Using an ICT System to Increase Efficiency and Effectiveness in Patient-Practitioner Communication

Magnus Gustafsson
Jonas Norlinder
Michael Rehn

Master Programme in Computer and Information Engineering
Antalet människor med mentala sjukdomar så som depression ökar i Sverige. Ett verktyg som används av vårdgivare under behandlingen av depression är självskattningsformulär, dessa kan uppskatta hur allvarlig depressionen är. Den nuvarande arbetsmetoden är att patienter fyller i formulär med papper och penna, en arbetsmetod som kan vara mycket ineffektiv. För att lösa problemet utvecklades ett IT-system som möjliggör både produktivare och effektivare kommunikation mellan patienter och dess vårdgivare. Produktivare i den bemärkelsen att föröver formulärsvar från papper till patientjournalen inte längre kommer vara nödvändigt och effektivare då systemets datavisualisering tillåter vårdgivaren att fatta mer välgrundade beslut. Resultatet var ett IT-system, färdigt att användas av en vårdenhet, som dessutom har utvecklats med den rådande lagen i åtanke. Systemet som utvecklades anses göra hanteringen av självskattningsformulär mer produktiv och effektiv, dock kan vissa funktioner i systemet förbättras eller tillföras för att ytterligare öka produktiviteten och effektiviteten i kommunikationen mellan patienter och vårdgivare.
Abstract

Using an ICT System to Increase Efficiency and Effectiveness in Patient-Practitioner Communication

Magnus Gustafsson
Jonas Norlinder
Michael Rehn

The number of people dealing with mental health issues like depression is increasing in Sweden. One common tool used by healthcare practitioners when treating depression is the use of self-assessment questionnaires, which estimates how severe the depression is. The current modus operandi is to let patients answer questionnaires with pen-and-paper, a practice with major inefficiencies. Our aim was to develop an ICT (information and communication technology) system that enables efficient and effective communication between patients and practitioners. Efficient in the way that manually transferring questionnaire results from pen-and-paper into the medical record is not necessary and effective in the way that the system enables data visualisation that makes it easier for the healthcare practitioner to make decisions. The result was an ICT system, ready for use by a healthcare provider, that was developed with the current regulations in mind. The system developed was deemed to make the utilisation of self-assessment questionnaires more efficient and effective. However, some features in the system can be further improved or added, to increase the efficiency and effectiveness of patient-practitioner communication.
Contents

1 Introduction 1

2 Background 1

2.1 Impact on Community 2

2.2 Using ICT Tools for Self-Assessment 2

2.3 Considerations when Using ICT Tools 4

2.3.1 Regulation on Data Privacy 4

2.4 Usability and Design 5

2.5 External Stakeholder 6

3 Purpose, Aims and Motivation 6

3.1 Delimitations 7

4 Related Work 7

4.1 ICT Systems for Patient-Practitioner Communication 7

4.1.1 Replacing Pen-and-Paper with a Mobile Application 8

4.1.2 Mood Tracking with a Mobile Application 8

4.1.3 Text Messaging with a Therapist 9

4.1.4 Usefulness of Self-Assessment Questionnaires 9

4.2 Security 10

4.2.1 Protecting Data 10

4.2.2 Authentication Methods 11

4.2.3 Decreasing the Number of Bugs 11

5 Implementation Method 12
5.1 User Interface ..................................................... 13
5.2 Server Logic and Data Communication ...................... 14
5.3 Authentication and Session Management ...................... 15

6 System Structure .................................................. 15
   6.1 Users and Roles ............................................... 16
   6.2 API Server and Web Server ................................. 18
   6.3 Mobile Phone Application ................................... 18

7 Requirements and Evaluation Methods ....................... 18
   7.1 Protecting Data ............................................... 19
      7.1.1 Identification Checks ................................. 19
      7.1.2 Data Encryption ....................................... 20
      7.1.3 Authentication Method .................................. 21
   7.2 Effective and Efficient Patient-Practitioner Communication .......... 21
      7.2.1 Interview Method ...................................... 21
   7.3 Bridging the Gulf of Execution and Evaluation .................. 22

8 System Implementation ........................................... 22
   8.1 The System from a Patient’s Perspective ..................... 23
   8.2 The System from a Practitioner’s Perspective ................. 24
      8.2.1 Data Visualisation ...................................... 27
   8.3 The System from an Administrator’s Perspective ............... 28
   8.4 Data Protection .............................................. 30
   8.5 Authentication Process ...................................... 31
      8.5.1 Authenticating Using BankID .......................... 32
8.6 Session Management ............................................. 32
8.7 Regulation Compliance ............................................ 34

9 Evaluation Results ................................................. 37
  9.1 Protecting Data ...................................................... 37
    9.1.1 Identification Checks ....................................... 37
    9.1.2 Data Encryption ............................................. 38
    9.1.3 Authentication Method ..................................... 40
  9.2 Effective and Efficient Patient-Practitioner Communication .................. 41
  9.3 Bridging the Gulf of Execution and Evaluation ............................. 42
    9.3.1 Analysis Using Norman’s Criteria ............................ 42
    9.3.2 Summary of the Usability Analysis ............................ 46

10 Results and Discussion ........................................... 47
  10.1 Regulation Compliance .......................................... 47
  10.2 Usability .......................................................... 47

11 Conclusions .......................................................... 48

12 Future Work .......................................................... 49

A Appendix ............................................................... 57
  A.1 Results of Unit Tests ............................................. 57
  A.2 Entity–Relational Diagram ...................................... 61
  A.3 List of Endpoints that Triggers a Log Entry ....................... 62
  A.4 List of Tasks ....................................................... 63
  A.5 List of Prepared Questions Asked during the Interview ............... 65
List of Figures

1. Example of one of the current modus operandi when writing the medical record notes to a patient’s answers to a self-assessment questionnaire. Where each number indicates what answer was made to a specific question.

2. The overall technical structure of the system, depicting how code is reused for the different parts.

3. System structure from the perspective of a patient. Depicts how the patient communicates with the API server and how the API server responds to that communication.

4. System structure from the perspective of a practitioner/administrator. Depicts how the user access the system and how it communicates with the API server.

5. A test runner executes one test at a time against the API server, which responds with a result. The test runner interprets the result as either pass or fail.

6. The user interface presented to a patient, which informs what questionnaires are available to answer.

7. An illustration of the process of which a patient is answering the MADRS-S questionnaire.

8. The user interface presented to a practitioner.

9. The complete view when inspecting data for an individual patient.

10. Example of how the visualisation of answers to the daily mood questionnaire is charted.

11. Example of a chart that displays the results of the individual questions in the MADRS-S questionnaire.

12. The user interface presented to an administrator.

13. Example of the user interface when an administrator is examining the system logs.

14. The authentication prompt.

15. The process of authenticating a user with BankID.
16 The process of authenticating a user. .................................. 33
17 The process of requesting privileged data in the system. ............ 34
18 Data captured in-transit before and after encryption while a user re-
quests to authenticate. .................................................... 39
19 Inspection before and after encryption of the raw data that includes the
contents of the “users” table. ............................................. 40
20 Informing the patient that no additional actions can be taken. ...... 43
21 The relational diagram representing the database structure ........ 61
Glossary

affordance A relationship between a physical object and a person, e.g. a user affords starting the mobile phone application, or, the user affords logging into the system. [54, p.10].

conceptual model A model created by a user of how something, like a system, works. May or may not correspond with how the actual system works, as long as it helps to explain to the user how the system will behave [54, p.25].

constraint Something that limits the set of possible actions. A constraint can be physical, cultural, semantic and logical [54, p.125].

CSPRNG “Cryptographically secure pseudorandom number generators (CSPRNGs) are pseudorandom number generators that protect against attack while still providing high quality pseudorandom values” [43].

feedback Information that helps the user understand what has happened [54, p.72].

ICT Information and communication technology.

need-to-know “Denoting or relating to a principle or policy of telling people only what is deemed necessary for them to know in order to carry out a task effectively” [59].

plaintext The state of data before it is encrypted [10].

principle of least privilege “The principle of least privilege is a concept in computer security, promoting minimal user profile privileges on computers, based on users’ job necessities” [39].

push notification A notification sent to a client using push technology, i.e. where data can be transferred from a server to a client without the client requesting the data [38].

signifier Something that communicates where an action can take place [54, p.14].

unit test A type of testing where each component of the system is run one at a time in isolation [15].
1 Introduction

The cost of mental health issues in Sweden is estimated to 2% of its GDP and at least 50% of that is attributed to work absence and out-of-work benefits [56]. If the treatment were to be improved, e.g. by decreasing convalescence, patients would not only be better off, but costs could be reduced as well.

Torous et al. found that a common flaw in healthcare practices is the manual administrative labour introduced by self-assessment questionnaires [71]. Not only was the management of such questionnaires found to be time-consuming when answered on pen-and-paper, but they were also more dishonest as opposed to a computerised system for answering questionnaires.

An information and communication technology (ICT) system, the one described in this report, was developed to combat the inefficiencies in healthcare. The system is designed to enable a continuous flow of communication between the healthcare practitioner and their patients. This is accomplished by having the patient answer self-assessment questionnaires in a web-based application installed on their personal smartphone.

If the data is collected digitally, it enables possibilities that were unfeasible before. One possibility is enabling the healthcare practitioner to easier see trends in a patient’s well-being by plotting the patient’s questionnaire results over time. The end-goal is to enable an efficient communication between patients and practitioners with the help of self-assessment questionnaires. In addition to data visualisation, a warning system was implemented that is able to alert the patient’s practitioner if the patient’s questionnaire answers indicated a serve result.

While designing the system, four major concerns had to be considered: (i) How can the system improve the effectiveness and efficiency in-between appointments between healthcare practitioners and their patients? (ii) How is sensitive data transmitted and stored securely? (iii) How can the system be designed so that it is deemed usable? (iv) Does the system comply with the current regulations? This report describes how we tried to answer these four questions.

2 Background

ICT tools for communication between patients and medical practitioners in healthcare have historically been poorly implemented [57]. While other areas greatly benefit from the ICT revolution, the healthcare industry struggles to modernise in some areas as
indicated by Lind [44].

Meanwhile, mental health issues in Sweden, especially among children and young adults, have increased in recent years [8]. This is not a problem specific to Sweden. The World Health Organisation has reported a 14.9% increase in anxiety disorders and an 18.4% increase in depression on a global scale occurred between 2005 and 2015 [76].

