works providing similar processes, such as by Nielsen [Nie94b], Mayhew [May99] or Allen & Chudley [AC12]. Arvola’s process was chosen in part because all team members had previous experience with it, but also for its recency and because it is based well on research and scientific methodology.
5.4 Designing the application to be user-friendly and intuitive

Since the application had the specified goal (in Section 3) of being user-friendly and intuitive, the physical phase, where much of the foundation for the UI design is laid, was especially important to the application. An intuitive UI can be seen as being familiar to the user [Ras94]. In other words it should (in part) conform to the user’s expectations and previous experiences with similar interfaces. With this in mind, we used concepts and visual metaphors mimicking the system we were replacing. In addition, all team members worked collaboratively on the UI elements, since this assured a broader perspective on whether the produced design lived up to being familiar and user-friendly.

Initially, a sitemap was created, which is a map over pages within the application and links between them [Arv14, p. 110]. This was used to define which pages were needed and roughly how they related to each other. Then, these pages were designed with two aspects in mind: making it easy to act and making it easy to understand the effect of the action. This was done using Arvola’s four principles [Arv14, p. 125-126]:

- Make sure elements are visible.
- Have a good conceptual model (e.g. a paper for writing on).
- Have a good mapping between act and response (e.g. pressing a downwards facing arrow shouldn’t scroll the view to the right).
- Make sure the feedback on actions is clear and visible.

All team members collaboratively evaluated proposed design choices with these principles in mind.

5.5 Techniques when finalizing interface details

Two important techniques used from Arvola’s details phase were:

1. To keep an overarching and consistent structure
2. Emphasize the most important elements with the visual hierarchy

The first stipulates that it is beneficial to have some sort of general structure to the design, for example a simple grid [Arv14, p. 143]. This enables the user to quickly scan
the page for their goal, speeding up and facilitating interaction. Moreover, keeping the same structure across multiple pages allows the user to recognize the pattern, yielding an easier to understand interface for the user. This was an important technique in making the interface feel familiar to the user, which (as described in Section 5.4) is a step in making it intuitive.

The second technique suggests that it is important to keep in mind the visual hierarchy of the page when trying to accentuate elements [Arv14, p. 144]. According to Arvola, elements stand out due to three main factors: being higher up in the page, being larger, and having high contrast. For example, should an important button be clearly visible to the user, it could be placed high up in the middle of the page, be large relative to other elements, as well being of a unique color compared to its surroundings.

5.6 Implementing the system as a web application

One of the main advantages of developing a web application, as opposed to a native application, is that the same code can be run on almost any device as long as the device is capable of running a web browser. With a web application, there is also no need for users to download or install any software in advance. A significant drawback, however, is that a web application typically cannot access specific features of the underlying operating system of a device, such as sending push notifications or accessing hardware sensors [Was10]. We chose to implement the application as a web application, most importantly to allow us to target a wider range of devices with less development time. Moreover, the main functionality of the application requires no native device features, and as such it was not necessary to develop a native application.

It is worth noting that features such as notifications and voice calls, which may be implemented in the future (see Section 3.3 and Section 15), require access to native device features. There are ways to attain access to such features without developing fully native applications for each supported platform. Specifically, an existing web application may be run within a native application with an API for native features [CL11], or the application can be developed within a framework that is capable of outputting native code for several different platforms [Was10].

5.6.1 Front end technologies

To develop the user interface of the application, the Typescript framework Angular was used. This framework provides extensive support for building a single page application (SPA). A SPA is a web application only loaded once, after which the web page’s content
is modified dynamically — for example when the user presses a button — without having to load a new web page [Ang19]. This is advantageous for heavily interactive pages, for example to allow rapid changing of day for which to display the schedule. Without a SPA, the page would have to reload frequently, which could lead to a disruptive user experience. We also used the library Angular Material, which provides a set of basic interface components (checkboxes, buttons etc.) with a unified and globally configurable style.

There are other libraries available for developing SPAs, such as Vue.js or React. Angular was chosen because we wanted a framework using Typescript, to be able to use features like static typing, enumerations and interfaces.

5.6.2 Back end technologies

To handle the data of the application, the C# framework .NET Core was used. This was chosen for two main reasons. Firstly, we had significant previous experience with the language, leading to quicker development. Secondly, and more importantly, the framework has good support for creating REST APIs, which we intended to use to communicate between the data layer (back end) and interface layer (front end). An API is an interface provided by a program, containing methods for retrieving and modifying data of the program. It is a specification of a set of methods a program (our back end) provides to another program (our front end) seeking to communicate with it. In the case of our back end, this includes endpoints for retrieving a list of employees and their assignments for the day.

The benefit of creating and using an API is to decouple the data layer from the interface layer. The API clearly specifies how data in the back end can be retrieved and modified, regardless of the platform and language of the client using the data. This facilitates making changes to the front end in the future, should we for example want to develop a native Android or iOS app.

The main alternative to a RESTful API is a SOAP-based API. According to Ruby and Richardson [RR07, p. 300], REST APIs are more suitable for resource-oriented web applications, whereas SOAP APIs are better suited for distributed process-oriented applications. REST APIs are also significantly faster, owing to the high verbosity of SOAP messages and the time it takes to process them [BSMG12]. This makes REST preferable in cases when bandwidth is limited, such as on a slow wifi or 3G connection. Because all data in the application (employees, assignments) can be modeled as resources, and to ensure that the application is responsive even in parts of the hospital with poor wifi signal, we opted for a REST API.
6 System structure

In broad terms, the system consists of two components: the front end and the back end. The back end is a long-running server constituting the data layer of the system, providing employee and scheduling data. The front end is the presentation layer, and uses the data and of the back end to present users with a user-friendly interface. This terminology is related to the client-server model described in Section 2.6.1; the back end acts as a server providing services to clients, while the front end software runs in each client to deliver the end-user experience.

A representation of the system can be seen in Figure 2. The user(s) interact with the interface provided by the front end, on their device of choice. For example, a user could be viewing the schedule for the day. To be able to provide the schedule to the user, the front end then requests schedule data from the server over HTTP. Upon receiving a response, the front end can display the schedule to the user.

