A Comparison between Chatbots for Handling Administrative Healthcare Tasks

En jämförelse mellan chatbotar för hantering av administrativa vårduppgifter

FRED HALDÉN
JOEL YAO HÅKANSSON
A Comparison between Chatbots for Handling Administrative Healthcare Tasks

En jämförelse mellan chatbotar för hantering av administrativa vårduppgifter

FRED HALDÉN
JOEL YAO HÅKANSSON

Examensarbete inom Datateknik, Grundnivå, 15 hp
Handledare på KTH: Maksims Kornevs
Examinator: Ibrahim Orhan
TRITA-CBH-GRU-2020:006

KTH
Skolan för kemi, bioteknologi och hälsa
141 52 Huddinge, Sverige
Abstract

Chatbots are used in many fields to reduce the amount of repetitive work. While chatbots are proved to be great help in retailer business, tourism related services, it is little known about usability of chatbots in more sensitive field such as healthcare. A study was conducted to examine the possibilities whether a chatbot could be used to automate administrative tasks within the Swedish healthcare. Three chatbots were developed, each one using different framework. Each chatbot was tested with participants acting as a patient, freely using each chatbot, and filling out a quantitative evaluation after the test. Results from participants' responses to the quantitative evaluation and SWOT analysis showed that chatbots are helpful, user-friendly and have the opportunity to integrate with various databases and many services. Participants preferred the chatbot that could handle voice interaction. The results also showed that the chatbots were helpful in registering cases and answering frequently asked questions. Additionally, it was observed that chatbots are efficient when their work is complemented by a healthcare specialist. This allows concluding that a chatbot is a promising potential tool to automate administrative tasks within the Swedish healthcare.

Keywords
Chatbot, healthcare, SWOT analysis
**Sammanfattning**


**Nyckelord**
Chatbot, sjukvård, SWOT-analys
Acknowledgement

This report was the result of a bachelor thesis that was conducted within Computer Science and Engineering at the Royal Institute of Technology.

We want to thank everyone at Patientnämndens förvaltning in Stockholm, for testing and evaluating the chatbots. Most especially we would like to thank David Waleh and Gisela Rosenquist for providing us with information regarding the Patientnämndens förvaltning, the Swedish healthcare and follow up meetings regarding the chatbot implementations.

We would like to thank our mentor Maksims Kornevs for his support and advice during this thesis.
List of acronyms and abbreviations

AI - Artificial Intelligence
AIML - Artificial Intelligence Modelling Language
ASR - Automatic Speech Recognition
GDPR - General Data Protection Regulation
ML - Machine Learning
NHS - National Health Service
NLP - Natural Language Processing
NLU - Natural Language Understanding
PdL - Patient data Law
TTS - Text to Speech Synthesis
# Table of content

1 Introduction.................................................................................................................. 1

1.1 Limitation of Artificial Intelligence in Healthcare .............................................. 1

1.2 Research aim............................................................................................................. 2

1.3 Objective.................................................................................................................... 2

1.4 Delimitations............................................................................................................. 2

2 Theory and background ............................................................................................... 3

2.1 Healthcare analysis ................................................................................................. 3

2.2 Patientnämndens förvaltning ................................................................................... 4

2.3 Chatbot architecture ............................................................................................... 4

2.4 Machine Learning.................................................................................................... 5

2.4.1 Natural Language Processing (NLP)................................................................. 6

2.5 Chatbot frameworks .............................................................................................. 6

2.5.1 Botpress............................................................................................................... 6

2.5.2 Microsoft Bot Framework .................................................................................. 7

2.5.3 Dialogflow .......................................................................................................... 8

2.5.4 Pandorabots ....................................................................................................... 8

2.5.5 Wit.AI ................................................................................................................ 9

2.6 Test and evaluation methods of bot frameworks .................................................... 9

2.6.1 Input validation .................................................................................................. 9

2.6.2 Quantitative and qualitative research............................................................... 9

2.6.3 Strength, Weakness, Opportunity and Threat (SWOT) Analysis ..................... 10

3 Method and result ....................................................................................................... 11

3.1 Selection of chatbot tasks ....................................................................................... 11

3.2 Chatbot development methods .............................................................................. 11

3.2.1 Develop Botpress chatbot ............................................................................... 11

3.2.2 Develop Microsoft Bot Framework chatbot ...................................................... 12

3.2.3 Develop Dialogflow chatbot ............................................................................. 13

3.2.4 Develop Pandorabots chatbot ......................................................................... 13

3.2.5 Develop Wit.ai chatbot ................................................................................... 14

3.3 Choice of frameworks ............................................................................................ 15

3.4 Implementation results .......................................................................................... 17

3.5 Evaluation process .................................................................................................. 19

3.6 Evaluation results .................................................................................................... 19

4 Analysis and discussion ............................................................................................. 21
4.1 Analysis of non-technical aspects ......................................................... 21
4.2 Analysis of feedback from evaluation ..................................................... 21
4.3 SWOT analysis of Botpress framework ................................................... 22
4.4 SWOT analysis of Microsoft Bot Framework ............................................ 23
4.5 SWOT analysis of Dialogflow framework ............................................... 25
4.6 SWOT analysis of all frameworks .......................................................... 26
4.7 Discussion .................................................................................................. 27
  4.7.1 Prototypes effectiveness ................................................................. 27
  4.7.2 Prototype optimizations ................................................................. 28
  4.7.3 API:s relationship according to the theoretical model ....................... 28
  4.7.4 Chatbots performance in predefined use-case scenarios ..................... 28
  4.7.5 User communication with the chatbots ............................................. 28
  4.7.6 Chatbots and use of audio .............................................................. 29
  4.7.7 Response speed of the chatbots ...................................................... 29
  4.7.8 Benefits and consequences of an implementation of chatbots in administrative healthcare tasks .............................................................. 29
  4.7.9 Chatbots ability to complement a staff member of the healthcare ........ 29
5 Conclusions ............................................................................................... 31
References ...................................................................................................... 33
1 Introduction

A chatbot is a conversational software that communicates with users in a written or spoken natural languages. They are used for a wide range of use cases, such as to offer support to customers for purchasing, booking or retrieve information about a specific product, personal assistants, information retrieval agents and more.

In general a chatbot is a conversational agent which is designed to simulate an intelligent human dialogue in a natural language [1]. The agent provides a conversational interface which allows users to type in input in different formats, such as in text, voice, web documents, etc. The input is then processed and triggers an event or pattern that generates an action or response message based on the context and the input of the conversation.

Artificial Intelligence (AI) refers to human intelligence within machines, in which machines mimics human-like behaviour and actions in achieving a specific goal [2]. AI is the backbone engine behind a chatbot, which uses machine learning algorithms such as Automatic Speech Recognition, Natural Language Processing and Natural Language Understanding to extract input and make it understandable for the computer to take responsive action.

Machine Learning algorithms has drastically enhance chatbots AI performance [3]. It has given arise to highly sophisticated bot framework to automate many tasks within many different fields of areas. However in practise the research is still lacking in many fields. Such as within public healthcare.

1.1 Limitation of Artificial Intelligence in Healthcare

AI research has been on the rise since the last decade and they are currently expanding at an increasing rate within many different fields of areas, such as in retail business, tourism related business, finance, and within many private organizations. However there are many fields of areas where automatization and AI research is lacking, one particular field is in healthcare [4].

The public healthcare in Sweden is characterised with high workload, long queue for patients, and shortage of staff [4]. According to the survey “Doctors: Worse healthcare today then it was ten years go” 44% of the doctors that were asked considered that the healthcare has deteriorated since the last decade [5].

