Faculty of Science and Technology
Department of Computer Science

AI Chatbots in Health: Implementing an LLM-Based Solution to Promote Physical Activity

Sondre Elvebakken Løvås

INF-3981 Master's Thesis in Computer Science - June 2024

Supervisors

Main supervisor:	André Henriksen	UiT The Arctic University of Norway, Faculty of Science and Technology, Department of Computer Science
Co-supervisor:	Eirik Årsand	UiT The Arctic University of Norway, Faculty of Science and Technology, Department of Computer Science
Co-supervisor:	Dillys Larbi	UiT The Arctic University of Norway, Faculty of Health Sciences, Department of Clinical Medicine
Co-supervisor:	Elia Gabarron	Østfold University College, ICT and Learning, Department of Education
Co-supervisor:	Denecke Kerstin	Bern University of Applied Sciences, Institute Patient-centered Digital Health, Department of Engineering and Computer Science

This thesis document was typeset using the *UiT Thesis LaTeX Template*.
© 2024 – `http://github.com/egraff/uit-thesis`

Abstract

With the emergence of powerful generative AI models comes the possibility of creating knowledgeable and engaging chatbots, which have the potential to significantly enhance several areas of the user's life. This thesis focuses on the design and implementation of FysBot, an application with an integrated chatbot that aims to increase the user's physical activity levels. In collaboration with a PhD project, FysBot's design is guided by scientific evidence on user preferences in physical activity chatbots, with a strong emphasis on the security and privacy of the users' data.

To facilitate personalized conversations on different topics, the use of *chatbot personas* is proposed. A chatbot persona is specialized for a single conversation topic and is responsible for generating an appropriate response to the user. FysBot implements personas for the following conversation topics: exercise recommendations, hiking trip suggestions and step progress review. In addition, a default persona is used to handle queries unrelated to the listed conversation topics. In order to mediate a query to the correct persona, queries are first presented to a routing persona, which then routes the query to the appropriate destination.

FysBot's design has undergone testing via a usability study conducted using the think-aloud method. From this study, the overall impression of FysBot has been positive. However, several areas require improvement to enhance the user experience and intuitiveness. Addressing these areas will, in turn, increase engagement and promote healthier lifestyles among users.

Acknowledgements

I would like to thank my main supervisor, André Henriksen, and co-supervisors Eirik Årsand, Elia Gabarron and Kerstin Denecke for their support and guidance not only in the supervision meetings but also for taking the time to answer my spontaneous questions and checking in on me to see how I was doing. They have shown genuine interest in the success of the thesis and have always motivated me to do better.

Many thanks to my co-supervisor and the PhD student referred to in this thesis, Dillys Larbi, whom I have worked closely with and who has helped me with the usability tests, application design, and much more. I am extremely grateful for all you have done and wish you all the best for the future.

Thank you to Assoc. Prof. Elisavet Kozri, thank you for taking the time to answer my questions about security design.

To the participants of the usability test, thank you for your participation and honest feedback.

To my friends at the university, thank you for all our discussions. Your dedication to the field of computer science is an inspiration.

To my family, thank you for your limitless support and to my brother, who has always been my role model.

And lastly, Benedikte, thank you so much for moving with me, believing in me, and pretending to be interested in my incoherent ramblings over the last five years.

Contents

List of Figures

List of Tables

List of Listings

List of Abbreviations

AAB Android App Bundle

BCTs Behaviour Change Techniques

CLI Command-Line Interface

ERP Exercise Recommender Persona

GDPR General Data Protection Regulation

JPA Java Persistence API

JWT JSON Web Token

LAP Location Activity Persona

LLM Large Language Model

NLP Natural Language Processing

OIDC OpenID Connect

ORM Object-Relational Mapping

OS Operating System

PA Physical Activity

PRP Progress Review Persona

RAG Retreival Augmented Generation

RN React Native

RP	Routing Persona
SDK	Software Development Kit
TLS	Transport Layer Security
UI	User Interface
VM	Virtual Machine
WHO	World Health Organization

1 Introduction

1.1 Background

Individuals living with cancer, type 2 diabetes, heart diseases and obesity face an elevated risk of mortality[1]. It has been found, however, that physical inactivity is a modifiable risk factor for these conditions[1, 2] and can enhance quality of life[2]. A sedentary lifestyle has become a global problem, and in Norway, it has been observed that 19% of the adult population is obese and 60% is overweight[3, 4]. It has also found that 27% of men and 24% of women in Norway are insufficiently active[5].

With the widespread use of smartphones and other smart devices, designing and developing software applications specifically to help individuals become more physically active is now possible. Most smartphone manufacturers produce phones that come pre-equipped with health applications that use the phone's internal sensors to monitor the user's steps, heart rate and metabolism. Third-party apps like Strava[1] also use the geopositioning of the smartphone or smartwatch to track the user's activity and have a feature to share the activity in a social network that exists within the app.

The field of Natural Language Processing (NLP) has seen a lot of improvements in recent years, and research on chatbot applications used for promoting physical activity has become a hot topic[6, 7]. The prominent features of chatbots

1. https://www.strava.com/

are that they can simulate human-to-human interactions and contain a vast knowledge base. A chatbot is also a very effective tool for packing many different functionalities into a single user interface. This is an innovative solution for keeping the user engaged and can have positive effects on individuals who otherwise struggle to reach out to other people for social interactions. A chatbot is always available when the user needs it and can be designed as a companion with a personality that is tuned to the user's liking. With this, a chatbot could be the source of motivation that the individual needs to become more physically active.