![Figure 2 An overview of the system structure, where arrows represent data flow.](image)

6.1 Back end

The back end is data layer of the application, providing data to the front end. As described in Section 3.3, this project uses a bare-bones back end serving only mock data (that is, a set of fake employees and fake schedule assignments). The mock data is not changeable and only serves to provide newly connected clients with some meaningful data to display. In a finished application, the server-side software would be able to fetch
requested data from a database, and package this data into a response. More crucially, it would be able to receive requests to add or modify data, for example scheduling an employee to be in a room at a specific time.

As discussed in Section 5.6.2, the back end provides the endpoints to access its data through a RESTful API. An example of such a request can be seen in Figure 3. The front end sends a GET request for information about the employee with id 0, and receives a JSON object response containing the employee’s information. This information can subsequently be used by the front end to display it to the user and allow interaction with it.

![Diagram showing interaction between front end and back end through API]

**Figure 3** An illustration of a REST API request for employee with id 0. Response is in the dotted square.

### 6.2 Front end

In terms of software, the front end can be defined as the code that is executed and run on the client’s hardware. The front end uses AJAX and the Typescript framework Angular to update the page content according to the data from the back end.

AJAX is a set of web development techniques used when developing asynchronous web applications. These type of web applications can send and receive data from a server in the background — *after* the page has loaded — as well as modifying the contents of the page (for example when receiving new data). AJAX is used in the system in conjunction with the aforementioned back end API to request data from the back end, for instance about the schedule of the day, and subsequently update the already loaded web page with the newly received data. The framework used to manipulate the page is Angular. As was discussed in Section 5.6.1, an important benefit of Angular is its ability to provide a single page application (SPA) by manipulating the content of the page in several ways.
6.2.1 Differentiation between mobile and desktop views

The two main platforms used by the users are identified as mobile and desktop. To that end, the front end is able to provide two different web pages: one for mobile and one for desktop. The reason for two different pages are the general difference in screen space between the platforms, as well as differences in how the user interacts with their device. Desktop users tend to use mouse and keyboard, whereas mobile users generally use touch-screens.

The general structure of the desktop site can be seen in Figure 4. Each line represents a path to another page the user can navigate to or an action a user can take from the current page. Similarly, the general structure of the mobile site can be seen in Figure 5. A notable difference is that the mobile page includes an overview page as well as a detailed page for each room, whereas the desktop site simply has a large overview of all rooms. This is due to space constraints limiting the amount of information that can be fit on the screen on mobile devices.

Figure 4 A map over the pages available within the desktop web application

Figure 5 A map over the pages available within the mobile web application

7 Requirements and evaluation methods

The requirements on the system, as well as methods used to evaluate how well the system meets those requirements, are presented in this section. The requirements are
mostly centered around aspects of the application’s usability, and are accordingly evaluated primarily through usability tests.

### 7.1 Requirements

It is essential that the core functionality of the application is usable; ideally, it should be more usable than the analog system it builds upon and replaces. One common definition of usability, which we use here, is by Jakob Nielsen. Nielsen defines a usable interface as one that fulfills the five usability attributes *learnability, efficiency, memorability, errors, and satisfaction* [Nie94b, p. 26].

As we described in Section 3, the goal of the application is to facilitate day-to-day staff coordination and reduce the administrative burden on healthcare personnel, in part by delivering a schedule that is intuitive and efficient to use. This means that, aside from general usability, emphasis should be placed on those two qualities.

In terms of Nielsen’s definition of usability, intuitiveness is best translated into learnability, how easily users can accomplish tasks for the first time. Thus the system is also required to be learnable. In practice, this means that users should be able to perform all basic tasks, such as moving tiles around and editing times and comments, without having used the system before or getting help from anyone else.

Efficiency is defined as the ability for users to perform tasks quickly. Aside from designing an efficient user interface, the application as a whole can help be made efficient by ensuring it loads quickly. The efficiency of performing a single small task, such as looking something up on the schedule, is highly dependent on the time taken between the user accessing a page of the application and them being able to start interacting with it. Therefore another requirement of the application is that it loads quickly. To understand when a long load time starts to feel tedious for users, we looked at research made by Google on how load times affect the probability of users abandoning a web page, see [An18]. Based on this, we set a load time limit of 3 seconds.

### 7.2 Usability testing methods

As previously mentioned it is crucial that the system has high usability. This specific requirement is largely the responsibility of the user interface. To evaluate how well the system meets the usability requirements, usability tests were employed.

Usability tests are an approach used to evaluate a product by means of having a product
tested on the users. Two types of usability tests were used to evaluate the system: think-aloud and heuristic evaluation.

### 7.2.1 Think-aloud evaluation

The basic principle of think-aloud evaluation is to have users complete a set of tasks on the system it intends to evaluate. Any thoughts during the process of completing the tasks are then verbalised. The intention behind verbalising thoughts during or after the tests is to obtain data which is a direct reflection of the actual use of the tested system instead of user’s judgement of its usability.

Think-aloud evaluation has two variations based on when the thoughts are verbalised. With concurrent think-aloud evaluation, thoughts are verbalised during the completion of the tasks, whereas with retrospective think-aloud evaluation thoughts are verbalised after the completion.

In a study comparing the two methods for evaluation and discussing their respective advantages and disadvantages [vdHJS10], it was found that concurrent evaluation revealed more problems that were detected through observation of the user. In contrast, retrospective evaluation could only detect problems by having the user verbalise said problems after the test. The study concluded that concurrent assessment is more reliable in terms of providing the thought processes occurring during the test whereas when assessing retrospectively the users tended to provide a spectrum of reactions to different components of the system during the tests.

As the system was designed with the intention of having an intuitive interface, the concurrent evaluation method was used. The test subjects were asked to complete a list of tasks while voicing their thoughts. The tasks were split up into two categories; basic tasks that, in a real life setting, would be performed by all employees; and administrative tasks that would only be performed by administrative personnel. The basic tasks were:

- Find the schedule for the whole department.
- Find an individual’s staff members schedule.
- Find out if there is an employee named “Rolf”.
- Find out who is absent today.
- Add a time and comment to a name tag.
• Assign someone to lunch shift 2.
• Find all anesthesia personnel.