In the survey the most contributing factors for this deterioration is the lack of accessibility in which, the shortage of nurses and fewer hospital facilities has increased the waiting time for patients to receive their medical aid and doctor’s opinions. The most contributing factor stated in the article however is the increased administrative tasks, that has put more work on the already high workload per staff member within a healthcare provider [5].

There exists a need to automate parts of their workload to reduce the staff’s daily administrative tasks and increase the efficiency for the healthcare establishment. One technology within AI research and automatization which is suitable to address this issue is chatbot development.
1.2 Research aim
The aim of this research is to investigate the possibilities of introducing a chatbot as a tool within the healthcare system, and to discover the potential areas of use they could have to reduce the workload burden for healthcare employees.

This research aim has following sub-questions:

- How well do the chatbots perform in predefined use-case scenarios?
- How should input be validated for the chatbot?
- How can the user communicate with the chatbot?
- Can the chatbot understand and use a sound voice?
- How fast does the chatbot respond to user input?
- What impact does a chatbot have from a non-technical perspective (social perspective, ethics and ecological sustainability, etc)?
- What are the benefits and consequences of a chatbot?
- Is the chatbot a relevant tool to complement a staff member of the healthcare?

1.3 Objective
In order to reach the final conclusion for this research aim, the objective of this thesis is to develop and evaluate three chatbots with the purpose of being able to produce an effective prototype and give answers to this research aims sub-questions. Each chatbot will use a different bot framework. The evaluation will consist of different test cases and of a quantitative investigation based on a questionnaire which will be provided to a test group. The effective prototype should have the intention of reducing the workload in the healthcare and provide answers to this research’s aim sub-questions, which will be answered later in the report to reach the final conclusion for this thesis.

1.4 Delimitations
This work is only delimited to the cost-free version of each bot framework, therefor not all functionality could be tested and evaluated. The chatbots will not be using e-identification or bank-id for authentication. Only some free frameworks will be tested and evaluated due to time constraints of the project. Work is tested not by patients, but by expert group.
2 Theory and background

This section will give a brief overview of what administrative challenges the healthcare faces today and how the healthcare address these issues, followed by a description of chatbot architecture; how they are implemented and key aspects of them. This chapter will also describe how chatbots could be evaluated.

2.1 Healthcare analysis

Many hospitals are facing several issues with long waiting lists for their patients, shortage of staff and an increased burden of workload per staff member, which have resulted in a negative patient satisfaction and worse healthcare [5].

The Patient Data Law was introduced by the Swedish government in 2008, states which information is to be recorded within a patient’s journal and how the treatment is done. PdL states the rights for patient to access their documentation and also regulates which people who will get access to these documents [6].

General Data Protection Regulation is an EU integrity law to ensure patient’s personal integrity by regulating the personal data processing. GDPR has put higher demand on healthcare providers on the handling of personal data [7].

Between the years 2010-2017 in Sweden, the administrative workforce within the healthcare system, such as managers and administrators positions increased with 35,87% and the healthcare staff, such as nurses increased with 1,97%, biomedical analyst decreased with -1,11%, midwife increased with +5,27% [8].

The correlation between the statistics show that more time is focused on administrative tasks, rather than on the treatments of patients. In some healthcare facilities, doctors are spending a third of their working time treating patients, and the rest on administration tasks [9]. Therefore, there is an increased need for automatization to handle these tasks and reduce the workload amongst healthcare providers staff members [4].

Automatization within the healthcare development has introduced many technologies, such as virtual assistants, doctors and chatbots [10]. The rapid growth of technology advancements have increased the possibilities for automatization of many tasks in different areas within the healthcare, such as case handling, patient registration and patient-doctor interaction [4, 11].

The concept of chatbot technologies within healthcare has proven to be in some countries successful, such as in England. The chatbot Babylon was introduced by the National Health Service, where it could converse with a patient to verify if they needed to seek treatment for their symptoms [12].
In Sweden there have been a few studies investigating and evaluating the potential of automatization and chatbot development within healthcare. For example:

- Patients could be pre recorded before their arrival to a specific healthcare provider [4].
- Chatbots or virtual assistants could act as the first line of defence in healthcare to make diagnoses and recommend treatments for patients [4].
- Chatbots could checkout patients automatically with regard to medication, reportings and redistributions of patients to other healthcare facilities [4].
- Have a chatbot to answer patient’s healthcare related questions [5].
- Have a chatbot to handle the process flow of case registration within a healthcare facility [5].

According to Swedish Association of Local Authorities and Regions (SALAR) 36% of work within various fields in healthcare could be automated with the use of AI technology and chatbots [4]. With the outcome to relieve the work of healthcare employers and increase patients satisfaction. Another advantage is that new occupations is needed to create, construct and manage these systems [4].

Chatbots are considered to have a huge beneficial profit, since they can process information faster than human capabilities with the outcome of less margin of error in decision making, independent from human emotions. Chatbots are also available around the clock to perform their designated tasks without manual interference [4].

The high workload burden of administrative tasks for healthcare staff members could result with working overtime, with induced stress that could leads to sick reporting and result with worse quality of healthcare services. In which an implementation of chatbot could minimize the risk for that scenario [13].

However the use of chatbots within healthcare is limited due to many reasons, such as how chatbots should handle sensitive patient data according to PdL and GDPR. More research is needed to achieve a better knowledge base within chatbot development and their usability within the Swedish healthcare [4].

### 2.2 Patientnämndens förvaltning

There are many healthcare providers in Sweden that faces the challenges that are described in section 2.1.1. Patientnämndens förvaltning handles most of the administrative work related to healthcare.

Patientnämndens förvaltning is an independent and impartial instance that helps patients and their relatives to solve problems that have arisen in contact with healthcare and dental care, e.g. deal with deficiencies in care, treatment and communication [14].

Their work is characterized with heavy workload of administrative tasks and documentation of patients. A lot of their work could be automated with the use of a chatbot [14].

### 2.3 Chatbot architecture

A chatbot could be divided into two model groups: retrieval-based model and generative model [15]. The retrieval-based model uses a repository of predefined messages that matches them to an appropriate response message based on the input and context of the conversation. Figure 1 describes the retrieval-based model [15].
The heuristic for selecting a response could be implemented as either rule-based or with the use of ML classifiers [15].

In the rule-based approach the chatbot generates responses based on predefined rules. The advantage with this approach is that chatbots is able to handle simple queries, but becomes more prone to error when handling more complex tasks [15].

Unlike the retrieval-based model, the generative model is not dependent on a storage of predefined messages. They use ML algorithms to translate each inputted word and analyze previous messages to generate a response message [16]. Figure 2 describes the generative model.

**2.4 Machine Learning**
ML is a subfield within AI that explores and develop algorithms for systems to use in learning and making better outcome prediction on data [17]. These algorithms are based on a mathematical model which takes input data (also known as training data) into considerations of making data-driven predictions or action for system to use in completing a specific task, without the need for rule-based pattern or manual interaction [18].
There exists many ML algorithms to enhance the AI performance of a chatbot, most commonly used ML algorithms in chatbots are NLP and NLU.

2.4.1 Natural Language Processing (NLP)
NLP is a term within computer science and AI which tries to explore and define the communication between a computer program and human user in a natural language [19]. It is an essential part of how a chatbot learns to process input from a human user, and take a suitable action or give a response based on it.

NLU is a subset of NLP that goes in more-depth of understanding the human input, to extract unstructured data from human input and convert them to structural form which a computer can understand and act accordingly [19].

2.5 Chatbot frameworks
This section will list some chatbot frameworks that could be used for this research aim.

A chatbot is generally built up of a frontend, graphical user interface accessed with a computer or a smartphone, and a backend, servers that handles information used for frontend functionality, with middleware to couple the two together. An example of this can be seen in Figure 3.