2.1 Impact on Community

It was estimated in 2004 that the annual cost of depression in Europe was 118 billion EUR [64]. According to OECD, the estimated cost of mental health issues in Sweden reached 70 billion SEK in 2013, around 2% of Sweden’s GDP [56]. Furthermore, a cost analysis of depression in Germany, performed in 2013, found that 44% of the cost was inpatient care, i.e. care given to patients residing in a hospital [42].

Since the reported number of people having mental health issues have increased [76], it is not unreasonable to assume that the costs have increased as well. If the cost associated with mental health issues could be decreased, more resources could be spent on each individual patient resulting in better care.

If effective treatment is defined as less convalescence and the treatment’s effectiveness indeed is improved, then the patient and its family would experience less ailment. Together with the economic aspect, it would improve the quality of life for not only the patient but also for the community.

2.2 Using ICT Tools for Self-Assessment

The Montgomery-Åsberg Depression Rating Scale (MADRS-S) and the Beck Depression Inventory II (BDI-II) are commonly used as self-assessment tools for depression [74]. Both are questionnaires that yield a score. This means, in the case of MADRS-S, that a total score between 0-54 is possible, where the level of depression is defined as (i) minimal: 0-12, (ii) mild: 13-19, (iii) moderate: 20-34 and (iv) severe: \( \geq 35 \).

Karl Engström, one of the external stakeholders, works as a psychiatrist in Sweden. He is under the impression that the modus operandi of managing self-assessment questionnaires is inefficient [26]. One example is that practitioners are required to manually enter the answers to a questionnaire alongside with the text of the medical record, this is depicted in Figure 1. Engström believes that this practice makes it hard to track changes in individual answers and that a lot of time is wasted on transferring results from paper
into the medical record.

```
Medical record for patient John Doe, 19010101-0101

2018-01-01: Feeling worse. Result of MADRS-S: (6, 5, 5, 4, 4, 3, 4, 4)

2018-01-10: Changed SSRI medication. Result of MADRS-S: (6, 5, 5, 4, 4, 3, 4, 4)
```

Figure 1: Example of one of the current modus operandi when writing the medical record notes to a patient’s answers to a self-assessment questionnaire. Where each number indicates what answer was made to a specific question.

Engström further assesses that most of the self-assessment questionnaires used in healthcare are answered with pen-and-paper and that questionnaires are usually answered before appointments. It is also the case that patients may be asked to answer the questionnaire while in the waiting room to the emergency ward [26]. He noted that one possible problem with this is that this could yield misrepresentative data, since if a patient is in the emergency waiting room they are ill, which would result in less truthful answers as opposed to continuously answering the questionnaire at home.

One article claims that practitioners in the United States use a similar modus operandi to what Engström describes above [71]. The article also claimed that using pen-and-paper practice is labour intensive, which wastes valuable time and resources. This could imply, in our opinion, that patients who need help may not receive it in time. If the questionnaires can be answered with the help of a proper ICT tool, it can hopefully make the use of self-assessment questionnaires more desirable.

There is some interest from the consumer market of applications that tracks the user’s mood. One of these applications, Daylio, which targets a general audience has more than 1 000 000 downloads on Google Play [20].
2.3 Considerations when Using ICT Tools

A major concern that ICT tools introduces is that collection and storage of digital data impose security risks [2]. In recent years, reports about data leakage of databases that included medical records [35] or attacks on pacemakers [36] has been published. Therefore, emphasis must be put on achieving a high security protecting the patient. When storing and processing personal data digitally, the current regulations must also be taken into consideration.

2.3.1 Regulation on Data Privacy

All entities, that process or store personal data within EU, must comply with the General Data Protection Regulation (GDPR) according to The Swedish Data Protection Authority (Datainspektionen) [19]. This means for example that the user has a right to request an excerpt of all data, that has been stored about the user. Before deploying a new system, that stores personal data, an analysis that considers all risks and making all necessary attempts to reduce those risks must be performed. If a security incident occurs, the incident must be reported to the respective country’s data protection authority within 72 hours.

Another law, that regulates how personal information must be stored and used within healthcare in Sweden, is the Patient Data Law (Patientdatalagen) [17]. According to The Swedish Data Protection Authority (Datainspektionen), the Patient Data Law has precedence over GDPR but cannot lower GDPR’s standards [18]. Three important matters the law requires are: (i) that only people who need access to medical records in their capacity as a practitioner should have access, (ii) all actions performed must be traceable to one specific individual and (iii) that a strong method of authentication must be used. The National Board of Health and Welfare (Socialstyrelsen) has compiled a handbook on how to deal with the Patient Data Law, which will be useful in a system that must take the Patient Data Law into consideration [65].

The National Board of Health and Welfare (Socialstyrelsen) requires that a strong method of authentication is used when patient data is processed in a computer system. They define it as a method that uses at least two out of the three following criteria: (i) using something the user knows (e.g. password, pin code, etc.), (ii) using something that the user has (e.g. a certificate, smart card etc.), (iii) or with the help of the user itself (e.g. fingerprint, iris, etc.) [65].
2.4 Usability and Design

It is important for a system to have a high degree of usability, e.g., making it easy for a first-time user to understand the system or minimizing the cognitive cost of using the system. Don Norman, who has a total of 111,994 citations [32], states in his book “The design of everyday things”, which is cited 18,896 times [32], that there are two gulfs a user have to cross when they want to use something: The Gulf of Execution and The Gulf of Evaluation [54, p.38]. A user tries to cross the Gulf of Execution when they are evaluating how to reach their goal and they try to cross the Gulf of Evaluation when they are evaluating what happened. According to Norman, it is the designer’s role to help their users to bridge these two gulfs in order to create a usable system.

To bridge the Gulf of Execution a designer should make use of signifiers, constraints and a good conceptual model. A good conceptual model is also needed to bridge the Gulf of Evaluation but the designer also has feedback to their disposal [54, p.40].

Norman further describes seven stages of action. A user goes through these seven stages when performing an action [54, p.40-41]. To make the stages more tangible, Norman breaks the stages down into seven questions a user must answer during the action. These questions are:

1. What do I want to accomplish?
2. What are the alternative action sequences?
3. What action can I do now?
4. How do I do it?
5. What happened?
6. What does it mean?
7. Is this okay? Have I accomplished my goal?

Question 2, 3 and 4 are questions the user must answer to cross the Gulf of Execution and question 5, 6 and 7 is something the user must answer to cross the Gulf of Evaluation. If these questions can be answered through sufficient use of the tools described above, the designer has helped the user to bridge the gulfs.
2.5 External Stakeholder

Steffi Knorn and Christian Rohner, both researchers at Uppsala University, have together with the industry professional Karl Engström identified the problem with the current modus operandi in patient-practitioner communication. Engström is a psychiatrist working at a clinic in Gävle, Sweden, and serves as the industry expert on this project. Knorn, Rohner and Engström sparked the idea of a possible solution and with their input we developed a system that overcomes the problems mentioned in Section 1.

Knorn, Rohner and Engström believes that the communication between patients and practitioners could be more efficient and effective utilising an ICT system. This ICT system has to be able to continuously gather data via questionnaires before the appointments, allow a practitioner to use that data to better prepare for appointments and alert the practitioner when that data has changed drastically or reached a critical level.

3 Purpose, Aims and Motivation

The aim of this project was to develop an ICT system that enables communication between a patient and their practitioner in-between appointments. As defined by our stakeholders in Section 2.5, the ICT system should be capable of (i) continuously gathering data via questionnaires before the appointments, (ii) allowing the practitioner to use that data to better prepare for appointments and (iii), alerting the practitioner when that data has changed drastically or reached a critical level.

Since the data will be collected digitally it opens new possibilities that have not been feasible before, e.g. automatic alarms, better tools for visualising trends in a patient’s well-being and automatic documentation over the course of a patient’s illness. This should be possible since the questionnaires (as mentioned in Section 2.2) often yield a quantifiable score.

We believe that our end result will solve some of the inefficiencies in the patient-practitioner communication as described in Section 2.2, which may lead to better treatment that will hopefully reduce unnecessary suffering.

There are four major questions to be answered if a patient-practitioner communication solution is to be developed: (i) How can the system improve the effectiveness and efficiency in-between appointments between healthcare practitioners and their patients? (ii) How is sensitive data transmitted and stored securely? (iii) How can we bridge the Gulf of Execution and Evaluation in the system? At last, (iv) how well does the system comply with the current regulations?
3.1 Delimitations

The initial target country will be Sweden and thus the system will be limited to citizens of Sweden. This is necessary since a strong authentication method is needed as described in Section 2.3.1 and the most widespread method in Sweden as of today is BankID which is only available for people permanently residing in Sweden.

The system will be developed as a general patient-practitioner questionnaire system that could be used for any questionnaires given to patients on a regular basis. Our stakeholders have asked us to focus on adults with depression and to have the MADRS-S questionnaire in mind while developing the system.

Only questions with quantifiable answers will be an option and no support for text input will be implemented even though supporting other types of question types, like text input, could be of great value, since more detailed answers could be provided from the patients. Nevertheless, having quantifiable data will make it easy to implement automatic alarms and visualise data, ergo we have delimited the system to quantifiable data.

Data for each patient will not be integrated with their official medical records. Having integration with the current medical records system would be desirable to increase the effectiveness and efficiency. However, due to time constraints, this integration was left for future work.

The application will only be developed for web browsers, Android and iOS. Even if more platforms could be added the time to test and perform quality assurance on more platforms would be out of scope for this project.

4 Related Work

There are two main problems the system aims to solve – first, create a more efficient and effective way for patients and practitioner to communicate, second, store confidential information about patients safely in a database.

4.1 ICT Systems for Patient-Practitioner Communication

Some attempts have been taken to use ICT systems for patient-practitioner communication. This includes, but is not limited to, a trial to replace self-assessment questionnaires with a mobile application, mobile applications that track a patient’s mood or allows a
patient to talk to a licensed therapist using text messages, or an entire new work methodology at child health centres (Barnavårdscentraler).

### 4.1.1 Replacing Pen-and-Paper with a Mobile Application

An article published in Journal of Medical Internet Research describes how Torous, et al. developed a custom mobile application that would allow patients dealing with depression to answer the Patient Health Questionnaire-9, a questionnaire similar to MADRS-S described in Section 2.2 [71]. Their objective was to examine the correlation between scores retrieved from the mobile application and scores retrieved by pen-and-paper.

Torous, et al. found that the results from the mobile application strongly correlated with the questionnaire answered with pen-and-paper. One main difference was that the results provided by the mobile application recorded a higher percentage of patients with suicidal thoughts compared to questionnaires answered with pen-and-paper [71]. Their conclusion was that answers from the mobile application may more reliably capture suicidality than questionnaires answered with pen-and-paper, i.e. the answers were sincerer.