The administrative tasks were:

• Assign a nurse to Room 2 Anesthesia.
• Remove the person you assigned previously.
• Mark a person as ‘On leave’.

7.2.2 Heuristic evaluation

A heuristic evaluation is a type of usability evaluation in which a person evaluates a user interface according to certain usability design principles (called heuristics). Heuristic evaluations do not require the involvement of users, and are thus typically less time-consuming than user-centric evaluations. Since the problems found with such an evaluation can vary significantly between individuals, it is advisable to perform evaluations with several independent evaluators. [Sch04]

We chose to conduct a heuristic evaluation as a complement to the think-aloud evaluation, to test various usability aspects of the interface. Each contributor to the project conducted their own separate evaluation. We used heuristics proposed by Nielsen [Nie94a], who is one of the original inventors of the heuristic evaluation method [NM90]. The exact heuristics are defined and described in Appendix B.

The same tasks were used as for the think-aloud evaluations, and for each task the system was assessed with respect to Nielsen’s heuristics.

7.3 Evaluating load time

The load time of the application may vary depending on the network speed between server and client, and thus to obtain meaningful and reproducible results it was necessary to standardize this factor. To this end, we utilized the ability within the Google Chrome to throttle the network speed. This option was available in the network tab within Chrome’s developer tools, and was set to ‘Fast 3G’ to emulate a realistic low-bandwidth scenario. The load time was then measured as the time from the first request was sent to the server to when all elements within the page were loaded.
8 Formulating questionnaire questions and interpreting answers

As mentioned in Section 5.1, we sent out a questionnaire to the employees of the pediatric surgery department intended to use the application. The purpose of this was in part to obtain knowledge about the previous system for staff coordination in order to help uncover problems with it and to help formulate requirements on the new system. The questionnaire also allowed employees to voice their opinions about the previous system and what they wanted from the new one. This data was then used in the design process to create personas and to make decisions about the design of the user interface.

We first asked employees about their usage of the current whiteboard schedule with close-ended questions about their satisfaction with the system and how often they use it. We also asked users to recount any problems they experience with the whiteboard system. Then, we asked about their interest in a mobile application for managing the schedule, and posed an abstract, open-ended question about what the employees wanted from the new system. Lastly, we asked the respondents to enter their profession, so that we would be able to differentiate between doctors, nurses, and assistant nurses. The exact questions and responses can be found in Appendix A.

The survey respondents were moderately satisfied with the previous system, rating their satisfaction 4.8 on average, on a 1-6 scale. Unfortunately, we did not ask respondents to explain their rating, and so it is only possible to speculate about what they like about it. One possible explanation for their satisfaction with the system is its ease of use; moving magnets and writing on a whiteboard is likely familiar to most, especially those who have used similar systems on other departments. The system also allows anything to be written down or drawn on the whiteboard, in order for instance to add temporary assignment category or relay some important piece of information. This makes the system highly adaptable and customizable. Another factor might be that non-administrative personnel simply do not encounter or are not as affected by some of the shortcomings of the system. 70.6% of respondents answered that they typically make only one or no changes at all to the schedule during a work day, meaning they would not be bothered by the time it takes to make those changes. Other things, such as the inability to plan ahead for more than two days or the lack of history and statistics, would not directly affect non-administrative personnel at all.

When asked about problems they encounter with the whiteboard system, several employees responded that it tends to get cluttered and messy, and can become difficult to read. The same respondents all specifically point out the comments made with whiteboard pen as the cause of the messiness. The fact the most common complaint was
9 Developing use case data

about readability is likely a reflection of how most employees use the schedule. Almost 30% of respondents reported that they look at the schedule 4-6 per day with just as many looking at it 7 or more times per day, and as mentioned before a clear majority of employees make at most one change to the schedule per day.

The responses also made it clear that there are problems with the information on the whiteboard not reaching the employees. When asked how often they encounter problems as a result of lapses in communication regarding the schedule, 41% answered that they encountered such problems once every week, and 35% reported encountering them about once per month. This may be partly due to the lack of structure on the schedule, and its poor readability. One employee complained about the time it takes to walk to the whiteboard to check something, and it’s possible that reducing the effort it takes to check the schedule would help employees stay more up to date with the schedule. Ideally, this problem would be solved by an integrated communication system whereby changes to the schedule are automatically and directly communicated to, and subsequently confirmed by, those involved. However, as explained in Section 3.3 implementing a communication system is beyond the scope of this project.

Lastly, the questionnaire answers gave insight into the potential use of being able to view and manage the schedule on a mobile phone. When asked how often they would look at the schedule on their phones, should they be able to, 29% answered that they would do so daily, and slightly fewer answered that they would do so a few times per week. Significantly, 47% answered that they would ‘rarely or never’ look at the schedule on their phones. A few possible reasons for this are given in the responses to a later question. Several people expressed concern that using a phone in a hospital setting might be unhygienic or unsanitary. Especially for those who frequently spend time in sterile environments, it may become tedious to have to resterilize their hands just to look at the schedule. The problem is not present in non-sterile environments, making the mobile functionality useful for administrative personnel who spend more of their time in meetings and offices. One employee also expressed that they wished to keep their phone for private use only, while another expressed a general dislike and distrust of digitalized systems.

9 Developing use case data

Our insights from the questionnaire results and from interviewing with our stakeholder were used in conjunction with the methods provided by Arvola to understand the needs of the users and how these relate to the functionality and content of the application. Together with collecting it, this analysis and development of use case data constitutes
10 Designing the interface

the conceptual phase of Arvola’s design method.