![Figure 3. Theoretical model of a chatbot](image)

2.5.1 Botpress
Botpress is an open-source framework written in Javascript. It is powered by built-in modules such as QnA, Analytics, Flow editor and NLU to build conversational chatbot for any system. A Botpress chatbot could be built, deployed and managed either locally or at any server, with the use of the built-in connectors or through public API for the preferred platform [20].

A chatbot developed with the use of Botpress framework will receive messages from a user through a channel, it then processes the messages for the chatbot to understand and translate them to structured data. The chatbot then decides which action to take and generate a response back to the user through the module Dialog Manager. Figure 4 describes how the Botpress chatbot process user input.
The Botpress NLU module will process every incoming messages with the use of intent classification, language identification, entity extraction, slot tagging and language understanding [20].

Intent classification helps the chatbot to understand what the user is trying to say. With the use of entity extraction the chatbot could extract structured information from phrases such as dates, time, cities, names, etc. Slots is another major concept in the Botpress NLU module. A slot is a necessary parameter to complete an action that is associated with an intent. With the use of slot tagging, the chatbot identifies necessary parameters to fulfill a given task [20].

Once the user input has been processed and transformed into structured data for the chatbot to understand, the module Dialog Manager determines what the chatbot should say or which action to take [20].

Botpress has support for many third-party databases and messengers channels to integrate with and has a built-in emulator for the developer to use for debugging and troubleshooting purposes. Botpress framework comes with two editions, the community version which is free to use, but with restricted number of bots and less functionality then the pro edition which costs $49.95 per month [20].

Some prerequisites for development with Botpress are Botpress-v12_0_1 or higher, a Javascript IDE, a web browser and an email account. A basic knowledge of Javascript programming is also preferred [20].

2.5.2 Microsoft Bot Framework

Microsoft Bot Framework was developed by Microsoft to build conversational chatbots for enterprises. The framework consists of Bot framework SDKv4, which enables developers to model conversations and build bot applications using .NET [21].

The bot framework has support for Javascript and C# programming language and could be deployed on many platforms, such as Azure Bot Service, Heroku, on websites and other servers [21]. With the Azure Bot Service many built-in tools is included to make it easier for the developer to build and deploy a chatbot on many channels, such as Skype, Cortana, Facebook, Messenger, etc [21].

The framework has support for speech and intents understanding, however the framework does not have have built-in NLU functionality, it is included in Language Understanding Intelligent Service (LUIS), which is another bot framework developed by Microsoft.
LUIS interprets user intentions in many natural languages for a variety of use cases [21].

Microsoft Bot Framework comes with two editions, a standard version which consists of a free tier with unlimited messages on standard channels (Cortana, Skype and Microsoft Teams) and for premium channels, the message quota is limited to 10000 messages per month. The S1 edition comes with unlimited messages for standard channels and $0.50 per 1000 messages on the premium channels.

Some prerequisites for development with Microsoft Bot Framework are Visual studio 2017, Bot framework SDKv4 template for C#, Bot Framework Emulator and a web browser. Knowledge of ASP.Net Core and asynchronous programming in C# is also needed [22].

2.5.3 Dialogflow
Dialogflow (formerly known as API.AI) is a bot framework owned by Google which provides human-computer interaction in a voice or text-based conversational interface, which is powered by AI [23].

The framework has support for intents understanding, entity extraction and speech recognition to build conversational interfaces for websites, messaging platforms and mobile applications. The framework also has a built-in emulator for the developer to troubleshoot and debug a chatbot and provides SDK:s for Java, Python, Javascript and many more programming languages [23].

Dialogflow has two editions, the standard edition which provides unlimited text messages for the chatbot and Google assistant, with a limit of user request quota for speech recognition and a limited amount of knowledge connectors. The Pro edition offers text messages at the cost of $0.002 per request and $0.0065 per 15 sec of audio request [23].

Some prerequisites for development with Dialogflow are a web browser and a Google account. A basis knowledge of Javascript programming is preferred [24].

2.5.4 Pandorabots
Pandorabots is an online web service for building and deploying chatbots. The Pandorabots platform supports the open standard script language Artificial Intelligence Markup Language (AIML) to act as the knowledge base and partially the behavior of the chatbot.

Pandorabots does not include any machine learning algorithms to process human input. The developer must use AIML to define the data sets for the rules of communication between an user and the chatbot [25].

AIML uses pattern based communication, and stores and matches user input with chatbot responses in tags. AIML tags is small knowledge building block which contains the AI of the chatbot. The most important tags are <category>, <pattern> and <template>. The category contains rules to map user input stored within the pattern tag with an response stored in the template tag [25]. The figure below describes how an user question and a chatbot response could be stored in AIML.

```xml
<category>
    <pattern>Hello, how are you doing </pattern>
    <template>Hi, I'm doing fine, thank you for asking.</template>
</category>
```

Figure 5. How a user question and the chatbot response is stored in AIML.
The platform offers SDK:s for Java, Ruby, PHP and Javascript programming and could be integrated with many third party services and channels which includes voice and messaging platforms such as Messenger, Kik, Telegram and Twitch [25].

The Pandorabots framework is free to use in any projects, but has a limited amount of request to 1000 messages per month. The developer edition costs $19 per month with a limit of 10000 messages per month [25].

Some prerequisites for development with Pandorabots are a web browser and an email account registered at Pandorabots website. A basis knowledge of chatbot implementation and programming language which Pandorabots support to build a chatbot [25].

2.5.5 Wit.AI
Wit.AI is an free open-source bot framework developed by Facebook, to build text or voice conversational chatbots. The framework could be integrated with custom made websites, web and mobile applications and with any platform that has an public API [26].

The framework has built-in NLP features to understand intents, entities extraction, actions, context and has support for over 137 languages, to build and deploy chatbot for many channels and platforms. The framework also provides SDK:s for Javascript, Python, Ruby and HTTP API [26].

Some prerequisites for development with Wit.ai are a web browser, a Facebook or Github account and a basis knowledge of chatbot implementation and programming language which Wit.ai support to build a chatbot prototype [26].

2.6 Test and evaluation methods of bot frameworks
This section will describe some of the test and evaluation methods that could be used to test and evaluate bot frameworks.

2.6.1 Input validation
To test the general input the chatbot needs to handle a unexpected request from the user and send the appropriate response. The unexpected request from the user consists of random information that a human would say and the appropriate response should be more specific to human behaviour. This behaviour can be tested through human interaction, when a user interacts with the chatbot and gives feedback after usage on what could be done better, and by using a framework that can communicate with the chatbot to be tested, validating the conversation data with or without a unit testing framework [27-28].

For specific subjects the pyramid approach is more appropriate testing approach. In the pyramid approach, input is validated by validating the different outcomes when a user replies in a process of answering a specific set of questions [28].

2.6.2 Quantitative and qualitative research
Quantitative research is the process of collecting, analyzing and decipher data with the purpose of explaining the relationship among variables being tested by the researcher. These variables can be measured with instruments in the sense that numerical data can be analyzed with the use of statistical methods [29]. The data collected by the researcher for a quantitative observation must be filtered through the researcher’s objective standpoint, in order to achieve higher precision in the measurements [30]. Quantitative observations are typically close-ended in that the researchers ask questions to the participants that only have predetermined answers, answers such as yes, no or some other alternative [29].
A qualitative observation is when the researcher takes field notes on the behavior and actions of participants at the research site. In the notes, the researcher documents in a either an unstructured or semistructured approach the activities at the site of observation. Using some prior questions that the inquirer wants to know. Qualitative observers and the researcher may partake in different types of roles to collect more data and make more assessments. Qualitative observations are typically open-ended in the sense that researchers ask general questions to the participants and allow the participants to freely provide their opinions and views [29].