The mobile application used by Torous, et al. was only focused on the Patient Health Questionnaire-9, while our solution is a generic questionnaire system. Torous, et al. mainly focused on the patient’s perspective, whereas our solution has a more holistic approach, which includes the perspective of a practitioner and an administrator as well. This introduced parts that their system did not have, e.g. data visualisation and automatic warnings, which are core parts of our system.

### 4.1.2 Mood Tracking with a Mobile Application

There are many applications on the market today that can help with tracking a user’s mood. One of them, that more specifically target people with manic depression, is Bipopular.

Bipopular has daily questionnaires, with tracking indicators such as the patient’s depression level, energy level and so on. It will also ask the patient if they have taken their medicine according to their doctor’s specification [9]. The patient can also automatically synchronise their data with their doctor. The application does not use strong authentication as defined in Section 2.3.1, only username and password.

There are some undesired user design decisions in Bipopular that make it more difficult for a user to cross the Gulf of Execution and Evaluation discussed in Section 2.4. As
an example, answering a questionnaire makes it difficult to cross the Gulf of Execution since it uses a slider that looks continuous but whose values instead are discrete. This contradictory design makes it more difficult for the user to create a useful conceptual model which is needed to bridge the Gulf of Execution.

There are also problems when a user tries to cross the Gulf of Evaluation when reading the charts generated from the user’s answers. Many different data sets were plotted into the same area chart. This resulted in overlaps of different data. This made the chart difficult to read and once again it results in difficulties for a user to create a sufficient conceptual model and the user, therefore, has difficulties crossing the Gulf of Evaluation.

Our solution will use strong authentication as described in Section 2.3.1, to maximise security for the patient and it will carefully consider the Gulf of Execution and Evaluation, discussed in Section 2.4. It will be a more general solution, where the user can use any questionnaire they want to track anxiety, depression, pain levels and so on.

4.1.3 Text Messaging with a Therapist

Talkspace is a mobile application, that provides a secure messaging service between patients and a licensed therapist [68]. Other similar services are 7Cups and iPrevail [3] [40]. Talkspace’s own assessment is that their service is one-third of the cost of traditional therapy [70]. Thus using ICT tools could potentially reduce costs in therapy, which would lead to better resource allocation and potentially better treatment of patients. Talkspace is meant to be a substitute for treatment while our solution is a tool that aids practitioners in their treatment.

4.1.4 Usefulness of Self-Assessment Questionnaires

Researchers at Centre for Health Equity Studies, Karolinska Institutet, Stockholm University and the Child Care Department in Stockholm (Barnhälsovårdsenheten i Stockholm) developed a new work methodology at selected child health centres (Barnavårdscentraler) in Stockholm, with focus on behaviour, mental health and parental support for 3 and 4-year olds. The methodology was developed in a project called BVC-Elvis [7].

One of the tools BVC-Elvis makes use of is a web-based ICT system where parents find information and answer questionnaires. The nurses at the child health centre can access the system and receive a summary of the parents’ answers. The ICT system gives the nurses suggestions on what the answers in the questionnaires imply. The goal of using
the ICT system, according to the report, is to provide a basis from which to organise the appointments.

The report found that 77% of nurses have use of the questionnaires, while only 3% said they had no use. What is more, nurses said that it was easier for them to prepare the appointment and that the parents themselves were more prepared for the appointment. The BVC-Elvis project has similarities with our proposed solution and gives an indication that healthcare practitioners could get better insight with the help of well-designed ICT tools.

While our system and the ICT system in BVC-Elvis have many similarities, there are some significant differences. BVC-Elvis targets children and our system targets adults. Our system also tries to visualise trends by plotting a patient’s answers over time without giving any indication of the correct course of action and reminds patients to answer questionnaires through notifications on the mobile application. Furthermore, if a drastic change in a patient’s answer occurs, an automatic alarm will notify the practitioner.

4.2 Security

Security issues can arise from many parts of the system. This includes how to securely store, read and transfer data externally and internally within the system, authenticating user and ensuring that the system behaves in a well-defined manner.

4.2.1 Protecting Data

Microsoft has written a summary regarding data security best practices for users of their cloud service and how to follow them [77]. Although written with their cloud service in mind, some of these recommendations are applicable to computer systems in general. They describe two states of data, at-rest and in-transit, as well as their recommendation on how to secure these states. At-rest data is described as all information that exists statically on physical media while in-transit data is described as data that is being transferred between two different nodes.

To protect data in-transit Microsoft recommends to always use transport layer encryption and authentication protocols when exchanging data across different locations [77]. If possible, the communication channel should be isolated through a virtual private network. This protects from man-in-the-middle attacks, e.g. data cannot be eavesdropped, and data integrity is guaranteed [37]. For data at-rest, the contents of the database should be encrypted and use a file system encryption.
Microsoft recommends using need-to-know and the principle of least privilege for ensuring that the security policy in place is enforced [77]. This means that a user should have the least amount of access to the system while still being able to do their work. Not following these principles may lead to data compromise since users can access more data than they should. Nevertheless, implementing the principles is an achievable software engineering task, but it is not always easy to decide who should have access to what data.

4.2.2 Authentication Methods

There are different methods for authenticating a user. One method is electronic identification, which is a solution that provides secure identification online [50]. Eggestig and Wodajo developed a mobile phone application which used BankID, an implementation of electronic identification, as a secure method of authentication [25].

Eggestig and Wodajo performed a security analysis on their own system and concluded that while BankID provides a secure method of authentication, the phone can be stolen or lost during a session, left behind for whatever reason or lent to another person, which would result in an attacker having access to the user’s information. They noted that while some of these problems could be solved by the user, e.g. the user could always lock their phone with a strong passcode, that it could also be solved by the system itself by simply requiring re-authentication after a fixed interval of time.

4.2.3 Decreasing the Number of Bugs

It is of utmost importance to minimise bugs since they can cause security issues. One example of a bug that affected a lot of users was Heartbleed, which was a security bug in the OpenSSL cryptography library [23]. It was estimated that between 24%-55% of the most popular websites were affected. This bug allows attackers to read sensitive memory from vulnerable servers. This could include login credentials, keys and other private data.

Different programming languages use different type systems. A type system is a set of rules and types that expressions in a programming language can adopt [16]. These rules are enforced by the language’s type checker. Type checkers are usually classified into two categories: static and dynamic [29]. A statically typed programming language performs type-checking at compile time while in a dynamic type system the type of an expression is derived at runtime. Static typing is generally more favourable than dynamic typing since dynamic typing typically is more demanding on the central processor.
5 Implementation Method

An article concluded that introducing static typing into the otherwise dynamically typed language JavaScript eliminated a subset of bugs. This subset was empirically proved to be 15% of all bugs [29]. With the statically typed system introduced, these bugs will be detected at compile time as opposed to using dynamic typing. Statically typed systems should be better since bugs can be detected and handled earlier.

A language with a strictly defined syntax forces a developer to write code in a more concise way and should be easier to read and understand if formatted in a readable way. The developer can introduce a stricter style by using linters [72]. A linter is a tool to statically analyse source code to detect errors, bugs and deviations from the standard coding style. One study proved inter alia that developers typically use linters in JavaScript since it is good at detecting errors before they even become a runtime bug. The use of linters helps to avoid ambiguous and complex code and maintain a consistent coding style which further helps to reduce bugs.

5 Implementation Method

If all parts of the system can be developed using the same programming language it could enable a coherent system and maximise code re-use, e.g. type declarations on the server can also be used on the client. If the code base can be run on all targeted platforms without any additional work from the programmer’s side time could be saved since only one application has to be written instead of one for every targeted platform.

ECMAScript, more commonly known as JavaScript, is a programming language currently at its 8th edition. ECMAScript 5th edition [24] is at least 96% implemented by the platforms we target [62]. The missing 4% of the implementation is not essential to the system subject to be developed, which is why the 5th edition is a reasonable choice as a programming language to implement our system.

Since JavaScript is a dynamically typed system instead of statically typed as described in Section 4.2.3 we have opted to use the programming language TypeScript instead. TypeScript is a statically typed superset of JavaScript [47] that is able to compile to ECMAScript 5th edition [34]. An alternative to TypeScript would be Flow, a static type checker for JavaScript [28]. Because TypeScript has support for additional features that Flow lacks, such as enumerators, classes etc, it was preferred over Flow.

The development of the system will make extensive use of TSLint [60], a linter that supports TypeScript. The linter enforces a strict coding style which helps make the
code coherent as described in Section 4.2.3.

Since a system like ours deals with patients, special consideration regarding GDPR and the Patient Data Law must be considered. To achieve compliance, we will make use of the handbook mentioned in Section 2.3.1 that is written by The National Board of Health and Welfare (Socialstyrelsen). The recommendations deemed relevant for the system are described in Section 8.7 below, as well as how they were considered during the development of the system.

5.1 User Interface

To enable bundling and publishing our code as a mobile application, we used PhoneGap. PhoneGap is a tool that allows using web-based technologies to develop mobile applications. These applications support both Android and iOS [4]. The application is written as a website, with the exception that the PhoneGap API is available to use. The PhoneGap API allows access to native mobile functions such as the camera.

The plugin `phonegap-plugin-push` is used to enable push notification support in a PhoneGap application [61]. Push notifications was implemented to remind the user to answer a questionnaire, a feature requested by the external stakeholders of the project.

There are many different methods of creating a user interface (UI) for a web application, e.g. HTML with CSS, PHP [69], or Angular [31]. The UI of our system will be developed using React, a JavaScript library for creating UIs [27]. React uses a declarative style of UI programming, which means that the program describes how the result should look like, but not how the result is obtained [11].

It is possible to use another approach, e.g. a procedural programming style. However, we preferred a declarative programming style, since we believed that many parts of the system could easily be written declaratively. React is module based by design, which means that the UI written in React is divided into components. The components can be re-used inside an application which further improves code re-use. Furthermore, there is a large ecosystem around the React library, e.g. searching for packages related to React yields a result of 54 603 packages[55].

Writing user interfaces in React, as stated above, consists of designing components which can later be used within the application. An example of the ecosystem React provides is the library Material-UI [46]. The Material-UI library provides a set of React components that implements Google’s design language “Material Design” [33]. The Material-UI library will be used, where the predefined styling will save time during development, but also help bridge the gulfs described in Section 2.4.
The Material-UI library should help bridge the gulfs since the “Material Design” language is the default and recommended design language in the Android mobile operating System [5], which holds over 70% of the mobile phone operating system market share [67]. This implies that users are accustomed to the constraints, signifiers, feedback and conceptual model implemented by the “Material Design” Language.