We identified some important qualities for the interface to have. Firstly, it was clear from the questionnaire that the readability of the system was important to users, from this we concluded that we should place emphasis on making the interface clear and easy to read. Secondly, it seemed the employees made heavy use of the ability to write notes on the whiteboard, and we decided to try to achieve similar adaptability in the new system. Lastly, since a fair number of small changes are made to the schedule throughout the day, we decided that it should be uncomplicated and quick to make changes.

Two personas were also created. These are hypothetical users intended as representatives of the different user groups, in order to make sure the user group is well-defined and to promote empathising with the users [Arv14, p. 65]. One persona, Beatrice, was created to represent users with management-oriented roles who are very involved with maintaining the schedule. The other persona, Hamud, was created to represent those employees who mostly look at the schedule without changing it. Certain scenarios were devised to describe how the personas might want to use the application, some examples follow below:

- Hamud wants to find an anesthesiologist to get a second opinion
- Hamud wants to find Beatrice’s phone number
- Hamud wants to notify others that he might need to leave work early
- Beatrice wants to make sure every operating room has a nurse assigned for the entire day
- Beatrice wants to find an unoccupied surgeon she can lend to another department
- Beatrice wants to sketch tomorrow’s schedule without filling in any details

These scenarios were then used to reason about the data and functionality needed within the application, which resulted in the sitemaps shown in Section 6.2.1.

10 Designing the interface

Before implementing the interface, it was almost fully designed as a set of still images, detailing what it should look like. This was to done to speed up the design process, avoiding having to tweak HTML and CSS to make tentative changes to the design. It
was decided early on that two different designs were needed: one for desktop and one for mobile. This is because of differences between the platforms in screen size and interaction methods, as discussed in Section 6.2.1. Screenshots of the final designs can be seen in Figures 6, 7, 8, and 9.

To achieve the goal of making the interface intuitive and easy to use, the method used was the one discussed in Section 5.4. Part of it is making the interface familiar to the users. To that end, the desktop schedule interface (seen in Figure 6) was designed as a grid with a very similar structure to that of the previously used whiteboard schedule. Additionally, the staff tiles — the colorful elements with names and times of day — were color-coded by profession with the same colors used internally at Uppsala university hospital, which can be seen in Table 2.

<table>
<thead>
<tr>
<th>Profession</th>
<th>Swedish name</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medical Doctor</td>
<td>Läkare</td>
<td>Red</td>
</tr>
<tr>
<td>Nurse</td>
<td>Sjuksköterska</td>
<td>Blue</td>
</tr>
<tr>
<td>Assistant Nurse</td>
<td>Undersköterska</td>
<td>Green</td>
</tr>
</tbody>
</table>

Table 2 The color coding of professions in the application.

Since the mobile design could fit less information on the screen, the mobile schedule view was split into two: one overview (Figure 7a) and one detailed view with a tab for each room (Figures 7b and 7c). Nevertheless, the mobile elements were designed to be visually close to the respective elements on the desktop site; for example the staff tiles and the navigation bar look the same. This allows users to use their experience with one of the site’s variants to more easily navigate the other one.

Another important point described in Section 5.4 to make the interface intuitive and easy to use is Arvola’s principles. Consequently, these too were used in the design, a few examples of which can be seen below, where the principle is in italic:

- **Have a good conceptual model**: The conceptual model is a whiteboard, their previous type of scheduling system.

- **Have a good mapping between act and response**:
  - Right arrow is used to move forward in time with respect to both date and actions (redo). Left arrow is the opposite: back in time.
  - The buttons to add a tile in the mobile view are shown at the exact position where the tile will be added.

- **Make sure the feedback on actions is clear and visible**:
10 Designing the interface

- When dragging a tile to a new position, a placeholder element is always shown where the tile would end up if dropped.
- When dropping a tile after dragging it, it is animated to move towards the new position where the placeholder element is.
- Impactful actions such as saving or discarding changes are highlighted via a small message on the bottom of the screen.

![Figure 6](image)
The implemented design for the schedule view of the desktop application

One potential problem with the design is the high contrast of the staff tiles compared to other elements. This makes them highly emphasized, as was discussed in Section 5.5 regarding the visual hierarchy: that high contrast elements become accentuated and draw the attention of the user. With too many emphasized elements demanding attention, the user might miss vital information [Arv14, p. 127]. For example, the user might not immediately see the headlines, making it hard for them to understand the general structure of the page. However, we think this trade-off is worth it to be able to color code professions with colors that the users are familiar with.

When designing the schedule overview for mobile (Figure 7a), we needed to decide what to keep and what to relegate to the detailed view. We decided that the lunch- and break times should be kept in the overview, while the distinction between anesthesia and surgery staff for each room could be moved to the detailed view. This was based partly on an answer in the questionnaire (first answer for question 7) stating that lunch and break assignments are much more important than the room assignments. Additionally,
our stakeholder — head of the department — said that it often is enough to see who is assigned to what room, regardless of whether they are surgery or anesthesia staff.

Figure 7 The implemented design for the schedule views of the mobile application

Figure 8 The implemented design for the staff view of the desktop application
11 Communication between back end and front end

The communication between the front end and the back end is done over HTTP, as was mentioned in Section 6. Specifically, the data is sent serialized in JSON format. To achieve this, there were two main tasks. The back end (the server) needed to be able to serialize the data into JSON, and the front end needed to be able deserialize the received data into runtime objects that it could use. The back end’s serializing is handled completely by the framework used: .NET Core. However, the front end did not have an included deserializer, which prompted the creation of a custom-built one from scratch.

11.1 Implementing the JSON deserializer

The JSON deserializer builds an object by first creating an empty object of the correct type. Using this object, the deserializer then iterates over all its fields, setting their values to that of the JSON data. If a field’s datatype is a non-primitive, an object instance is created instead of just copying the value. This too is done by iterating over all the fields and filling them with values from the JSON data. The process continues recursively until the whole object has been created. A flowchart illustration of this process can be seen in Figure 10.
Currently, in Typescript there is no way of knowing at runtime what fields are available in an object type and what their types are. This prevents the deserializer from knowing which fields to just set directly to the value of the corresponding value in the JSON data, and which fields to set to a new object instance created by deserializing more JSON data. To circumvent this problem, all objects that should be instantiable from JSON data were given a field containing metadata of fields’ names and type constructors. This data was filled in manually when creating the classes.