2.6.3 Strength, Weakness, Opportunity and Threat (SWOT) Analysis
SWOT analysis is a strategic planning technique that provides assessment tools used for assessing the performance, competition, risk, and potential of a business or a part of a business such as a product line or division, an industry, or other entity. Identifying core strengths, weaknesses, opportunities, and threats lead to fact-based analysis, fresh perspectives and new ideas. The SWOT analysis technique can guide businesses toward strategies more likely to be successful, and away from those in which they have been, or are likely to be less, successful. An independent SWOT analysis analysts, investors or competitors can also guide businesses on whether a company, product line or industry might be strong or weak and why [31].
3 Method and result

This section describes the methods used for the choice of frameworks, the development and evaluation. This section also describes the results of the development process and evaluation process.

3.1 Selection of chatbot tasks
In collaboration with Patientnämndens förvaltning, information that was needed to create a chatbot was collected during the first few meetings. The information that was needed to identify a case in the chatbot is described as follows:

- Healthcare business, consists of at least one healthcare unit. E.g. primary health care, emergency hospital, dental care, psychiatric care and more.
- Healthcare unit, consists of at least one healthcare department. Health care units specify the county or hospital.
- Healthcare department, consists of specific departments in the healthcare unit e.g. medical department.
- Description of complaint.
- Anonymity, the user can specify yes or no.
- Full name, if no anonymity.
- Recontact, the user can specify yes or no.
- Phone number, only numbers in the format 07xxxxxxx, or e-mail if user wants to be recontacted.

Once the information was gathered to handle the healthcare related tasks, a knowledge base could be established for the chatbots to use as training data to handle the specific tasks. The necessary information to create the chatbots can be found in appendix.

3.2 Chatbot development methods
This section describes the methods that can be used to develop the chatbots with different bot frameworks.

3.2.1 Develop Botpress chatbot
The information that was gathered from the data mining process could be used as training data for the chatbot and translated into intents and entities within the NLU module. The data is then stored in the framework’s internal database SQLite.

Intents in Botpress chatbot are named after the intention of what the user wants, e.g register a case, in which the name of the intent could be named “register_a_case”.

An user could ask various phrases such as to how to register a case, where every synonym phrases would be stored in the intent “register_a_case”.

Entities store names of healthcare businesses, healthcare units and healthcare departments to slot the variables when a user is communicating with the chatbot.

The built-in function slot-tagging tags keywords within intent, such as “I would like to register a case within a healthcare establishment”, where the words “healthcare establishment” would be tagged as a slot.

The function of slot tagging is to map keywords within intents to an entity, in which an user could directly register a case within a healthcare establishment, instead of going through a step-by-step process of registering a case with the chatbot.
An user task could be divided into smaller tasks with the use of the built-in module Flow editor in which a chatbot could be divided into multiple flows to handle different scenarios of a task. A flow consists of several nodes, where each node executes a sequence of instructions.

An intent could represent the transition of nodes depending on the intention of the user. Such as if the user would like to register a case within a specific healthcare business, redirect the complaint to a specific healthcare department or unregister a case, etc.

To test the chatbot the built-in chat emulator could be used to simulate different use case scenarios. Figure 6 describes the architecture of how a Botpress chatbot communicates with an user.

3.2.2 Develop Microsoft Bot Framework chatbot
The Microsoft chatbot consists of a GUI, a middleware and a Microsoft SQL database. The GUI consists of Microsoft Bot Framework emulator. The middleware is developed mostly in C# to couple the GUI with the database using Microsoft Bot Framework SDK, Microsoft Bot Builder SDK and its dialogs.

The database has tables for user phrases, bot phrases, contexts, bot names, healthcare businesses and healthcare units. The figure below displays a ER model of how some of the tables for the database are organized.

Other tables, not shown in the figure, for healthcare businesses and healthcare units each have their own tables providing a list. Tables for each healthcare business provide a list of healthcare units it consists of e.g. the table for emergency hospitals contains a list of
emergency hospitals. Tables for each emergency hospital contains a list of departments in the emergency hospital. The table for bot names contains a list of names the bot can use to pretend to be human rather than saying it is a bot from the startup. The tables make changes more dynamic for future changes as most changes need to be made in the database and not the middleware. The figure below displays the overview of the chatbot.

![Figure 8. Overview of the Microsoft Bot Framework chatbot. Microsoft Bot Emulator communicates with middleware coded in C# that communicates with a MSSQL Database.](image)

### 3.2.3 Develop Dialogflow chatbot

The Dialogflow chatbot consists of an integration with Google assistant and features that Dialogflow provides. Intents are named after the intention e.g. when a user asks “What is your name?” the intent would be named “what_is_your_name”. Intents store the different ways of asking and answering a question.

Contexts can separate the intents from each other so that the user can’t break out of the conversation whenever possible and a different outcome can be made depending on what the user sends to the chatbot.

Events can send a message to the user when the bot needs to e.g. a welcome message appears when the user first starts the conversation, an event is triggered at start of conversation to make the chatbot send a welcome message and make the user feel welcome.

Entities store names of healthcare businesses, healthcare units and healthcare establishments to slot the variables when a user is communicating with the chatbot.

The built-in fulfillment feature is used to give more dynamic responses e.g. when the user is asking if the hospital is open, Dialogflow then checks using its built in fulfillment feature to look up if the hospital is open or not and sends back an appropriate response back to the user. The figure below displays the overview of the chatbot.

![Figure 9. Overview of the Dialogflow chatbot. A Google assistant device communicates with Dialogflow servers.](image)

### 3.2.4 Develop Pandorabots chatbot

The chatbot with the use of Pandorabots framework requires knowledge within AIML script language to define the communication between the chatbot and the user. The information
that was gathered from the data mining process could be stored within AIML to build up the knowledge domain for the chatbot.

A Java chat application could be developed and integrated with the chatbot hosted at Pandorabots platform with the use of Pandorabots API.

To be able to setup a Java application the developer needs an account on developer.pandorabots.com and require an user key which is received once the account has been created and the development licence has been paid.

To communicate with the chatbot it is required to for the developer to use Talk API together with the user key. For every API request to the chatbot, the application id, botname and userkey is required parameters to use within the request.

The user input is then processed in which the chatbot checks if keywords or if the whole phrase is described in the AIML script. If the chatbot could not process the message it sends back a predefined error message. Figure 10 describes the communication of how an API request is made to the chatbot by the Java chat application.

Figure 10. The architecture of how an API request is made to a Pandorabots chatbot.

### 3.2.5 Develop Wit.ai chatbot

In the Wit.ai chatbot, the user communicates with Wit.ai through a custom made GUI in Java that processes the reply from Wit.ai using a database, where all phrases are stored. The Java application uses HTTP API to send a message to Wit.ai and extract a reply from Wit.ai. The Java application processes the words extracted in the reply from Wit.ai using a database to collect IDs of phrases. When the IDs are collected the values are then checked in the database for a relevant reply that is sent to the custom made GUI to be displayed by the user. A client token is used for identification and access to the Wit.ai chatbot with HTTP API. The figure below displays the overview of the chatbot.
3.3 Choice of frameworks

In order to make a suitable choice of framework for this work, an investigation needed to be done to examine which bot frameworks offer the best functionality to handle the tasks. The investigation can be performed by e.g. using multi-criteria analysis, performance and unit tests of different frameworks and previous finished surveys of different bot frameworks. There are many different frameworks that can be investigated for this purpose. Due to time constraints, the frameworks investigated are Botpress, Microsoft Bot Framework, Dialogflow, Pandorabots and Wit.ai.