5.2 Server Logic and Data Communication

To execute code on the backend server – such as user authentication, permission checks and more – we require a JavaScript runtime, a system that allows execution of a high-level language like JavaScript [14]. Node is a JavaScript runtime [51] with a total of over 650 million downloads [53] and it was chosen as a runtime since no other viable alternative was found together with the fact that the developers of the system already were familiar with Node.

The client will communicate with the backend server through an application programming interface (API) specifically designed for the system. As an example, the API will manage user authentication and fetch survey data.

The communication between the client and the API server is done with TLS, a protocol that allows both encrypted and digitally signed connections [22]. The API will be implemented using Express, a web framework running on Node [48]. Express is capable of handling HTTP requests over encrypted, as well as unencrypted, connections and respond accordingly.

As a database server, MariaDB was chosen since it has free support for encrypting the contents of the database tables, a method called data at-rest encryption [45]. MySQL supports data at-rest encryption as well, but only in the paid enterprise editions of the database server [58]. This is a necessary measure to secure the sensitive data stored in the case of a malicious attacker gaining access to the database.

Interfacing with MariaDB will be done using object-relational mapping (ORM), which maps software objects to relational database tables [66]. The system will use Sequelize, an ORM made for Node [21]. Using ORM, development will in our opinion be accelerated since ORM allows us to focus on writing application logic in the main programming language instead of writing complex SQL queries.

Firebase Cloud Messaging (FCM) is a solution, that enables delivery of push notifications to both Android and iOS devices [30]. While there are other alternatives available, like Apple Push notification service [6], FCM will be used due to the cross-platform functionality.
5.3 Authentication and Session Management

Secure authentication is a vital part of the system since it is required by law as described in Section 2.3.1, but also because it is important that the users can trust that their sensitive data is secure. The electronic identification system BankID, as described in Section 4.2.2 will be used as authentication instead of the traditional username and password combination.

While BankID is not the only electronic identity provider, it is widely spread in Sweden; 8 million Swedes will have access to BankID [73] which is a wide majority of Sweden’s roughly 10 million inhabitants (February 2018) [63]. This widespread use indicates that users are accustomed to BankID and already have a conceptual model of how it works. This helps bridge the Gulfs of Execution and Evaluation (see Section 2.4).

It is necessary to keep track of a user’s session, i.e. whether the user is authenticated and who the user is. To track a user’s session the open industry standard JSON Web Token will be used [41]. JSON Web Tokens allows the system to safely transfer claims between different parts of the system, e.g. the user claiming to be authenticated. JSON Web Token’s role is verifying that claim.

The JSON Web Token standard allows the server to issue a token to the client, encoded with data that the user needs, e.g. role type, expiration date of the token or any other relevant data. The token also includes a signature created by the server using a secret key. A signature allows the server to verify the issuer of the token [12]. If the client sends this token to the server with every request the server can verify the signature to make sure that this token indeed was created by the server itself. In short, a token is used to represent a session.

6 System Structure

The overall technical structure of the system is divided into four different parts as illustrated in Figure 2. As mentioned in Section 5 the usage of TypeScript as a programming language allows the system to achieve a high level of shared code. As depicted in the figure below, all the code of the system is written in TypeScript. The system has a shared set of code such as custom type definitions and functions that can be used both in the frontend and the backend.
Figure 2: The overall technical structure of the system, depicting how code is re-used for the different parts.

All technologies explained in Section 5 are placed within Figure 2 to illustrate where they exist in the system. The frontend consists of the UI framework React, the library Material-UI and the shared code. The backend is served by the Node runtime, utilising the Express framework, the Sequelize ORM, the shared code and communicate with FCM to send push notifications. Finally, the mobile application is packaged by PhoneGap, together with the frontend and PhoneGap specific code to enable plugins like phonegap-plugin-push.

6.1 Users and Roles

The system is designed with three roles in mind: administrators, practitioners and patients. As seen in Figure 3, a patient may answer questionnaires assigned to them by sending data to the API server via a mobile phone application. The API server validates who the user is, what role it has and if it can answer this questionnaire. If all checks are passed the data is stored in the database.
The practitioner’s flow of events is similar to the patient’s, but not identical, as illustrated in Figure 4. It is different since the practitioner access the system UI through a website, served by the web server. Additionally, rather than producing data, e.g. answer questionnaires, the practitioner consumes data. It is similar in the way that the checks of whether the user can perform the action are performed on the API server.

Finally, administrators cannot perform any of the actions that the practitioners and patients can. Instead, this role is used to manage practitioners and questionnaires. The
reasoning behind separating this functionality from the practitioner is to follow the principle of least privilege. The role of a practitioner is used to give care, while the role of an administrator is used for system configuration. The administrator receives information in the same manner as the practitioner.

6.2 API Server and Web Server

To give access to the database as well as business logic such as authentication, notifications and more, we need an API server. The frontend clients communicate with the API server, as described in Section 6.1. This communication is done through HTTP requests, a protocol that can be used for transferring data [13]. The requests are transferred encrypted and digitally signed with the use of the TLS protocol [22].

As previously mentioned in Section 6.1 the UI is accessed differently depending on if the user is a patient or an administrator/practitioner. The web server’s purpose is to serve the frontend UI, as can be seen in Figure 4. The frontend UI is built with React, Material-UI and TypeScript.

6.3 Mobile Phone Application

PhoneGap is used to bundle the application code for the patient as mentioned in Section 5.1. This means that the patient has no use of the web server which is apparent in Figure 3. Instead, the application code is bundled with the mobile phone application installed on the patient’s phone. In contrast to the patient, the practitioner accesses the system through a web browser.

An installed application was chosen as the UI delivery method for the patient because the patient must be able to receive push notifications. While push notifications are starting to become possible for websites accessed by browsers, there is narrow support for the technology [49], ergo PhoneGap is used in conjunction with the `phonegap-plugin-push` plugin to enable push notifications.

7 Requirements and Evaluation Methods

In Section 3 four major questions where presented that must be answered when developing the system. These questions are: (i) How can the system improve the effectiveness
and efficiency of the communication in-between appointments between healthcare practitioners and their patients? (ii) How is sensitive data transmitted and stored securely? (iii) How can we bridge the Gulf of Execution and Evaluation in the system so that it is deemed usable? (iv) Does the system comply with the current regulations? Evaluating regulation compliance is outside the scope of this project. However, all implementation decisions were made according to the handbook describing how to comply with the Patient Data Law written by The National Board of Health and Welfare (Socialstyrelsen).

7.1 Protecting Data

Verifying that all parts of a system are completely secure is difficult. Whereas, verifying that the system conforms according to some requirements is more achievable. The security aspects translate into three overall requirements: (i) necessary identification checks to make sure users can only access the data they need according to the specification, (ii) encrypting both at-rest and in-transit data and (iii) a suitable authentication method is implemented.

7.1.1 Identification Checks

The system requires, that identification checks are done according to the need-to-know and least privilege principles. This leads to separation of privileges using three defined roles: administrator, practitioner and patients, as described in Section 6.1. Their needed privileges are as follows:

1. Only the patient, to whom a questionnaire is assigned, may answer the questionnaire.
2. Only a practitioner may access a patient’s answers.
3. Only a practitioner may assign a questionnaire to a patient.
4. Only a practitioner may register a patient to the system.
5. Only an administrator may create or edit a questionnaire.
6. Only an administrator may register a practitioner or another admin to the system.

These conditions were decided based on guidance given by the external stakeholders while having need-to-know and least privilege principles in mind.
To verify that these conditions have been met, unit tests were written and run against the API server. Unit testing is a type of test where each component of the system is executed one at a time in isolation [15]. The process of executing tests can be seen in Figure 5 and consists of (i) the test runner calls a test on the API server and (ii) verifying if the API server conforms to the intended behaviour by interpreting the result as either pass or fail. If all the tests are interpreted as passing, the system is deemed to have met the least privilege requirement.

![Diagram of API Server and Test Runner](image)

Figure 5: A test runner executes one test at a time against the API server, which responds with a result. The test runner interprets the result as either pass or fail.

### 7.1.2 Data Encryption

The system requires that all data, both at-rest and in-transit, is encrypted. The purpose of this requirement is to make it more difficult for a potentially malicious actor to access stored and sent data. Furthermore, Microsoft recommends to not only encrypt the database tables but the file system itself [77]. However, this will not be implemented due to limited resources and the fact that it is more a feature of the operating system running the system rather than the system itself.

To verify whether this requirement is fulfilled, an inspection of the files stored and the data transferred between nodes, will be inspected. If the data both at-rest and in-transit are encrypted, we deem that the system satisfies the requirement. The analysis of in-transit data will be done with Wireshark, a tool that allows inspections of all network packets that travel through a network interface [75]. To evaluate whether the database content is encrypted, an inspection of the files will be made to verify that no data is stored as plaintext.
7 Requirements and Evaluation Methods

7.1.3 Authentication Method

Since BankID is the only authentication method allowed in the system, it is important that it works as intended. The system requires, that all essential scenarios for authentication work. The essential scenarios are defined as: (i) successful/unsuccessful authentication, (ii) aborting authentication attempts and (iii) automatically opening the BankID application when using the patient mobile application. It will be evaluated by manual testing, which will consist of exploring each essential scenario with five different mock social security numbers and observing if the behaviour is correct.

7.2 Effective and Efficient Patient-Practitioner Communication

It is required that the system improves the efficiency and effectiveness of the communication between healthcare practitioners and their patients. Two questions arise from this requirement: (i) Will the system reduce time spent on manually processing self-assessment questionnaires (both from the perspective of a practitioner and the patient)? (ii) Will the practitioner be able to better adjust a patient’s treatment with the help of the information gathered by the system?

To evaluate this, an interview was conducted with one of the stakeholders described in Section 3. The interview focused on evaluating if the tool could be of help with inpatient care and tried to answer whether the interviewee believes that the requirements (i) and (ii) are satisfied by the system.

It would be preferable if an efficiency and effectiveness study of the system was conducted. This would include participants from all user groups. However, since the time of the project is limited, this is dedicated to future work.