During deserialization, all key names in the JSON data have their values copied into the fields with the same names in the object being created. For example, if the JSON data under key ‘name’ is ‘Anders’, then the created object will have the field name set to ‘Anders’. If the type metadata states that a field is of a non-primitive type, the field is set to a new object of that type. Then, the new object is populated with data in the same way.

**Figure 10** A flowchart of the deserializer built for this project. The *recursive step* close to the bottom invokes another identical step for the child object.
11.2 Setting up the RESTful API with .Net Core

As was specified in Delimitations (Section 3.3), the RESTful API was only employed to serve static mock data to the front end. To begin, data model classes were created. The instances of these classes are what is serialized and sent as a response to the front end, hence the classes were created to be as much as possible identical to that of the front end’s data models. This meant that the serialized data could be directly deserialized into fields with the same names on the front end, without having to specify certain JSON data keys for each field.

Follow creating data model classes, the API endpoints were set up to serve mock instances of these classes. The mock data was simply created as hard-coded objects created at application startup. To downside of this approach is that to change these objects, the application needs to be compiled again. However, for the purposes of just serving static mock data, this problem was not enough to warrant another solution.

The final step was to configure the CORS policy (Cross-origin resource sharing) to allow the Angular app to access the API. CORS is a security feature of browsers restricting the ability to make HTTP requests from within scripts [Moz19]. If the requested resource is not from the same origin as the application, the requested resource must explicitly allow the request it by including the application origin address in a header in the response. To simplify development, we decided to use an allow-all policy, meaning the API responses always include a CORS header that permits any origin of request. However, should the application be deployed in a production environment, this could be changed to only allow requests from the Angular application’s origin.

12 Evaluation results

The evaluation results include both the think-aloud evaluation and the heuristic evaluation, as well as from the load time evaluation. The results of the think-aloud evaluation consist of both general numbers about tasks performed and tasks successfully completed, as well more detailed comments about problems encountered during individual tasks. The heuristic evaluation focuses on specific aspects for each heuristics, both positive and negative, that came up during the evaluation.
12.1 Think-aloud evaluation results

During the think-aloud evaluation, two members of the pediatric surgery department and a member from the maternity ward at the same hospital participated: The head of the pediatric surgery department (tester A), assisting staff manager (tester B), and an assistant nurse at the maternity ward (tester C). A and B first both performed the tasks separately on the desktop site, after which tester A alone performed the tasks on the mobile version since an emergency situation occurred during the test. C performed the tasks on the desktop site at a later date, but did not have time to do a usability test on the mobile version.

On the desktop site, out of the ten tasks listed, tester A and tester C completed all tasks. Tester B was given eight of the tasks to perform and completed six, failing one basic task and one administrative task. The tasks not given to tester B were:

- Find out if there is an employee named "Rolf".
- Remove the person you assigned previously.

The reason for not doing all ten tasks was the aforementioned emergency situation that occurred. On the mobile site, out of ten tasks listed, tester A completed nine tasks, failing one basic task. The single failed task was Find out who is absent today. The tester navigated to the staff page where some tiles are marked as absent, but could not find a summary of all absentees today.

12.1.1 Problems encountered during tasks

Although most tasks were completed, not all were completed without confusion or misunderstandings from the tester. Notably, all three testers expected to be able to mark a person as “On leave” from the schedule page, but that functionality only existed on the staff page. Additionally, both tester A and B noted during “Assign a nurse to Room 2 Anesthesia” that they expected to be able to see the difference between anesthesia and surgery staff in the staff list on the right.

The problems only encountered by single testers are listed below:

Tester A

- During “Remove the person you assigned previously”, they took about 5 seconds to understand that staff tiles could be dragged by holding down the mouse.
• On mobile, during “Find out who is absent today”, they navigated to the staff page where some tiles are marked as absent, but could not find a summary of all absentees today.

**Tester B**

• During “Find an individual’s staff members schedule”, they had to be prompted to try to click on things to understand that tiles could be clicked on.

• During “Assign someone to lunch shift 2”, they failed the task by repeatedly dragging the mouse slightly when clicking the button to add lunch assignment, which initiated dragging of the staff tile.

**Tester C**

• During “Find out if there is an employee named Rolf” the tester noted that the employee could probably be found on the staff page. The tester unsuccessfully attempted to go to the staff page by clicking the staff list headline.

• During “Assign someone to lunch shift 2” the tester first attempted to drag their staff tile of choice into the lunch column. After unsuccessfully doing so they quickly noticed the “+” marks on the lunch column and was able to successfully complete the task.

**12.1.2 General remarks by testers**

During multiple tasks performed by all three testers when trying to change to the staff page, a popup showed which informed them that they had unsaved changes in the schedule. All three testers seemed unsure as to the exact meaning of this. Once understanding the message, they looked for a button on the popup to save changes, but there is no such button available.

In the staff page when performing tasks which required modifying employee data, tester A and B tried unsuccessfully to first click on the staff tiles. Only after it failed did they try to click on the correct button: the cogwheel.

**12.2 Heuristic evaluation**

As previously described, a heuristic evaluation was performed individually by each team member, assessing the usability of the system according to Nielsen’s heuristics (a de-
scription of each heuristics can be found in Appendix B). We then discussed and compiled the results. Below are the observations made for each heuristic:

Visibility of system status The user gets clear feedback in the form of loading spinners whenever the application is waiting on a response from the back end, and when moving a tile, it is clear what is being moved and where it would end up if the user dropped it. However, a problem was found with the undo functionality, where it is not immediately clear what unsaved changes the user has made to the schedule, and thus what would happen if the user pressed the undo or cancel buttons. This is a significant problem in the mobile view, where the schedule is split into several tabs; pressing the undo button might undo a change made in a tab different from the one the user is currently on, with no way for the user to know what happened.