Multi-criteria analysis was performed with the help of Patientnämndens förvaltning. All parties agreed that the most important aspect is security and cost of chatbot framework. Other aspects with highest weight were related to a need to be developer friendly in order to make simple prototypes to handle the test user tasks. These aspects are choice of programming language, supported communication languages, choice of databases, choice of machine learning algorithms, communication channels and availability of emulator. A comparison of Bot frameworks functionality and pricing which were evaluated can be found in Table 1.

The most developer friendly frameworks in which it required minimal coding to build an effective prototype were Botpress, Dialogflow and Microsoft Bot Framework. Pandorabots and Wit.ai had a larger learning curve, in which more time were needed to develop the chatbots, build the test cases and integrate them with an external database. All bot frameworks offered a cost-free version, however more options were available with Botpress, Dialogflow and Microsoft Bot Framework. Pandorabots free version offered limited functions, such as lack of integration with 3rd party channels and a reduced amount of channel messages which made it difficult to test and troubleshoot the chatbot. The Wit.ai solution was too complicated and was better without Wit.ai. Wit.ai divides a sentence into keywords and sends a reply back with those keywords. The process could be done easily without the framework in a programming language. Due to these reasons, Wit.ai and Pandorabots were not chosen for this research project. Patientnämndens förvaltning uses mostly Microsoft products in their systems which resulted in Microsoft Bot Framework being more preferred option than other frameworks.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Botpress</th>
<th>Microsoft Bot Framework</th>
<th>Dialogflow</th>
<th>Pandorabots</th>
<th>Wit.ai</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Speech recognition</strong></td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Pricing</strong></td>
<td>Community - Free, Pro - $49.95/month</td>
<td>Standard version - Unlimited messages on standard channels, up to 10000 messages on premium channels S1 edition - Unlimited messages on standard channels, $0.50 per 1000 message on premium channels.</td>
<td>Standard version - free, Enterprise Edition - $0.002 per text message and $0.0065 per 15 sec of audio requests</td>
<td>Sandbox edition -1000 messages per month, Developer edition - 19$/month</td>
<td>Free</td>
</tr>
<tr>
<td><strong>Programming languages</strong></td>
<td>Node.js, Botpress SDK</td>
<td>C#, Node.js, .NET</td>
<td>Javascript, Python, Java, Go, Ruby, C#, PHP</td>
<td>Java, Ruby, Go, PHP, Python and Node.js</td>
<td>Node.js, Ruby, HTTP API</td>
</tr>
<tr>
<td><strong>Languages</strong></td>
<td>Arabic, English, French, German, Japanese, Portuguese, Russian, Spanish,</td>
<td>Supports all languages if locale is setup correctly</td>
<td>Chinese, Danish, Dutch, English, French, German, Hindi, Indonesian, Italian, Japanese, Korean, Norwegian, Polish, Portuguese, Russian, Spanish, Swedish, Thai, Turkish, Ukrainian</td>
<td>183 Languages</td>
<td>137 Languages</td>
</tr>
<tr>
<td><strong>Databases</strong></td>
<td>SQLite 3, PostgreSQL, Third-party databases</td>
<td>Any database that works in C#</td>
<td>Google Cloud and any other database that can be integrated with fulfillment</td>
<td>Artificial Intelligence Mark-up Language (AIML)</td>
<td>Wit.ai database</td>
</tr>
<tr>
<td><strong>NLU/NLP</strong></td>
<td>Yes</td>
<td>No (Yes - With LUIS)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Channels</strong></td>
<td>Facebook Messenger, Web chat, Bing, Cortana, Email, Facebook Messenger, GroupMe, Microsoft Teams, Skype, Telegram, Twilio, Web chat</td>
<td>Facebook Messenger, Slack, Viber, Twitter, Twilio, Skype, Telegram, Kik, Line, Cisco Spark, Microsoft Cortana, Google Assistant</td>
<td>Voice and messaging platforms</td>
<td>Voice and messaging platforms, IoT devices</td>
<td></td>
</tr>
<tr>
<td><strong>Emulator</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 1. Summary of Bot frameworks functionality and pricing which were evaluated*
3.4 Implementation results

The chatbot with the use of Botpress framework was developed using Javascript and the built-in modules QnA, Flow editor, NLU and the built-in emulator to simulate a conversation between a user and the chatbot. Two screenshots of the implementation can be seen in Figure 12.

![Figure 12. Two screenshots of Botpress chatbot](image)

Dialogflow chatbot was developed on the Dialogflow webpage and Javascript programming for fulfillment to enrich the human-like experience for the chatbot. It was integrated with Google Assistant for a richer experience and had ability to voice chat with the user. Two screenshots of the implementation can be seen in Figure 13.
Microsoft Bot Framework chatbot was developed using C# and Microsoft SQL Server 2017. The chatbot does not have NLU built-in in its framework. This caused the chatbot to not be able to make future predictions or correct spelling errors as much as the other frameworks. Two screenshots of the implementation can be seen in Figure 14.
Once the figures of the complaint process were defined they were translated into each frameworks way of processing data. In Botpress and Dialogflow, healthcare business, units and establishments and other necessary variables were stored as entities and phrases were stored as intents. Microsoft Bot framework needed a database to store information, a table was created for each healthcare business and units. User, bot phrases and bot names were stored in its own table.

The final step in the development process was to teach the chatbots as many phrases and much information as possible to make the chatbots understand what the user wanted to know and make the chatbots’ answers as human as possible.

The focus was on how each question could be asked differently to a user and how differently a user could ask or answer a question to the chatbot.

The final versions of the chatbots are able to process 143 frequently asked questions in Swedish and was able to simulate a process of registering a complaint that the patient could want to forward to the Swedish healthcare.

3.5 Evaluation process
In the evaluation process, the participants use the chatbots and are required to answer a questionnaire for each chatbot. The participants are given instructions to test all three frameworks for 15 minutes which is sufficient time for such test. After the test, participants are asked to provide a feedback about the experience with the chatbots. All questions are closed type questions for better quantitative analysis of the feedback and are present as follows:

- “Did you receive the help you needed from the chatbot?”
- “Did you experience that the chatbot was user friendly?”
- “Did you experience that the chatbot was human?”
- “Did you need to wait a long time to get an answer from the chatbot?”
- “Did the chatbot answer any questions too fast?”
- “Did you experience the user interface was good?”
- “Is there any functionality that did not exist, but that you wanted the chatbot to have?”
- “Would you recommend other people to use the chatbot?”

3.6 Evaluation results
The evaluation was done with 10 participants. All participants were managed to fulfil evaluation test of three frameworks within provided 15 minutes with average required time being less than 7 minutes. After the test, participants were asked to answer 8 questions as seen in subsection 3.5. The evaluation results can be found in Table 2.
<table>
<thead>
<tr>
<th>Framework</th>
<th>Botpress</th>
<th>Microsoft Bot Framework</th>
<th>Dialogflow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Did you receive the help you needed from the chatbot?</td>
<td>80% YES, 20% NO</td>
<td>70% YES, 30% NO</td>
<td>90% YES, 10% NO</td>
</tr>
<tr>
<td>Did you experience that the chatbot was user friendly?</td>
<td>80% YES, 20% NO</td>
<td>60% YES, 40% NO</td>
<td>90% YES, 10% NO</td>
</tr>
<tr>
<td>Did you experience that the chatbot was human?</td>
<td>50% YES, 50% NO</td>
<td>40% YES, 60% NO</td>
<td>70% YES, 30% NO</td>
</tr>
<tr>
<td>Did you need to wait a long time to get an answer from the chatbot?</td>
<td>10% YES, 90% NO</td>
<td>100% NO</td>
<td>10% YES, 90% NO</td>
</tr>
<tr>
<td>Did the chatbot answer any questions too fast?</td>
<td>20% YES, 80% NO</td>
<td>30% YES, 70% NO</td>
<td>10% YES, 90% NO</td>
</tr>
<tr>
<td>Did you experience the user interface was good?</td>
<td>90% YES, 10% NO</td>
<td>70% YES, 30% NO</td>
<td>90% YES, 10% NO</td>
</tr>
<tr>
<td>Is there any functionality that did not exist, but that you wanted the chatbot to have?</td>
<td>60% YES, 40% NO</td>
<td>70% YES, 30% NO</td>
<td>80% YES, 20% NO</td>
</tr>
<tr>
<td>Would you recommend other people to use the chatbot?</td>
<td>70% YES, 30% NO</td>
<td>50% YES, 50% NO</td>
<td>90% YES, 10% NO</td>
</tr>
</tbody>
</table>

Table 2: Evaluation results from the questionnaire.