7.2.1 Interview Method

The interview layout consisted of three major parts: (i) having the interviewee performing given tasks (see Appendix A.4 for the full list) without any interaction with the interviewers, (ii) asking the interviewee according to a structured list of prepared specific and general questions (see Appendix A.5 for a list of all questions) and (iii) general discussion. To minimise bias, all questions were phrased as objectively as possible, e.g. instead of asking “was this good?”, the question was phrased as “what do you think about it?”. The purpose of no interaction in part (i), was to minimise bias that is
introduced if we help the interviewee use the system.

The interview will be conducted with one of the external stakeholders, Karl Engström, who works as a psychiatrist. General discussion about solving the problem and presenting him with some sketches was done before the interview, but it was Engström’s first experience with the system that was recorded. He had not seen or used any part of the real system before the interview.

Since Engström is one of the external stakeholders that sparked the idea of a solution, he could be biased, considering he probably wants the system to be as good as possible. Still, he is an industry professional and his opinions should provide some value.

### 7.3 Bridging the Gulf of Execution and Evaluation

The system needs to be usable (in the sense described by Norman, see 2.4). If the system is difficult to use, a user may get discouraged to use it. Thus, it is required that the system bridges the gulfs and the seven questions of design that Norman defined as

1. What do I want to accomplish?
2. What are the alternative action sequences?
3. What action can I do now?
4. How do I do it?
5. What happened?
6. What does it mean?
7. Is this okay? Have I accomplished my goal?

The requirement was evaluated by analysing the usability of the system. If the seven questions of design could be answered, it was deemed that the gulfs have been bridged and that the system is usable.

### 8 System Implementation

The system implementation can be viewed from three primary viewpoints: (i) as a patient, (ii) as a practitioner, (iii) and as an administrator. We will also highlight key
implementation areas, such as data visualisation, data protection, an authentication process, session management, authentication using BankID and regulation compliance.

8.1 The System from a Patient’s Perspective

If a user is successfully authenticated and identified as a patient, the system will present the patient home page. This can be seen in Figure 6 where the patient is presented with two courses of actions: either answer how they are feeling today or start answering the MADRS-S questionnaire. The figure also illustrates how appropriate feedback is given when a user has answered the daily mood question by pressing one of the smileys.

Figure 6: The user interface presented to a patient, which informs what questionnaires are available to answer.

When a patient has pressed the “STARTA” button next to “MADRS-S” in Figure 6, the questionnaire is displayed as illustrated in Figure 7. The questionnaire displays one question at a time, enabling the user to browse through questions with the help of a “previous” (Föregående) and a “next” (Nästa) button placed at the bottom. In-between the previous button and the next button there is a progress bar, indicating how many questions have been answered and how many are remaining. Radio buttons with each alternative to the question are displayed. Once an alternative has been chosen, the next button is changed from grey to colourised and the user can continue.
**8 System Implementation**

![MADRS-S Somn](image)

Sömn
Här ber vi dig beskriva hur bra du sover. Tänk efter hur länge du sovit och hur god sömnen varit under de senaste tre nätterna. Bedömningen skall avse hur du faktiskt sovit, oavsett om du tagit sömnmedel eller ej. Om du sover mer än vanligt, sätt din markering vid 0.

- **0** - Jag sover lugnt och bra och tillräckligt länge för minabehov. Jag har inga särskilda svårigheter att somna.
- **1** - Jag har vissa sömnsvårigheter. Ibland har jag svårt att sova eller jag sover alltför tidigt.
- **2** - Jag sover minst två timmar mindre per natt än normalt. Jag vaknar ofta under natten, även om jag inte blir stört.
- **3** - Jag sover mycket dåligt, inte mer än 2-3 timmar per natt.

Figure 7: An illustration of the process of which a patient is answering the MARDS-S questionnaire.

Since the patient accesses the system through a mobile application, push notifications can be used. The implemented notification is reminding the patient to submit an answer to the daily mood questionnaire. The push notification’s purpose is to minimise data loss. Data loss could make it harder for the practitioner to understand the patient’s state of well-being.

If notifications are sent too frequently it is possible for the patient to be annoyed and simply dismiss the notification without giving them much thought or even disable the notification, thus rendering the notification useless. To avoid this problem, the patient can select the frequency of the notification.

### 8.2 The System from a Practitioner’s Perspective

The practitioner’s page consists of dynamic boxes that can change its contents on demand. The practitioner’s start page is depicted in Figure 8 and has a list of current warnings of patients and a full list of all patients in the system. Both rows are clickable and clicking on either will trigger retrieval of questionnaire data for that patient which can be seen in Figure 9, where the retrieved questionnaire data is visualised on
the right-hand side of the page.

<table>
<thead>
<tr>
<th>Förnamn</th>
<th>Efternamn</th>
<th>Personnummer</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Doe</td>
<td>190101010101</td>
</tr>
</tbody>
</table>

Figure 8: The user interface presented to a practitioner.
Figure 9: The complete view when inspecting data for an individual patient.
8.2.1 Data Visualisation

To be able to help the healthcare practitioner make decisions, it was important to make it easy to digest the answers to the questionnaire. In Figure 10, an example is illustrated of how the daily mood question is visualised. The turquoise colour represents the answers over a period of time. Possible answers include: sad (encoded as -1), neutral, (encoded as 0) and happy (encoded as 1). The blue line is all the answers aggregated up to a particular day. This makes it possible to see trends. The result is a chart that makes it possible to see day-to-day answers alongside with the trends.

![Figure 10: Example of how the visualisation of answers to the daily mood questionnaire is charted.](image)

A questionnaire like MADRS-S consists of several questions, as opposed to the daily mood questionnaire. It could be of great value to track the answer to an individual question. In Figure 11, a practitioner can examine the results of each individual question of the MADRS-S questionnaire over time. By default, the results of all questions are displayed simultaneously, but it is possible to select what questions to view.
Figure 11: Example of a chart that displays the results of the individual questions in the MADRS-S questionnaire.

8.3 The System from an Administrator’s Perspective

The administrator can perform four tasks: (i) manage other administrators and practitioners accounts, (ii) contact the developer of the system for support, (iii) manage questionnaires and (iv) view the log. In Figure 12, an example is given of the full view of the administrator user interface.
One important matter for achieving regulation compliance is to be able to view the log. In Figure 13, an example is given on how this functionality looks like. An administrator can select a day and trace all actions taken by a user.
8.4 Data Protection

As described in Section 2.3, it is important that the system is secure, since the system deals with sensitive data. When designing data protection, we take both the requirements from the Patient Data Law and the recommendations from Microsoft described in Section 4.2.1 into consideration.

The database server is separated from the API server and runs on a different machine. Data at-rest encryption is implemented protecting the data from a malicious user that has access to the physical hard drive. The database server is not exposed to the Internet and allows only connections from the same private network. The communication over the private network between the API server and the database was done over TLS.

The two types of client interfaces (mobile application and web server) was protected by
not actually storing any data on the client. Instead, all data that needs to be collected and stored is sent to the API server. All communication with the API was done using TLS. A distinction with privileges between administrator and practitioners was made, as described in Section 7.1.1, according to the principle of need-to-know and least access principle. For instance, an administrator cannot view any patient in any way, they do not even know that they exist. Thus, only practitioners can manage patients. Likewise, practitioners cannot view any other administrator or practitioner.

8.5 Authentication Process

The authentication process is the same for all three roles, although administrators and practitioners receive the UI from a web server whereas the patient already has it bundled with the PhoneGap application.

In Figure 14, the UI while authenticating with BankID is portrayed. The user starts the authentication attempt by entering their social security number. If the social security number is valid (according to the specification given by The Swedish Tax Agency, “Skatteverket”), the authentication attempt will start when pressing the login button, otherwise an error message will be displayed.

![Authentication Process](image)

Figure 14: The authentication prompt.
8.5.1 Authenticating Using BankID

The process of authenticating against the system using BankID is depicted in Figure 15. The authentication process starts with (i) the user makes a request to authenticate by sending their social security number to the API server, (ii) the API server retrieves that request, re-packages the request according to the BankID API specification and uses the client certificate to sign the request such that the BankID server knows that request came from the API server, (iii) either the authorisation is successful/unsuccesful or gets timeout, after a given amount of time the API server will get a response from the BankID server stating whether the authentication succeeded or not, (iv) if it was successful, the API server must verify that the user exists in the system and then respond the client with a token or a message stating why the authentication failed.

![Authentication Process Diagram](image)

Figure 15: The process of authenticating a user with BankID.

8.6 Session Management

The system keeps track of a user’s session. This is necessary since we cannot require a user to authenticate every request they perform. As mentioned in Section 5.3, we implemented JSON Web Token to manage user sessions.

To be able to create and verify a token’s signature, a secret key is needed. This key must be kept secret from all, which means that generating it manually and supplying it to the system’s source code is not possible since if the code were to be leaked (or open sourced for that matter) a token would no longer be able to be trusted. Instead the
Node crypto.randomBytes function is used [52]. crypto.randomBytes uses a cryptographically secure pseudo-random number generator (CSPRNG). The secret key is thus generated at server initialisation. This means that every time the server restarts, a new authentication must be performed since the secret key has been changed.

The flow of a client requesting a token is illustrated in Figure 16 where (i) the user initialises the authentication process by submitting authentication parameters to the API server, (ii) the API server validates the authentication parameters provided by the user and denies the authentication request if the parameters are invalid, (iii) the API server fetches the user object from the database and (iv) creates a token with the encoded user object, metadata and digital signature and sends the response to the now authenticated client.

Between step (iii) and (iv) the token is encoded with the user object data. This is necessary for two reasons. First, the same authentication system is used for all types of users (admin, practitioner and patient). Thus, the client needs to know which page to display after a successful authentication. Second, for future user requests, the API server needs to know information about the requesting user to allow or deny the request. Only authorised users should be allowed to access privileged data. The metadata included is information about the token itself, e.g. when the token expires etc.

When a user tries to access privileged data, they must supply the server with a token. The process is illustrated in Figure 17, which illustrates (i) the user sends a request to the server, asking for privileged data. The token is verified on the server using the private key, (ii) if the token is valid, i.e. not expired and with a matching signature, the
privileged data is fetched from the database and a response with the fetched data is sent to the client. If the token is deemed invalid, a response will be sent informing the user that they have to re-authenticate.

![Diagram of System Implementation](image)

**Figure 17:** The process of requesting privileged data in the system.

### 8.7 Regulation Compliance

Complying with current regulations is a complex issue. To help mitigate this, The National Board of Health and Welfare (Socialstyrelsen) has compiled a handbook with guidance on how to comply with the current regulations when digitally storing and processing medical records.