Match between system and the real world The desktop site represents the schedule as a physical board with magnets that can be dragged around. On the mobile site, the connection between the layout and the real world is less clear, although the layout is still structured according to real world categories, such as operating rooms. Wherever relevant, we use terms and colors that the users are already familiar with from the real world.

User control and freedom On the desktop site, when dragging an employee tile it is not possible to tell where the tile came from, that is, where it was before the user started dragging it. This could lead to situations where a user starts dragging a tile by accident, and then is unable to cancel the action. Undo and redo are supported on both the schedule and personnel pages, and all dialogs have clearly marked ways to immediately close them.

Consistency and standards Most problems with consistency are between the desktop and mobile views. On the desktop view, the entire schedule can be viewed and edited on the home page, whereas the home page on the mobile view is an overview of the schedule, and editing is done on a separate page. The way most actions are performed also differs between the views.

Error prevention Some error prevention is done by automatically focusing newly appeared input fields to prevent the user from starting to type without the field being focused. Also, dialogs that have a clear primary and secondary action have their primary action clearly marked, to prevent misclicks.

Recognition rather than recall When adding assignments, it is not possible to see which work group employees belong to. Thus in cases where the user wants to assign a person from a specific work group, they would have to first find a suitable employee on the personnel page, and then remember their name as they go to the
schedule page to assign them. On the desktop view, the entire schedule is visible at all times and all editing is done in place, minimizing the need to remember what was clicked. On the mobile view most editing is done via dialogs, and these do not clearly specify what is being edited or added.

**Flexibility and efficiency of use** There are no keyboard hotkeys within the application, and no functionality for batch editing (such as removing many assignments at once). Assigning time or lunch shifts can be done via keyboard to achieve higher efficiency. On the desktop page, most actions can be efficiently performed, being able to be performed with one or two clicks. On the mobile page most actions require one or two more clicks.

**Aesthetic and minimalist design** The information on the schedule is limited to only that which is important, most actions are hidden by default and need to be 'discovered' by the user, in order to avoid cluttering. However, the high contrast of the staff tiles may make it difficult to see the structure on the schedule as a whole. The popup for editing staff members’ information contains a lot of information which is rarely edited, drawing attention away from the information which is edited very frequently, such as the employee’s absence status (sick, on leave etc.). Furthermore, the popup that appears when the user navigates away from a page with unsaved changes has a non-specific header and a lot of unrelated text.

**Help users recognize, diagnose, and recover from errors** Some error messages are provided in response to network errors, but these only show the HTTP status code and prompt the user to try again later. The error code is not useful to most users, and the message does not otherwise explain the error. On the mobile view, no error messages are provided whatsoever.

**Help and documentation** Some help is provided in the form of tooltips that specify what certain fields and buttons mean. Other than that, no explicit help is provided to the users.

### 12.3 Load time

When loaded five times in succession, with the page having previously been cached, the longest it took to load the desktop page was 2.49 seconds. With caching disabled, the longest time was 3.95 seconds. For the mobile site, the longest load time with cache was 2.45 seconds, while without caching it was 3.95 seconds.
13 Results and discussion

The final result of the project is a cross platform web application that can provide information about daily schedules and other employees to staff at a hospital department. Specifically, the application is tailored to the needs of the pediatric surgery department at Uppsala University Hospital. With minor changes, however, it has considerable potential to be able to be used at other departments and hospitals as well, should it be desired.

The project has fulfilled its purpose of providing an application that can display staff schedule, assign personnel to tasks and rooms, and find specific people and when they are available. The application has a schedule view and a staff view; the schedule view shows the schedule of the day, meaning it shows assignments to rooms and tasks along with optional comments and the times of day that these tasks or assignments take place. The staff view shows all the employees of the department, along with controls to filter the collection of staff. The staff view can for example be used to find a specific person, or all employees who are surgeons.

Another purpose of the application was to provide the schedule in an easily readable format. This was not specifically evaluated, making it difficult to establish whether the purpose has been fulfilled or not. However, it can be said that the application alleviates two problems with the previous physical schedule that could make it cluttered and messy; the application does not use handwritten text with whiteboard pens, which can be written with unclear handwriting or leave traces and specks if not erased properly. Thus the application fulfills the purpose of displaying the schedule more clearly in at least those regards.

In Section 3.1 about the application’s purpose, it is described that it should be easy to understand quickly how to use the application. To that end, a requirement set on the evaluation was that the users should be able to perform all the basic tasks, without having used the application before. Out of three testers each performing seven, six, and seven basic tasks respectively, only one task was failed on the desktop site. Moreover, the task failed because the user unintentionally clicked and dragged on the correct button instead of just clicking. On the mobile site, one tester participated, completing all but one basic task. Since almost all basic tasks were completed and one failed due to a minor issue, the application satisfies the criterion of being user-friendly and intuitive with respect to the basic tasks.

Also part of the application’s purpose was to help users save time. This purpose is partly fulfilled just by allowing the schedule to be accessed from anywhere, but it also depends on the efficiency of the application itself. The think-aloud evaluation does
not provide much insight into this efficiency, as users had not had time to learn the interface. Instead, we can look to the results of the heuristic evaluation and the load time evaluation. With the heuristic evaluation, we discovered an absence of efficiency-enhancing functionality. While most actions are by themselves fairly quick to perform, some could be performed quicker by defining keyboard hotkeys to them, or by allowing several similar actions to be performed in batches. The load time evaluation shows that the application loads quicker than the set requirement of 3 seconds, but only when it has previously been cached. For those who use the application frequently, it will most likely be cached, but that may not be the case for those who use it very infrequently or use an older phone. As discussed in Section 5.6.1, single page applications typically loading the entire application at once. However, since the most common use case is to simply look at the schedule without editing it, the average load time could be decreased by only loading editing functionality when it is needed. Weighing in both the efficiency of the user interface and the application’s load time, the application fulfills the requirement of efficiency only partly, and there is significant room for improvement in this regard.

Finally, a few general usability problems were discovered with the usability evaluations which concerned:

- Popup regarding unsaved changes.
- Modification of employee data.
- Finding information on whether an employee works in anesthesia or in surgery.