The evaluation results in Table 2 show that Dialogflow was the preferred chatbot. It performed better in all questions. While in question 4 and 6 there was a draw a with Botpress. The results also show that Microsoft Bot framework chatbot was the least preferred chatbot. In Figure 15 a graphical depiction of the evaluation results can be seen.

Figure 15. A graphical depiction of the evaluation results.
4 Analysis and discussion

This section describes an analysis of non-technical aspects and the SWOT analysis of the development process, with the intention of providing a description of what strengths, weaknesses, opportunities and threats different kind of bot framework possess when implementing a chatbot to handle specific tasks or conversations. The chatbot frameworks that were taken into consideration for this analysis were Microsoft Bot, Dialogflow and Botpress. Followed with a discussion of the results from the questionnaire.

4.1 Analysis of non-technical aspects

The implementation of the chatbots within the healthcare provider improved the performance of case registering and answering frequently asked questions about the institute. Most test users thought it would be an effective tool to automate some of their most time consuming administrative tasks, however they were critical to the lack of human aspects the chatbots displayed, such as the understanding of the context of the dialogue or lack of human emotions and moral support.

The advantages of implementing the chatbot prototypes were of economical and social benefits. Such as the time it would take to register a case, according to the test users it would take approximately 30 minutes - two hours, whilst with the chatbots implementations it took 30 seconds - 2.5 minutes. The conclusion which could be drawn here is that from an economical and social standpoint, the benefits of introducing chatbot technology within the Swedish healthcare, could result with saving time from time consuming administrative task and reduced stress levels for the healthcare employees.

From an ethical and moral standpoint, the chatbots showed lack of moral support and human intelligence to understand different scenarios the test users could setup for the chatbots. The test users wanted the chatbot to show compassion for patients, since they could have traumatic experiences with the treatments they received from their healthcare facility. A big knowledge base is needed to setup as many different scenarios in which a chatbot could experience with a user to make a chatbot as human as possible.

The biggest challenge of implementing chatbots within the healthcare is the legal aspects of handling sensitive patient data. Since the introduction of GDPR and PdL have introduced a lot of regulations, which have reduced the possibilities of implementing AI technologies within the general healthcare. More research is needed as to how chatbots should handle sensitive data.

4.2 Analysis of feedback from evaluation

Participants during the evaluation process shared their opinions about the experience they had.

Regarding the Botpress framework they mentioned that it was human at first, but after some time it was more robotic. It repeated itself, answered too fast and it was confusing to know if there is something at the other side that will give a response without a welcome message in the beginning of the conversation. Also it was fast and easy to register a errand and the questions felt like they gave a answer. There were suggestions in different areas so that it was easy to answer questions. But it was a bit too difficult to understand what “with help” or “without help” means. Regarding the user interface, participants said that it had a bit too big
chat for the best experience with the text chat and sometimes it took too long to receive an answer.

Regarding the Microsoft Bot Framework they mentioned it was limited in how questions could be asked, which was not human-like behaviour and also that the chatbot responded too fast sometimes. Regarding the user interface it had a more hospital feel in the looks. It looked the best with colors used in the user interface and had many options to choose from to answer or ask questions. But it was difficult to find the reply from the chatbot due to scrolling in the user interface. The chatbot stopped working and was slow with the response sometimes which made it seem more human but also as if it was a problem with the system.

Regarding the Dialog framework they mentioned it gave a response on things it should not need to. It was very similar to the Microsoft Bot Framework chatbot. However it lacked many options to choose from to answer questions. The participants thought the voice which was used was a bit too robotic, but better than nothing.

The participants thought the chatbot was user friendly and functioned properly. However sometimes it acted too fast to and the participants had to repeat their answers. The participants could use a smartphone and a voice chat to both speak and write to the chatbot, which made it different compared to the other chatbots. Regarding the user interface, the participants said it had more fitting structure with the chat since it didn’t limit itself to the size of the browser or screen or anything else.

4.3 SWOT analysis of Botpress framework

Botpress is a free to use for developers and open source software which allows developers to integrate their source code as middleware and integrate them with third party services. The developer is given a wide range of choices regarding data storage, deployment and platforms. The most fundamental functions of a chatbot exists within the framework, which decreases the development time of a chatbot.

One of Botpress strengths is the flexibility of extending the default functionality with the use of modules. The default modules that comes with Botpress are NLU, Channel-Web, QnA and Flow editor to build a functional chatbot prototype. Developers can also customize their own modules and colors in the graphical user interface to further enrich the functionality of the chatbot.

Botpress supports many platforms and message services and could be deployed either locally or on any server. Complex machine learning algorithms could be incorporated into the framework which would allow the chatbot to adapt to more use cases and scenarios within any organization.

Botpress allows developers to setup a chatbot in javascript. The capabilities of Javascript are therefore usable with Botpress and a knowledge base within javascript programming is a necessity.

For a Botpress chatbot to be perceived as a human and give the best solution to a problem, it needs as much information as possible about how the user responds and what solution is the most satisfying to the problem. The default database in Botpress is SQLite 3, the database has issues with securing the data and compressing it. There is no settings for user management within SQLite. If someone has access to the unit where the botpress chatbot is developed, the attacker has access to all information that is not encrypted and stored within the SQLite file. By default the database is not encrypted which causes concern.
Another potential threat is that, depending on the project size, the information that is stored within the SQLite file could be unnecessarily big, resulting with slowness within the environment or causing the chatbot to crash. However Botpress supports the use of other more secure databases that developers should think of using instead. SQLite does also support encryption and compression, but it needs to be configured.

Another potential threat is that sometime in the future the developers may have to pay fees for licenses to use the Botpress framework.

Most users thought that Botpress was helpful and user-friendly because it gave users the help they needed and was easy to understand and use. It also had fast responses and intuitive user interface which helped users handle questions by the chatbot. Despite all this the Botpress chatbot was not able to handle complex conversations and therefore needed more information and development time.

A summary of SWOT analysis for Botpress Framework can be seen in Figure 16.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helpful</td>
<td>Developer friendly</td>
</tr>
<tr>
<td>Low development costs</td>
<td>Robustness</td>
</tr>
<tr>
<td>Modularity</td>
<td></td>
</tr>
<tr>
<td>Open source</td>
<td></td>
</tr>
<tr>
<td>Supports Javascript</td>
<td></td>
</tr>
<tr>
<td>User-friendly</td>
<td></td>
</tr>
<tr>
<td>Opportunities</td>
<td>Threats</td>
</tr>
<tr>
<td>Customizable color in Graphical User Interface</td>
<td>License costs</td>
</tr>
<tr>
<td>Development capabilities</td>
<td>Data security</td>
</tr>
<tr>
<td>Integration with various databases</td>
<td></td>
</tr>
<tr>
<td>Platform independent</td>
<td></td>
</tr>
</tbody>
</table>

Figure 16. SWOT analysis of Botpress Framework

4.4 SWOT analysis of Microsoft Bot Framework

Microsoft Bot Framework is free for developers to use and allows developers to setup a chatbot in C#, with the capabilities of integrating the chatbot with applications and web services developed in C#.