Reading the handbook, we could conclude that only the following chapters of the Patient Data Law (Patientdatalagen) were deemed relevant for consideration when developing the system: 3 chapter 2 §, 15 § and 4 chapter 9 §.

We consider compliance with the paragraphs mentioned above by following The National Board of Health and Welfare’s (Socialstyrelsen) recommendations on how to comply with the regulation. Although the regulation is written for users in healthcare, a system developer must take necessary precautions to make the users of the system able to follow the regulation [65].
3 chapter 2 § HSLFS-FS 2016:40

**English translation**
The healthcare provider must ensure through the control system that

1. documented information is accessible and usable for those who are entitled
2. personal data is undistorted
3. non-authorised shall not read personal data and
4. all actions must be able to be traceable to one specific user in information systems that are partly or completely automatic

**What was done to be able to comply**

- User roles, with authorisation checks, were implemented to ensure access only for those who are entitled, the roles also help with making sure that the data is undistorted, e.g. patients should not be able to view or change other patient’s data. All user accounts are personal and linked to one’s personal security number.

- A system, that logs all actions so that every action is traceable, was implemented. The system only allows the logs to be viewed by an administrator.

- An authentication method (that prevents non-authorised user to read personal data) was implemented using BankID, which are deemed to be able to comply with the definition of a strong authentication. The National Board of Health and Welfare (Socialstyrelsen) defines a strong authentication method as a method that uses at least two of the three following criteria: (i) using something the user knows (e.g. password or pin etc.), (ii) using something that the user has (e.g. a certificate, smart card etc.), (iii) or with the help of the user itself (e.g. fingerprint, iris etc.) [65].

3 chapter 15 § HSLF - FS 2016:40

**English translation**
If the caregiver uses an open network in dealing with personal information, then the healthcare provider is responsible for
1. transferring of the data is done in such a way that unauthorised persons can’t access it and
2. electronic access or direct access to the data is preceded by strong authentication

What was done to be able to comply

- The system encrypts data in-transit via the TLS protocol [22].
- The electronic identification system BankID is implemented and is the only method of authentication in the system.

4 chapter 9 § HSLF - FS 2016:40

English translation
The care giver is responsible for

1. that it in the logs is clear what actions have been taken with the patient data,
2. that it is clear from the logs at what care unit or process the actions have been taken,
3. that it is clear from the logs at what date and time the actions have been taken,
4. that the users’ and patients’ identity can be found in the logs,
5. that systematic and re-occurring random checks of the logs are done,
6. that the controls of the logs are documented and
7. that the logs are saved for a minimum of five years to enable verification of access to the data about a patient.

What was done to be able to comply

- A log system was implemented
9 Evaluation Results

- From the logs it is possible to read (see Appendix A.3 for all endpoints in API server that get logged):
  - what actions have been taken with the patient data
  - who initiated the action
  - at what time did the action occur
- To increase the security of the logs, the database user used by the API server only have read/write access to the log. There is no automatic way for deleting the logs.

9 Evaluation Results

The requirements of the system were defined in Section 7 and where derived from the four major questions presented in Section 3. These questions were: (i) How is sensitive data transmitted and stored securely? (ii) Does the system comply with the current regulations? (iii) How can the system improve the effectiveness and efficiency in-between appointments between healthcare practitioners and their patients? (iv) How can the Gulf of Execution and Evaluation in the system be bridged so that it is deemed usable?

9.1 Protecting Data

To satisfy the security requirements defined in Section 7.1 the system is required to fulfil that (i) the system implements the needed identification check to make sure users can only access the data they need according to the need-to-know and least privilege principles, (ii) the system encrypts both at-rest and in-transit data and (iii) the authentication method works as intended.

9.1.1 Identification Checks

Unit tests were written and run against the API server to verify that the system’s identification checks comply with the conditions defined in Section 7.1.1 and listed again below.

1. Only the patient, to whom a questionnaire is assigned, may answer the questionnaire.

2. Only a practitioner may access a patient’s answers.
3. Only a practitioner may assign a questionnaire to a patient.
4. Only a practitioner may register a patient to the system.
5. Only an administrator may create or edit a questionnaire.
6. Only an administrator may register a practitioner or another admin to the system.

All conditions but the first could be tested with unit tests. The reason the first condition could not be tested is that the API server inserts answers to a questionnaire based on who is logged in. Hence, the test runner cannot supply the API server with another user id.

All other conditions could be tested and the output of the test runner is given in the appendix Section A.1. The modules relevant to each condition are

1. No test applicable
2. GET /api/answer
3. POST /api/assigned
4. POST /api/user
5. POST /api/questionnaire
6. POST /api/user

As can be seen in the output of the test runner, all applicable tests passed. This yields the result that all but one condition is satisfied.

### 9.1.2 Data Encryption

There were two requirements on the system regarding data encryption: (i) in-transit data should be encrypted and (ii) at-rest data should be encrypted. This was evaluated by inspecting the data in-transit and at-rest before encryption and comparing it with the data in-transit and at-rest after the encryption.

The in-transit data analysed was gathered when a client requests to start the authentication process. This is done by sending a request to the destination /api/login/bankid/auth together with the requesting user’s social security number. The data gathered with an unencrypted connection is displayed to the left in Figure 18 whilst the data from
the encrypted connection is depicted to the right. The figure demonstrates that information about the user such as their social security number, is sent in plaintext on an unencrypted connection and that the information is scrambled on the encrypted connection. This is the case for all data transferred over the encrypted connection and thus the requirement of encrypting data in-transit is deemed satisfied.

Before Encryption

<table>
<thead>
<tr>
<th>Request to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>/api/login/bankid/auth</td>
</tr>
<tr>
<td>POST /api/login/bankid/auth HTTP/1.1</td>
</tr>
<tr>
<td>Host: 192.168.40.1:3000</td>
</tr>
<tr>
<td>User-Agent: insomnia/5.16.2</td>
</tr>
<tr>
<td>Content-Type: application/json</td>
</tr>
<tr>
<td>Accept: <em>/</em></td>
</tr>
<tr>
<td>Content-Length: 26</td>
</tr>
<tr>
<td><code>{ &quot;ssn&quot;: &quot;190101010101&quot; }</code></td>
</tr>
</tbody>
</table>

After Encryption

<table>
<thead>
<tr>
<th>Request to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>/api/login/bankid/auth</td>
</tr>
<tr>
<td>POST /api/login/bankid/auth HTTP/1.1</td>
</tr>
<tr>
<td>Host: 192.168.40.1:3000</td>
</tr>
<tr>
<td>User-Agent: insomnia/5.16.2</td>
</tr>
<tr>
<td>Content-Type: application/json</td>
</tr>
<tr>
<td>Accept: <em>/</em></td>
</tr>
<tr>
<td>Content-Length: 26</td>
</tr>
<tr>
<td><code>{ &quot;ssn&quot;: &quot;190101010101&quot; }</code></td>
</tr>
</tbody>
</table>

Figure 18: Data captured in-transit before and after encryption while a user requests to authenticate.

To evaluate that data at-rest encryption is working properly some data was simulated, e.g. creating fake users like John Doe with social security number 190101010101 (any similarity to a real person is a coincidence). The contents of a database table in MariaDB is stored in a file on the hard drive with the file extension ibd. Inspection of some of
the file’s raw data is depicted in Figure 19. The inspected file, in this case, was the one that stored information about users in the system. See Appendix A.2 for the complete structure of table “users”. The left-hand side illustrates that personal data can be read in plain text, e.g. the social security number. After encryption, this is not possible without having the correct decryption key.

Before Encryption

<table>
<thead>
<tr>
<th>Filename: users.ibd</th>
<th>Data beginning at offset: 0000C080</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before Encryption</td>
<td>After Encryption</td>
</tr>
<tr>
<td>..€85bc2cf4-ef47-4484-b992-37f426539700...U”&quot;...!.190101010101JohnDoe.f08c78f8-04c0-46c0-9f3e-9e1dda7e4482.c“”Y0”“”Y0”...$...$...&quot;&quot;&quot;&quot;ma6474053-7ef1-4632-9ec9-3656e312b806...U»...f.190101010102JaneDoe.f08c78f8-04c0-46c0-9f3e-9e1dda7e4482.c“”Y0”“”Y0”...</td>
<td></td>
</tr>
</tbody>
</table>

Figure 19: Inspection before and after encryption of the raw data that includes the contents of the “users” table.

9.1.3 Authentication Method

The authentication method has three scenarios, that must be managed to be regarded as working: (i) successful/unsuccessful authentication, (ii) aborting authentication attempts, (iii) automatically opening the BankID application, when the patient mobile application is used.

All three scenarios were tested using the application BankID distributes to developers, allowing to test fake social security numbers for authentication as if they were real. Each essential scenario with five different mock social security numbers was tested and each time the system behaved as defined.
9.2 Effective and Efficient Patient-Practitioner Communication

An interview was conducted, according to the method described in Section 7.2.1, with the external stakeholder Karl Engström to evaluate if the system (i) will help alleviate the time spent on manually processing self-assessment questionnaires and (ii) will the practitioner be able to adjust better to a patient’s treatment with the help of the information gathered by the system.

Engström described that he felt that his attention was drawn to the automatic warning that appeared in the system [26]. He found that it was hard to comprehend what caused the warning to trigger. He suggested that opposed to the current text, “Drastic mood change”, the text could be more descriptive, e.g. “Total score changed from XX to YY” or mark the data in the chart that caused the warning. According to Engström, the warnings system could provide useful insight if the warnings are properly described and would contribute to a practitioner reacting faster to the changes of the patient’s well-being.

The charts illustrated in Figure 9 provided, according to Engström, a possibility to quickly form an idea of a patient’s well-being over time [26]. He noted that the chart of a questionnaire’s total result gave him explicit and useful information, but the chart containing the results of all the individual questions, seen in Figure 11, was too cluttered which made it difficult to read any information. He felt that it was a rather tedious task to deselect all individual questions to be able to only view one at a time. Despite this, Engström’s assessment is that undeterred by the tedious task of deselecting all questions, the feature of being able to view one of the answers of individual questions over time in a chart is very valuable since it makes it easier e.g. to evaluate whether a certain medication yields any effect or not.

Generally speaking, Engström said he believes that the system can improve how the practitioner perceives a patient’s well-being [26]. Not only because of how the charts enable the data to be easily digestible, but because the collection of data is moved to another part within the treatment, not at medical practices but inside the patient’s home. This enables a more continuous flow of information that will, in his opinion, better reflect the patient’s true state of well-being. His opinion aligns with the conclusions that Torous et al. found in their article [71].