The first issue regarding the popup was believed to be caused by the failure to communicate the option to save, undo or redo changes to the user. When the user makes a change, a toolbar with the aforementioned actions appears at the bottom of the screen, which can be observed in Figure 6. If a user then attempts to leave the page a popup appears informing the user that they have unsaved changes. With no prior knowledge of the option to save changes the popup can be confusing and disruptive. To better communicate these options to the user the toolbar’s location could be changed to somewhere the user might spend more time looking, such as next to the navigation bar. The toolbar could also be made larger to make it more prominent on the screen.

Another general issue encountered was the modification of employee data which was necessary to complete the task “Mark a person as ‘On leave’”. When the users were instructed that the task was to be completed on the staff page, the staff tile itself was pressed instead of the correct button. The misstep may have been caused by the visual likeness between the tiles in schedule and staff. Because a prior task was to bring up an employee’s individual schedule the users may have thought that since pressing the tile brought up some information of the employee, that data could also be changed.
The almost identical appearance of the tiles between the staff page and the schedule page may convey the idea that the tiles have the same functionalities on both pages which they do not. To solve this particular issue the view that displays an employee’s individual schedule could also provide options for modifying data which is subject to frequent change, such as unavailability status or lunch shift.

The last general issue arose when finding out whether an employee worked within anesthesia or surgery. During the task “Assign a nurse to Room 2 Anesthesia” the testers attempted to find an anesthesia nurse from the list to the right on the schedule page. That list however does not include any information of whether an employee works in anesthesia or surgery. The possible cause may have been that the list has the headline *Personnel* (Swedish: *Personal*) which could have interpreted as the list having the same information or functionality as the staff page. A possible recourse that could be taken to fix this issue is to change the name of the staff page to a title that implies more details of the employee’s could be found there. It could also be useful to have a link to that page in the list. However, being that the two categories of staff, anesthesia and surgery, are so central in their work it would be convenient or even important to include that information in that list since it is being accessed so often.

### 14 Conclusions

This project shows how a web application can be developed for daily staff coordination in a hospital department. The developed application satisfies the criteria for required functionality; it provides a schedule where employees can be assigned to tasks and rooms, as well as providing a staff view where employees can be found by name or profession. Furthermore, the schedule and employee properties can be edited, for example to create a new assignment or mark an employee as absent.

The development of the application was focused on usability and intuitiveness. During evaluation of the application, the basic tasks were all completed with minor issues, and thus it was concluded that the resulting application is sufficiently intuitive to use. There is some room for improvement, with some minor usability issues that could be fixed, and some enhancements that could be made to the efficiency of the application.

All in all, we managed to develop and application that is tailored to the specific needs of the pediatric surgery department, and that mitigates some of the major issues with the previous analog system. If a functional back end was developed to support the application, it could be of real value to both administrative and non-administrative personnel. It also serves as a base upon which many future extensions could be built.
15 Future Work

Should the application be developed further, there are several areas that could be improved. Most importantly, a database could be implemented for the back end. This has been mentioned several times earlier in the report, including the system structure in Section 6.1. A database to store employee and scheduling data would enable modification of the data without recompiling and restarting the back end. More specifically, this would enable the front end to save changes by sending them to the back end, which for the users would mean that their modifications to the schedule would show on everyone else’s devices as well.

Another important point of improvement is a messaging system. This would serve as a way for employees to send notifications to other employees, for instance about scheduling changes. Ideally, a notification would show on their devices as well. Another potential use case would be to notify administrative staff of sickness or leave. To implement the messaging system, a messaging view could be integrated into the application. However, a simpler solution might be to use their email addresses, for example by sending emails about important changes to affected employees.

To alleviate some of the ethical problems discussed in Section 3.5 of employee and scheduling data being available to third parties, an authentication system could be integrated into the application. This mainly concerns protection of the data in the back end, requiring authorization to access and modify data. However, an authentication system could also require users of the front end to identify themselves. Both of these measures would provide control over who can access the data.

A feature that would potentially be of interest to the employees is to present scheduling history and statistics in a readable format. Since this application provides the schedule in a digital data format, it greatly facilitates such data analysis compared to using the previous system of a whiteboard schedule. A potential use case of statistics could be to see which operating rooms are used the most, or what days are the most difficult to find staff for.

Lastly, the application could be improved by generalising many aspects of it and providing greater customization options. Since it was developed with a specific department at Uppsala University Hospital in mind, some aspects of it might not be suitable for other departments and hospitals. This could include room names, profession color coding, profession names, and staff filtering options. Allowing customization of these could enable the application to be used by a wider variety of departments and hospitals.
References


A User questionnaire

Below are all the questions and possible answers comprising the questionnaire we sent out to the staff of the hospital department dealt with in this report. Translations into English are provided in *cursive*.

1. Hur nöjd är du med er nuvarande lösning för salsplacering? (Whiteboarden) *How satisfied are you with your current solution för room scheduling? (The whiteboard)*
   - Scale from 1, Inte alls nöjd (*Not at all satisfied*), to 6, Mycket nöjd (*Very satisfied*)

2. Ungefär hur många gånger under en arbetsdag kollar du whiteboarden med salsplaceringarna? *Approximately how many times during a work day do you check the whiteboard with the room schedule?*
   - 0-1
   - 2-3
   - 4-6
   - 7 eller fler (*7 or more*)

3. Ungefär hur många gånger under en arbetsdag gör du ändringar på whiteboarden (vad som helst, t.ex. flyttar på en magnet eller skriver en kommentar)? *Approximately how many times during a work day do you make changes to the whiteboard (anything at all, e.g. moving a magnet or writing a comment)?*
   - 0-1
   - 2-3
   - 4-6
   - 7 eller fler (*7 or more*)

4. Finns det något problem med whiteboarden som du ofta stöter på? *Is there any problem with the whiteboard that you frequently encounter?*
   - Short text answer

5. Hur ofta upplever du att information kring salsplaceringarna som inte når ut leder till problem? *How often do you experience problems as a result of lapses in communication around the room schedule?*
6. Om du kunde se salsplaceringarna i mobilen, hur ofta tror du att du skulle använda det? If you were able to check the room schedule on your phone, how often do you think you would do so?