The framework uses the Botbuilder SDK to build sophisticated conversations, which the user can either communicate with the bot through simple text or rich cards that includes images, buttons and choices.

The framework can also be integrated with Microsoft Language Understanding Intelligent Service (LUIS) to provide NLU so that dialogues could be interpreted in written or spoken language. The variation of these functions offer the developers to build robust chatbot that could handle conversations in large enterprises. The framework also has support for integration with existing or custom database, which offers a higher level of security for storage of data for the intents and entities, depending on the choice of database.

Microsoft bot framework also has support for many channels such as Messenger, Bing, Email, Twilio, Web chat, etc, which allows the chatbot to be integrated within many platforms.
When developing a chatbot to handle a specific task with the use of Microsoft Bot Framework, the cost for development is limited to the knowledge base within C# programming and management costs to maintain the service. It costs money to deploy the service within Azure cloud, which limits the accessibility and exposure for the chatbot.

For a Microsoft Bot framework chatbot to be perceived as a human and give the best solution to a problem, it needs as much information as possible about how the user responds and what solution is the most satisfying to the problem. The Microsoft bot framework does not have a built-in database to store information. The developer needs to use a secure database for the service to be available at all times.

Microsoft Bot Framework does not have built-in NLU functionality. Instead it needs to integrate with an existing or custom made NLU. This could be a good thing for development purposes if existing NLU is not good enough for the chatbot and improvements have to be made.

The documentation for the Microsoft Bot framework is lacking, there are not many commentaries of how to use classes in the framework or the guides for the framework are incomplete. The development is therefore harder to comprehend than other frameworks.

Microsoft Bot framework uses an emulator to chat with the chatbot. The emulator is limited in graphical capability that it is not a fully enjoyable chat experience for users. This can be seen in the evaluation results.

Another potential threat is that sometime in the future the developers may have to pay fees for licenses to use the Microsoft Bot framework.

Most users considered the chatbot to be helpful and user-friendly. Despite all this the Microsoft Bot framework chatbot was not able to handle complex conversations and therefore needed more information and development time.

A summary of SWOT analysis for Microsoft Bot Framework can be seen in Figure 17.
4.5 SWOT analysis of Dialogflow framework

Dialogflow Framework is free for developers to use and engages conversation with users in either a text-based or voice format. The framework has support for a high level of NLU functionality, such as speech recognition, which includes Text-To-Speech and Speech-to-Text.

The framework has also support for many languages, including Chinese, English, French, Swedish, etc. Most users thought the Dialogflow chatbot gave the most human-like experience, since they could speak directly with the chatbot.

Dialogflow allows developers to setup a chatbot with fulfillment integration where a backend is contacted to create more dynamic responses.

The Dialogflow chatbot could be integrated with many messenger services, such as Slack, Google Assistant, Facebook Messenger, Skype, etc. With the support of many languages and integration availabilities makes the bot framework highly scalable with deployment of the chatbot to many different platforms. The users were able to converse with the chatbot through their smartphones with the in-built integration of Google Assistant, which they thought increased the accessibility to the chatbot.

One of the biggest weaknesses with the bot framework was the costs for maintenance and the amount of requests a user can make to the chatbot. Developers might also experience difficulties with testing the chatbot for the standard cost free edition. With this restriction, the development of a chatbot with Dialogflow would increase the development process time and a limited chatbot.

Dialogflow stores entities and intents in the Google Cloud service. This causes an issue with data security as vital information could be sold to third parties that customers of a company do not want to be leaked.

Dialogflow framework is free for now for developers to use. However, in the future, developers may have to pay fees for licenses to use Dialogflow.
Dialogflow has the authorization to shutdown the service for e.g. maintenance. If Dialogflow goes down then the chatbot will not be available until the service is back up again.

Despite all this the Dialogflow chatbot was not able to handle complex conversations and therefore needed more information and development time.

A summary of SWOT analysis for Dialogflow framework can be seen in Figure 18.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free for developers</td>
<td>Robustness</td>
</tr>
<tr>
<td>Fulfillment</td>
<td>Standard free edition</td>
</tr>
<tr>
<td>Helpful</td>
<td></td>
</tr>
<tr>
<td>Human-like appearance</td>
<td></td>
</tr>
<tr>
<td>NLU Functionality</td>
<td></td>
</tr>
<tr>
<td>Scalable</td>
<td></td>
</tr>
<tr>
<td>Support for many languages</td>
<td></td>
</tr>
<tr>
<td>User-friendly</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Opportunities</th>
<th>Threats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessibility</td>
<td>Availability</td>
</tr>
<tr>
<td>Integration with various databases</td>
<td>License costs</td>
</tr>
<tr>
<td>Integrated with many messenger services</td>
<td>Data Security</td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
</tr>
</tbody>
</table>

Figure 18. SWOT analysis of Dialogflow framework.

### 4.6 SWOT analysis of all frameworks

All chatbots were considered by the users to be helpful when registering a case within a healthcare provider or getting an answer to questions. The chatbots were also considered to be user friendly, since it did not take long time for the users to get accustomed with the chatbots.

All the users considered that all chatbot had potential to be implemented within a healthcare provider, with many advantages in reducing the workload for the staff members within healthcare.

The chatbots have support for integration with various databases and could be integrated with many services. Such as web- and mobile applications, which increases the availability and accessibility of the chatbots. Since many healthcare providers are using different systems and databases to store and retrieve data, the chatbots could easily be deployed within the systems for a specific use case.

For the implementation of all chatbots, the most difficult challenge was the data mining process. Every chatbot needed lots of information to appear as human as possible for the users. All chatbots had functionality to handle specific tasks. If the user gave wrong input, the chatbots would interpret it as wrong context, and appear to the users to be less human, which resulted in a lack of robustness for all chatbots.

All frameworks are free for now for developers to use. However, in the future, developers may have to pay fees for licenses to use the frameworks.

Dialogflow has problems with availability when e.g. the site needs maintenance, the chatbot becomes unavailable for some time.
The default database in Botpress is SQLite 3, the database has issues with securing the data and compressing it. There is no settings for user management within SQLite. If someone has access to the unit where the botpress chatbot is developed, the attacker has access to all information that is not encrypted and stored within the SQLite file. By default the database is not encrypted which causes concern.

Another potential threat is that, depending on the project size, the information that is stored within the SQLite file could be unnecessarily big, resulting with slowness within the environment or causing the chatbot to crash. However Botpress supports the use of other more secure databases that developers should think of using instead. SQLite does also support encryption and compression, but it needs to be configured.

The Microsoft bot framework does not have a built-in database to store information. The developer needs to use a secure database for the service to be available at all times.

Dialogflow stores entities and intents in the Google Cloud service. This causes an issue with data security as vital information could be sold to third parties that customers of a company do not want to be leaked.

A summary of SWOT analysis for all frameworks can be seen in Figure 19.

<table>
<thead>
<tr>
<th>Strengths</th>
<th>Weaknesses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free for developers</td>
<td>Robustness</td>
</tr>
<tr>
<td>Helpful</td>
<td></td>
</tr>
<tr>
<td>User-friendly</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Opportunities</th>
<th>Threats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration with various databases</td>
<td>Availability</td>
</tr>
<tr>
<td>Integration with many services</td>
<td>License costs</td>
</tr>
<tr>
<td></td>
<td>Data Security</td>
</tr>
</tbody>
</table>

Figure 19. SWOT analysis of all frameworks.