The fact that the system can customise the questionnaires, enabling a practitioner to customise a questionnaire for each given patient as needed, is very valuable [26]. He said that it is valuable since questionnaires like MADRS-S are usually too general to apply to a specific patient’s problems.
To conclude, Engström believes that the presented system has a great potential to (i) help alleviate the time spent on manually processing data and (ii) help the practitioner to properly adjust the treatment of a patient [26]. He states that if data could be gathered more frequently and at home, the data should be a more accurate reflection of the patient’s state. The ability to customise the questionnaires for individual patients could be useful for both practitioner and patients and that the system could make self-assessment questionnaires more easily managed than the traditional method using pen-and-paper. Due to this, with the caveat of the mentioned bias in Section 7.2.1, the requirements are deemed to be satisfied.

9.3 Bridging the Gulf of Execution and Evaluation

To satisfy the requirement of usability as defined in Section 7.3 the system had to provide the user with the necessary guidance to be able to answer Norman’s seven questions of design. The seven questions of design explained in Section 2.4 are questions that the user must answer to successfully cross the Gulf of Execution (how the user reaches its goal) and the Gulf of Evaluation (how the user figure out what happened when trying to reach their goal).

9.3.1 Analysis Using Norman’s Criteria

A usability analysis has been performed of the system to ensure that a user of the system is able to answer the seven questions of design introduced in Section 2.4. The analysis was performed from the perspective of a patient and is given below.

1. What do I want to accomplish?

To start an action sequence the patient must realise what it is they want to accomplish. This can be difficult since it is usually not something the system can influence, e.g. a patient may be given the task to answer the MADRS-S self-assessment questionnaire by their healthcare practitioner, not the system.

The system can, however, help the patient realise what it is they need to accomplish. This has been done by push notification, that either once a day, every other day, every third day or once a week remind the patient to open the application. When the application is open, the patient is asked to authenticate themselves. While the system may be considered more usable without having to authenticate every time the patient accesses the system, it is necessary for the sake of security.
If the patient already answered the daily mood questionnaire and the MADRS-S questionnaire, there is nothing for the user to accomplish. Instead of displaying an empty page, making it difficult to answer “what do I want to accomplish?”, the system displays the page in Figure 20. This page informs the user that there is nothing more they can do which gives the user the answer to the question of what they want to accomplish: nothing.

![Figure 20: Informing the patient that no additional actions can be taken.](image)

When a patient has authenticated their identity, they are presented with the screen described in Section 8.1 and illustrated in Figure 6. The figure illustrates that the patient gets informed on what actions they can perform, which helps the patient realise what they want to accomplish. Thus, from a patient’s perspective, the question of “what do I want to accomplish?” is successfully answered.

### 2. What are the alternative action sequences?

After the patient has figured out what they want to accomplish, they must figure out the alternative action sequences. If the patient successfully realised that they need to answer the MADRS-S self-assessment questionnaire and they have the application open, they must be presented with actions, or affordances, to perform.
In Figure 6, a patient is presented with many perceived affordances. The affordances are perceived since there are necessary signifiers indicating where the action can take place, e.g. a button with the text “STARTA”. The left-hand side of the figure depicts signifiers for the affordances: (i) answering the question of how they are feeling today, (ii) start answering the questionnaire MADRS-S, (iii) sign out from the system and (iv) go to the settings page. The affordances (i) and (ii) are on the top and should be perceived before the other affordances.

If the user’s goal is to answer the self-assessment questionnaire MADRS-S, the alternative action sequences are easily identified since it is one of the perceived affordances mentioned above. Other scenarios, for example, if the patient’s goal is to change the frequency of the push notification, the alternative action sequences are not as clear since the signifier for configuring the notification frequency is not directly visible. At the same time, answering questionnaire’s is performed many times while configuring the notification frequency is most likely only set once to the desired setting. With this reasoning in mind, it is deemed that the system for the most common cases helps the user answer the question of what the alternative action sequences are.

3. What action can I do now?

Since the patient is presented with the alternative action sequences at the top of the screen as discussed in the second question of design it is possible for them to realise what they can do for the most common cases. One important aspect is that the system will never present an action that the users cannot perform. An example of this can be seen in Section 8.1 and Figure 20 where the figure informs the patient that there is nothing they can do.

This consistency of design helps the user to form a conceptual model of how the system works. They should realise, that all alternative action sequences are an action possible to perform. Consequentially, this implies that the question of what action the user can do are deemed to be answered.

4. How do I do it?

The question of how a user performs their intended action is answered with a good conceptual model. If the user realises how the system work, they should be able to navigate it more easily.

The system uses the Material-UI library, which brings a set of UI components that implement Google’s design language “Material Design” as further described in Sec-
tion 5.1. The reasoning behind this is to give the user a familiar environment, which will help the user apply conceptual models from other applications, that also implement the “Material Design” language to the system’s application.

The fact that there is an existing conceptual model applied to the application implies that the patient already knows how to, as an example, navigate between views using the bottom navigation bar seen in Section 8.1 and Figure 6 or how a clickable button looks in comparison to a disabled button.

The application of a conceptual model, together with the fact that common actions are placed at the top of the page, thus making them the most visible, helps the user realise how to do something.

5. What happened?

To help a user figure out what happened, appropriate use of feedback must be applied. The existence of feedback is not sufficient, but it should also provide guidance to the user such that they can make a proper interpretation of what happened.

Once again, a useful conceptual model is required. Since the Material-UI library is used, the system has a lot of feedback implemented by default, e.g. a button lighting up when pressed. The user should already have somewhat of a conceptual model of the system since they have a high probability of previous exposure since it is the recommended design language in Android [5], as described in Section 5.1.

Feedback should be implemented in more places than just the UI components. One example of this is the page where a user answers a questionnaire, which is illustrated in Figure 7. The figure illustrates inter alia a progress bar, indicating the user’s progress. Furthermore, on the right-hand side of Figure 6, the system explicitly states, that the user’s answers were correctly registered.

While these types of feedback help the user to answer the question of what happened, not only in UI level, e.g. “a button was pressed”, but also at a deeper system level, “the questionnaire was answered successfully”, but it does not help the user answer what did not happen. Error messages were only implemented during the authentication process and if something else in the system went wrong, e.g. the user’s internet connection is lost, no appropriate feedback was displayed.
6. What does it mean?

To further bridge the Gulf of Evaluation, a user must answer the question of “what does it mean?” in a response to something that happened. This question is closely coupled with the question “what happened?” but requires a deeper understanding of the system aided by the user’s conceptual model.

Despite the message on the right-hand side of Figure 6, which informs the user that an answer was registered, it is appropriate to help the user understand what happened, it does not help the user understand what it means. The patient may ask themselves if this means they cannot answer again? What if they change their mind?

To explain what something means, the user must form an appropriate conceptual model. While a model that explains behaviours in the system may form over time, it does not help explain to a first-time user what happened. A more detailed message could be added to further aid the user. It is deemed that the system does not aid the user enough in answering the question “what does it mean?”.

7. Is this okay? Have I accomplished my goal?

The last step to cross the Gulf of Evaluation is for the user to answer the question if what happened caused their goal to be accomplished. For some scenarios this is effortlessly achieved, e.g. if the user’s goal is to authenticate against the system and they are presented with the home page of the system, then the result is appropriate.

In other scenarios, messages should be presented informing that the action the user has performed was successful. An example of where this is applied is on the home page after a patient has registered an answer to a questionnaire. This is illustrated on the right-hand side of Figure 6. If the user’s goal was to answer a questionnaire and the message states that the user’s answer indeed was registered, then it is reasonable to assume that the user knows that they have accomplished their goal.

All actions in the system are supplied with a message like the one described above. This helps the user to answer the question if they have accomplished their goal.

9.3.2 Summary of the Usability Analysis

The result of the analysis is that the system helped the user to answer six out of the seven questions of design for the most common use cases. The only question unanswered were
question six “what does it mean?”. While the analysis does not cover all possible user scenarios it at least showcases that attempts were made to answer the questions.

If the attempts made to answer the seven questions of design is enough to bridge the Gulf of Execution and Evaluation cannot be concluded with a high degree of certainty. Nevertheless, the system was designed with gulfs in mind and more work should be done to ensure that the gulfs are bridged as much as possible.

10 Results and Discussion

The result is a system, where an administrator can create questionnaires, with questions that yield a quantifiable answer. A practitioner can assign a questionnaire to a patient, who can regularly answer the questionnaire. The patient can choose to be regularly notified and reminded to answer the questionnaire. The practitioner can view the results of the patient’s answers in a comprehensible manner, illustrated with the help of charts, that depict the patient’s state of well-being over time.

10.1 Regulation Compliance

As described in Section 2.3.1, a system that processes patient data must take regulations into special consideration. Using the handbook written by The National Board of Health and Welfare (Socialstyrelsen), we created a system that, in our opinion, conforms to the minimum requirements of the regulations. Since the implementation method is incomplete, it would be preferable to have a legal scholar to evaluate the compliance. The method was incomplete since all regulation considerations where based on the handbook, which gives no guarantee that all aspects of the law are covered.

10.2 Usability

As noted by the usability analysis in Section 9.3.2, six out of the seven Norman’s questions of design are deemed to be answered. The question that remains unanswered is “what does it mean?” as a response to something that happens. For this reason, it can be assumed that the system does not satisfy the requirement of usability as described in Section 7.3. This assumption, however, is not necessarily true, since if the patient’s conceptual model is aligned with the true model of the system, then the patient will have less problem answering the question.
There exist several potential solutions to help a patient more effortlessly answer the question “what does it mean?”. As previously mentioned in the analysis in Section 9.3.2, more information about each action performed could be presented to the user. Another solution would be to add a tutorial when a patient uses the system for the first time that thoroughly explains how the system works, thus more accurately aligning the patient’s conceptual model with that of the true system. Care must be taken to make the tutorial non-intrusive, since if the patient skips all steps the tutorial would yield nothing.

The usability of the system from an administrator’s or a practitioner’s perspective where not analysed due to the delimitations of the project and as a result, it is difficult to discuss the usability result of the complete system. It is presumed, that since all parts of the system were developed with Norman’s seven questions of design in mind, that several out of the seven questions could be answered by a practitioner and administrator. Additionally, the usability of the practitioner and administrator UIs is not deemed to be as important as that of the patient, since official training of the system could be given to employees of the healthcare practice utilising the system, unlike for the patient where this is not possible.