- Dagligen (Daily)
- Några gånger per vecka (A few times per week)
- Sällan eller aldrig (Rarely or never)

7. Finns det något övrigt du tycker borde finnas på hemsidan som vi utvecklar, eller något vi borde tänka på? Is there anything else you think should be on the web page we’re developing, or something we should think about?

- Long text answer

8. Vilken övergripande roll har du inom sjukvården? What overall role do you have within healthcare?

- Undersköterska (Assistant nurse)
- Sjuksköterska (Nurse)
- Läkare (Medical Doctor)
- Annat (Other)

A.1 Responses

Below are all the responses to the question in the questionnaire, as described above in Appendix A. Answer options in tables are translated from Swedish, but translation to open ended questions are not provided.

1. Hur nöjd är du med er nuvarande lösning för salsplacering? (Whiteboarden) How satisfied are you with your current solution för room scheduling? (The whiteboard)
2. Ungefär hur många gånger under en arbetsdag kollar du whiteboarden med salsplaceringarna? *Approximately how many times during a work day do you check the whiteboard with the room schedule?*

<table>
<thead>
<tr>
<th>Rating</th>
<th>Frequency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5.9%</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>5.9%</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>11.8%</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>52.9%</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>23.5%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>17</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

3. Ungefär hur många gånger under en arbetsdag gör du ändringar på whiteboarden (vad som helst, t.ex. flyttar på en magnet eller skriver en kommentar)? *Approximately how many times during a work day do you make changes to the whiteboard (anything at all, e.g. moving a magnet or writing a comment)?*

<table>
<thead>
<tr>
<th>Answer</th>
<th>Frequency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>1</td>
<td>5.9%</td>
</tr>
<tr>
<td>2-3</td>
<td>6</td>
<td>35.3%</td>
</tr>
<tr>
<td>4-6</td>
<td>5</td>
<td>29.4%</td>
</tr>
<tr>
<td>7 or more</td>
<td>5</td>
<td>29.4%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>17</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

4. Finns det något problem med whiteboarden som du ofta stöter på? *Is there any problem with the whiteboard that you frequently encounter?*
Answers (verbatim)

Fullt att folk skriver på samma tavla med olika färger, ser ostrukturerat ut, jobbigt för ögat, svårt att förstå vad som är extra viktigt

Den är ofta rörig men mycket kladd på

Det kan ofta se rörigt ut pga av för många anteckningar

Tuschen svår att få bort, ser ”skitig” ut. Sjukfrånvaro/vabb- listan inte alltid helt uppdaterad.

Det tar onödigt lång tid att gå till tavlan och kolla, när det är något man vill veta (ex lunchplanering)

Nej

5. Hur ofta upplever du att information kring salsplaceringarna som inte når ut leder till problem? How often do you experience problems as a result of lapses in communication around the room schedule?

<table>
<thead>
<tr>
<th>Answer</th>
<th>Frequency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>About once per year</td>
<td>4</td>
<td>23.5%</td>
</tr>
<tr>
<td>About once every month</td>
<td>6</td>
<td>35.3%</td>
</tr>
<tr>
<td>About once every week</td>
<td>7</td>
<td>41.2%</td>
</tr>
<tr>
<td>Almost every day</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>17</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

6. Om du kunde se salsplaceringarna i mobilen, hur ofta tror du att du skulle använda det? If you were able to check the room schedule on your phone, how often do you think you would do so?

<table>
<thead>
<tr>
<th>Answer</th>
<th>Frequency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily</td>
<td>5</td>
<td>29.4%</td>
</tr>
<tr>
<td>A few times per week</td>
<td>4</td>
<td>23.5%</td>
</tr>
<tr>
<td>Rarely or never</td>
<td>8</td>
<td>47.1%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>17</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

7. Finns det något övrigt du tycker borde finnas på hemsidan som vi utvecklar, eller något vi borde tänka på? Is there anything else you think should be on the web page we’re developing, or something we should think about?
A User questionnaire

Answers (verbatim)


Borde finnas bekräftelsefunktion att meddelandet mottaget.

På de flesta ställen inom sjukvården är det inte ok att ha mobil (hygien, ect) men en hemsida skulle vara till stor hjälp. Är det möjligt att koppla sjukskrivning till hemsidan? Ex genom att man skickar ett sms när man är sjuk/vabbar och att detta automatiskt kommer upp på hemsidan? Det skulle underlätta väldigt mycket! Just nu är det väldigt många samtal fram och tillbaka för att sjuknäla sig samt att ringa till den som är sjuk för att kontrollera när denne kommer tillbaka osv.


8. Vilken övergripande roll har du inom sjukvården? What overall role do you have within healthcare?

<table>
<thead>
<tr>
<th>Profession</th>
<th>Frequency</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assistant nurse</td>
<td>4</td>
<td>23.5%</td>
</tr>
<tr>
<td>Nurse</td>
<td>10</td>
<td>58.8%</td>
</tr>
<tr>
<td>Medical Doctor</td>
<td>2</td>
<td>11.8%</td>
</tr>
<tr>
<td>Other</td>
<td>1</td>
<td>5.9%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>17</td>
<td>100%</td>
</tr>
</tbody>
</table>

1The single answer in ”Other” (verbatim):

Har även varit PA,- alltså haft koll på tavlan, skrivit upp den, planerat den osv. Kan vara därför jag också kollar ofta på den. Vet inte om alla andra gör det lika mycket som mig (även de dagar jag ej är PA)
B Usability Heuristics

Below are the usability heuristics used for our heuristic evaluation. The heuristics were developed by Nielsen [Nie94a] (descriptions are also by Nielsen).

Visibility of system status The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom Users often choose system functions by mistake and will need a clearly marked “emergency exit to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall Minimize the user’s memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
Help and documentation Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user’s task, list concrete steps to be carried out, and not be too large.