4.7 Discussion

4.7.1 Prototypes effectiveness

The prototypes did not appear to be fully human, but they did appear to be helpful and responded fast. The prototypes were able to help the user registering a case and answer some frequently asked questions. But for the prototypes to be fully effective, each prototype would need improvements.

Botpress chatbot needs a voice integration and more information how to structure the data sets within the framework, with the intention of gaining a better understanding of human input and enrich the chatbot with more options to handle different use cases and provide better moral support for the users.

Microsoft bot framework chatbot needs a human voice, integration with LUIS, more information to interpret user messages and integration with other channels. It also needs a
custom GUI for the users’ needs to avoid the scrolling issue in the emulator GUI and provide more answer to questions the user could ask.

The Dialogflow chatbot needs a less robotic voice, more information to interpret user messages and integration with other channels. It also needs to have more options to help the user with registering a case and provide answer to questions the user could ask.

Overall more improvements need to be made to make the chatbots more effective e.g. deeper understanding of complex tasks, custom GUI for a friendlier experience towards users’ preferences, human voice chat for all chatbots and integration with other channels to reach a bigger market.

4.7.2 Prototype optimizations
The prototypes could use more data to answer correctly. The chatbot that use Botpress and Microsoft Bot Framework could use implementation of voice chat. The Microsoft Bot Framework chatbot could use questions outside of the suggestions text scope and a new user interface without so much scrolling for the answer asked to the chatbot for a more human like experience. All chatbots could use integration with most other services to reach out to a bigger audience and a implementation of simulated video chat. Also errors should be handled more effectively in all chatbots and databases should be changed to more secure databases that are not light, e.g. SQLite 3 should be changed to something more secure for Botpress and Dialogflow should use some other secure database with fulfillment for more control over data processed by the chatbot.

4.7.3 API:s relationship according to the theoretical model
The API:s relate pretty well to each other according to the theoretical model. They all use frontend, middleware and backend to change and improve. This will help development in the long run as there is possibilities for improvements on the prototypes wherever it is wanted. Possible improvements can be seen in section 4.7.2.

4.7.4 Chatbots performance in predefined use-case scenarios
Most users expressed the chatbots did not have a more human conversation as in they did not understand everything a user told it, e.g. the user tells the bot a story and the bot responds with a message of “I do not understand.” notifying the user it does not understand the story. For the bot to handle stories, the bot needs more information to process it or more complex ML algorithms to adapt to human conversations.

4.7.5 User communication with the chatbots
The optimal chatbot would use text, clickable options, voice and video. Video however is harder to develop in simulation as it would require advanced 3D-graphical capabilities to look real. Video simulation of a chat is something that should be developed in the future for a more human experience.

In this particular case each chatbot had its own way of communicating with the users to provide more data on an optimal chatbot. The Dialogflow chatbot used voice, clickable options and text to communicate. However sometimes the voice spoke to fast making it less human.

Botpress chatbot used clickable choices and text. Most users thought it was confusing that you could choose between help and no help options for chatting.
The Microsoft framework chatbot used clickable options to communicate with the user and was very limited with text chat. Text was only writable when it was obvious that text was supposed to be written, as in no options were visible or the text in clickable options were writable. Something that most users did not like.

4.7.6 Chatbots and use of audio
During testing only Dialogflow chatbot was able to process it. Microsoft framework chatbot has support for voice communication as well. However not enough time was spent on implementing it. Botpress developers are currently working on developing an API that would integrate the bot framework with Google Assistant that supports voice communication.

4.7.7 Response speed of the chatbots
Botpress chatbot had a built-in function for delaying responses to user input, which gave the chatbot a more human impression. Dialogflow chatbot had support for text-to-Speech function which delayed the answers, that made the users find Dialogflow chatbot most human. While Microsoft framework chatbot gave the least impression of being human as it responded too quickly to user input.

4.7.8 Benefits and consequences of an implementation of chatbots in administrative healthcare tasks
The consequences of an implementation of chatbots within the Swedish healthcare could result with more work being introduced implementing the chatbots and automating easy work tasks, resulting with reduced time staff members spend on doing administrative tasks and spend more time on healthcare related tasks.

Most users thought the chatbot was useful despite it not being fully human. This would mean that more people could receive better healthcare in the long run assuming the optimal chatbot is able to act as a human would.

4.7.9 Chatbots ability to complement a staff member of the healthcare
Most of the users saw the true potential of what a chatbot can do and recommended it to other users and wishes that it could be implemented within the establishment. A chatbot has the ability to handle administrative tasks like a receptionist but only faster. Instead of one patient speaking to one person, multiple patients can speak to a chatbot on multiple devices. This could decrease waiting times and give the user help that is needed at the time without waiting for its turn to speak. A chatbot is able to act human as it has e.g. voice, text, a graphical user interface, spell correction and information storage to be able to communicate with the user from a distance. The administrative tasks that these chatbots implemented were able to handle were to register a case for patients within different types of healthcare area, such as primary care, psychiatric care, emergency hospitals, etc. They were also able to handle frequently asked questions that people had about the organizations and their work.
5 Conclusions

Three chatbots were designed and compared for handling administrative healthcare tasks. The results of the quantitative analysis is that the users had a high opinion of introducing a chatbot as a resource to handle routine and daily tasks for staff members within a healthcare facility, in order to reduce the induced stress and maximize the performance within the general healthcare.

The chatbots implementations were able to perform the task they were designed to do within the predefined use cases, with some fault error generation. The user were able to communicate with the chatbots easily and ask some questions regarding the responsibilities and tasks handling within the healthcare institute.

Botpress had a built-in function to delay its responses, which made it more human appealing for the user, while Microsoft Bot Framework responded too quickly to be perceived as a human. Dialogflow framework had a delayed responses, as it read out the response text with voice.

Out of all the chatbot implementations, only Dialogflow had support for voice communication, which resulted with test users thinking it was the most human chatbot. Which increased it’s value to be the preferred chatbot framework for this prototype. This is also supported by the strengths and opportunities from the SWOT-analysis. In which Dialogflow provided more options for the developer than the other frameworks.

The benefits of a chatbot implementation in healthcare is that it has the potential to automate many administrative tasks, such as patient registration, in- and out-check of patients, case registration, make diagnoses, answering frequently asked questions, etc. Based on SWOT-analysis and results from the quantitative analysis, chatbots have a potential to reduce the workload and the stress that staff members within healthcare experiences on a daily basis, to increase the work performance and to give patients a better healthcare.

Although more research is needed for this technology to expand, the conclusion of this work is that a chatbot could be a relevant tool to automate administrative tasks within the Swedish healthcare.
References


Appendix

The figures below shows a process of registering a case and were painted at meetings with Patientnämndens förvaltning in Stockholm.
The figure below describes the process of registering a case anonymously.

Chatbot

Do you want to be anonymous?

No

Are you a man or a woman?

Man

What is your name?

Name

User

Skipped if user answered yes

Skipping stops here
The figure below shows a process of registering contact information in the complaint process.
The figure below shows a process of registering complaint in the complaint process.

The list of healthcare businesses and their components can be found here: https://docs.google.com/spreadsheets/d/1GcmOr3lbTMduYKMmK5AxSdDri6WIdAAHb8jJb8p4pLU/edit?usp=sharing

The chatbot FAQ can be found here: https://docs.google.com/document/d/1MBOfF9ofsA4cDKXzuaislQFtEyZ1mwSuqt5VT1e3mQ/edit?usp=sharing