11 Conclusions

We have developed a system that enables patient-practitioner communication in-between appointments using an ICT tool. The system allows a user to authenticate and manages the user’s session. An administrator can manage questionnaires, a practitioner can register patients in the system and a patient can answer questionnaires. The answers are visualised such that the practitioner can make more informed decisions regarding the treatment of the patient.

As described in Section 4, there exists similar systems, such as BVC-Elvis, that allows users to answer questionnaires through an ICT tool, or Bipopular which visualises answers to the user’s self-assessment questionnaires. Regarding the inefficiencies identified in the patient-practitioner communication, other systems are lacking necessary considerations for solving that problem. We believe that our system implemented a more holistic approach to these specific problems, e.g. the ability to answer questionnaires and be reminded that a questionnaire is available, effective visualisation of the answers to a questionnaire and automatic warnings.

While the developed system is deemed to have the potential of making a positive impact on the treatment of patients, it has plenty of room for improvements. Engström, one of the external stakeholders of the project, noted in a personal interview, that the data visualised on the charts has the risk of becoming too cluttered [26]. Another pos-
sible improvement is to implement more detailed messages as a response when the user performs an action, to help the user identify what the action implies. There is also the possibility of providing a tutorial when a patient uses the system for the first time with the purpose to help the user to understand the system.

Some attempts were made to make sure that the system adheres to the regulations described in Section 2.3.1, although further investigation must be performed to guarantee compliance. If the system is to be used by healthcare practitioners and patients, a legal scholar should review the system.

12 Future Work

Further investigation of how the system can help practitioners make better decisions regarding a patient’s treatment is needed. Such investigation could include a trial that incorporates real patients and healthcare practitioners to answer this question in depth.

Features could be added to the system that could further improve efficiency and effectiveness in treatment, e.g. integration with the medical records system, that the practitioner use. Further efforts into improving the usability should be undertaken, as indicated by the usability analysis in Section 9.3.2

As noted in the delimitations in Section 3.1, integration with medical records could not be implemented due to time constraints. Engström mentioned, in a personal interview, that integration with quality records (Kvalitetsregister) may be one of the most exciting potential for a system like the one developed [26]. A quality record is a record with personally bound information, similar to the official medical records, but focused on a specific healthcare area, usually with the purpose of research but also as a compliment to the official medical records [1]. As it stands today, Engström’s perception is that it is cumbersome to register data into the quality records as well as later retrieving the data from the records when it is needed.

The system could be expanded to other users than adults with depression. Since the system allows any questionnaire to be created and assigned to a patient, it could be used in many, if not all, sections of healthcare.

Since the system was delimited to Swedish users only, a possibility of further expanding the system is to internationalise the system. Not only by providing translations of all text but also allows for other methods of authentication than BankID which is only supported in Sweden.
References


50


References


53
References


References


References


A Appendix

A.1 Results of Unit Tests

Below is a list of all the tests, the test runner ran against the API server. First listed is the module being tested preceded by the HTTP method used. Possible methods in the tests are GET, POST, PUT and DELETE. Where GET implies reading data, POST implies creating data, PUT implies updating data and DELETE implies removing data. If the test itself is preceded by “OK” the test passed. If it is preceded by a number it implies that the test failed. The test text itself, usually starting with “should” is the intended behaviour of the system.

GET /api/answer
- OK should return status code 401 when not logged in
- OK should return status code 403 when administrator
- OK should return status code 403 when patient
- OK should return status code 200 and the answers when practitioner

POST /api/answer
- OK should return status code 401 when not logged in
- OK should return status code 200 on correct answer

POST /api/assigned
- OK should not allow patient to assign questionnaire
- OK should allow practitioner to assign questionnaire
- OK should not allow admin to assign questionnaire

GET /api/log
- OK should have test users
- OK should return 401 when not logged in
- OK should only allow admins to view log

POST /api/login
- OK should successfully return token when giving correct password
- OK should fail to login when giving wrong password
- OK should fail to login when giving wrong ssn

/api/push
POST
OK should successfully update the users device id
OK It should return 401 when not sending token

DELETE
OK should successfully clear the users device id
OK It should return 401 when not sending token

GET /api/questionnaire
OK should return correct response when correct parameters
OK should return dailyMood false after you have answered dailyMood
OK should not have any available questionnaire after an answer
OK should have available survey after interval has elapsed

POST /api/questionnaire
OK should return 401 when not logged in
OK should return 403 when practitioner
OK should return 403 when patient
OK should return 200 when administrator

GET /api/questionnaire/:id
OK should correctly get the assigned and started questionnaire
OK should return 403 when questionnaire is not yet started
OK should return 403 when questionnaire is not assigned
OK should return 401 when not logged in

POST /api/users/legalAccept/:id
OK should have test users
OK should return 401 when not logged in
OK should only allow an admin to accept its own legal agreement
OK should only allow a practitioner to accept its own legal agreement
OK should only allow a patient to accept its own legal agreement
DELETE /api/users
   OK should have test users
   OK should not allow a user not logged in to delete a user, testing to delete a patient
   OK should not allow a user not logged in to delete a user, testing to delete a practitioner
   OK should not allow a user not logged in to delete a user, testing to delete an admin
   OK should not allow a patient to delete any user
   OK should allow a practitioner to delete a patient
   OK should allow an admin to delete a practitioner

GET /api/users/:id
   OK should have test users
   OK should return 401 when not logged in
   OK should not allow a patient to retrieve any user
   OK should allow a practitioner to retrieve any patient
   OK should allow an admin to retrieve any practitioner or admin but not a patient

GET /api/users
   OK should have test users
   OK should return 401 when not logged in
   OK should not allow a patient retrieve any user
   OK should allow a practitioner retrieve any patient
   OK should allow an admin retrieve practitioners and admins

PUT /api/users
   OK should have test users
   OK should return 401 when not logged in
   OK should not allow a patient to edit any user
   OK should allow a practitioner to edit any patient or admin
   OK should allow an admin to edit any practitioner
   OK should gracefully return an error on unknown user id

POST /api/users
OK should not allow a user not logged in to register a
new patient
OK should not allow a user not logged in to register a
new practitioner
OK should not allow a user not logged in to register a
new admin
OK should not allow a user register any user
OK should allow a logged in practitioner to register a
patient
OK should not allow a logged in practitioner to register
another practitioner or an admin
OK should allow an admin to create another admin
OK should allow an admin to create a practitioner
OK should not allow an admin to create a patient
OK should allow to create a user only with bankId

DELETE /api/warnings
OK should not allow an admin to remove any warnings
OK should not allow a patient to remove any warnings
OK should not allow a not logged in user to remove any
warnings
OK should allow a practitioner to remove any warning

GET /api/warnings
OK should have test users
OK should return 401 when not logged in
OK should not allow an admin to get any warnings
OK should not allow a patient to get any warnings
OK should allow a practitioner to get all users warnings

gETranslations
OK should get both swedish translations
OK should get english translation and fallback to key

logEvent
OK should be able to insert log entries
A.2 Entity–Relational Diagram

Figure 21: The relational diagram representing the database structure
A.3 List of Endpoints that Triggers a Log Entry

- POST /api/login
  - Success
  - User does not exist
  - Password is wrong
  - Missing parameters
  - Does not allow password login
  - User is not active

- POST /api/bankid/auth
  - Success

- POST /api/bankid/collect
  - Success
  - User does not exist
  - On BankID status=failed
  - User is not active

- GET /api/warnings
  - Success

- DELETE /api/warnings
  - Success

- GET /api/users
  - Success

- POST /api/users/legalAccept/:id
  - Success

- DELETE /api/users/:id
  - Success

- PUT /api/users/:id
  - Success

- POST /api/users
  - Success

- GET /api/answer
  - Success

- GET /api/questionnaire
  - Success

- GET /api/questionnaire/all
  - Success
A.4 List of Tasks

This is a translation from the original in Swedish. All names and social security numbers where generated at random and any similarity to a real person is a coincidence.

Note: All social security numbers used should be written as yyyymmddxxxx, i.e. no spaces or dashes.

Use the system as an administrator

1. Log in as an administrator (19821106 5357)
2. Examine a question in the MADRS-S questionnaire
3. Create a new questionnaire by:
   (a) Click on the "Lägg till enkät" button
   (b) Enter a questionnaire title
   (c) Add a question
       i. Add at least one possible answer to the question
   (d) Save
4. Add a new user: Marianne Axén, 19550825 7846 as user type practitioner
5. Log out

Use the system as a practitioner

1. Log in as an administrator (19540318 2081)
2. Add a new patient: Hildur Bergstedt, 19650819 3460
3. List of sub-tasks:
   (a) Assign the questionnaire you previously made as an administrator to the newly created patient
   (b) Assign MARDS-S to this patient
   (c) Save
4. Examine a patient with a warning
(a) Remove the warning

5. Examine Helena Edvinsson’s (19730704 8640) charts
   (a) List of sub-tasks:
      i. Remove all alternatives in the MADRS-S graph except one in order to
         examine an individual answer component

6. Log out

Use the system as a patient

1. Log in with the patient you previously created in the system (19650819 3460)
2. Answer how you are feeling today.
3. Answer the questionnaire you previously assigned yourself as a practitioner
4. Answer MARDS-S
5. Log out

Pincode to use for all logins with BankID: 14 25 36
A.5 List of Prepared Questions Asked during the Interview

This is a translation from the original in Swedish. The interview was conducted in Swedish.

Introduction questions:

1. What is your name and what is your occupation?
2. How are self-assessment questionnaires in healthcare currently managed?

Questions regarding the administrator’s page:

1. Could you successfully perform the tasks you were given?
2. Was there anything you thought was missing?

Questions regarding the practitioner’s page:

1. Could you successfully perform the tasks you were given?
2. What, if any, information did you get from the warning?
3. Do you see any utility with the warnings?
4. Do you see any problems with the warnings?
5. What, if any, information did you get from the charts?
6. Do you see any utility with the charts?
7. Do you see any problems with the charts?
8. Was there anything you thought was missing?

Questions regarding the patient’s page:

1. Could you successfully perform the tasks you were given?
2. Was there anything you thought was missing?
General questions:

1. How could this system be integrated into a healthcare facility?

2. Would it, if at all, change the work method in a healthcare facility?
   (a) In what way?
   (b) Why?

3. Do you think the system could change how well one perceive the patient’s well-being?
   (a) In what way?
   (b) Why?

4. Do you think the information the system provides, may help to adjust the treatment patient receives?
   (a) In what way?
   (b) Why?

5. In what way will the patient’s answers be affected by answering through a mobile application as opposed to with pen-and-paper?