1.2 Previous Work

In a study conducted by Wlasak, Zwanenburg and Paton[7], a chatbot was created and used to motivate individuals to do physical activity during the COVID-19 pandemic[7]. They based the chatbot on three Behaviour Change Techniques (BCTS): goal setting, experimenting, and action planning, and evaluated the system based on changes in the user's step count. Telegram was used as the channel for interacting with the chatbot, and Google Fit was used to collect steps. The chatbot interactions in this work consisted of weekly status updates where the user was able to set their goals for the week. The authors found that participants had an increase of 143 weekly move minutes from the 272 minutes baseline, and 46.1% of the participants credited the chatbot for their increased physical activity[7].

Maher et al.[8] used a chatbot that allowed users to set goals and monitor progress. An accompanying website was used to show videos that educated the users on physical activity. The chatbot was powered by the NLP software Watson from IBM and was deployed on the messaging platform Slack. The study found that participants, on average, had a 109.8-minute increase in time spent being physically active compared to their baseline[8].

In their 2023 study, Vandelanotte et al[6]. explored the use of a chatbot system to educate users about physical activity, nudge them at opportune moments using machine learning, and answer PA questions using generative AI. A set of predefined conversations was used to help users become more motivated and could be initiated by the nudge engine or triggered when the user asked a question that closely related to one of the conversation. The nudge engine also used weather data and the users' locations to suggest suitable activities. The NLP behind the chatbot was implemented using Google DialogFlow, and the chatbot was released as a standalone app[6].

1.3 Objective

It is possible to integrate chatbots into popular chat applications like Messenger, Telegram, or Slack, which already have large user bases, thus making it easier for people to use them. Using these chat applications also removes the need to implement a standalone app, which usually takes considerable time. The downside is that the privacy of users' data cannot be fully guaranteed, and any unforeseen regulation changes could render the chatbot unusable. Furthermore, the chatbot's functionalities are limited to what already exists within the chat application. Therefore, the aim is to implement a chatbot in a standalone application that preserves the privacy of the user's data and provides a rich set of features to help the user achieve their physical activity goals.

1.4 Research Question

In the capstone for this project[9], the scoping literature review found a lack of detailed explanations on how to implement a physical activity chatbot in a standalone application. In addition, although there are existing studies about the features users prefer in a PA chatbot app, there is a lack of evidence on how well these elements work in practice. It was also found that most of the PA chatbots used higher level NLP software, were mostly rule-based, and provided static conversations. This has led to the following research questions:

- **RQ1**: How can a chatbot be developed using ChatGPT to help users become more physically active?

- **RQ2**: How can a physical activity chatbot be implemented in a way that preserves the security and privacy of the users' data?

- **RQ3**: How can integrated interactive features enhance the effectiveness of a physical activity chatbot?

1.5 Outline

The thesis contains the following chapters:

Chapter 2 - Background provides necessary technical details for understanding the analyses and discussions in the subsequent chapters.

Chapter 3 - Method details the methodologies used for designing and testing

FysBot.

Chapter 4 - Design presents the design of the FysBot application and the architecture supporting it.

Chapter 5 - Security explains the design of the security features in FysBot.

Chapter 6 - Implementation goes into detail about realizing the FysBot application, architecture, and security features, as well as how the application is deployed.

Chapter 7 - Evaluation presents the findings from the usability study and the results from running FysBot.

Chapter 8 - Discussion discusses the usability study findings, obstacles encountered with the implementation, as well as the strengths and limitations of FysBot.

Chapter 9 - Future Work goes into detail about potential features in FysBot and priorities for future work.

Chapter 10 - Conclusion presents the final thoughts about the thesis.

2 Technical Background

This chapter gives an explanation of technical terms and concepts that are used throughout the thesis. It starts with introducing what physical activity is in section 2.1. Section 2.2 then introduces chatbots, what they are used for, and how they can be used in relation to physical activity. Section 2.3 then explains what a Large Language Model (LLM) is, and is followed by section 2.4 which introduces a specific implementation of a LLM, namely ChatGPT. Section 2.5 goes into details about some of the security concepts that are used in the implementation of FysBot. Finally, section 2.6 introduces concepts that are used to implement the FysBot application.

2.1 Physical Activity

Physical Activity (PA) is defined as "any bodily movement produced by skeletal muscles that result in energy expenditure" [10]. Our bodies store energy from the foods we consume and consuming more energy than what we expend through physical activity and metabolic processes results in an increase in body weight[11]. The opposite is true; consuming less energy than the body expends will result in weight loss. The increase or decrease in weight can be caused by changes in muscle mass and the body storing energy as fat to be used when the body is at an energy deficit. This balance between muscle mass and fat storage is influenced by both diet and exercise[12].

The Norwegian Institute of Public Health found that as of 2022, about one in four Norwegians are not sufficiently active based on the WHO recommendations of at least 150 minutes of moderate-intensity activity per week[5][2]. It is important to encourage individuals living with obesity, cancer, or diabetes to become more physically active as the health benefits of PA can have a significant impact on their quality of life[2]. For these individuals, insufficient PA can even be life-threatening, so attempts must be made to make exercising more accessible and engaging.

Not all exercising has to be purposeful. It is important to be aware that daily activities, which can also be called non-exercises, can be significant towards increasing daily expenditure of energy[13]. By being aware of this, people can actively make choices that are more healthy for them and which feel more accomplishable compared to, for example, going to the gym. Some individuals might not have many opportunities to execute non-exercises and could, for various reasons, be uncomfortable with going outside for a walk or going to the gym. These individuals should be made aware that there exists a vast amount of different exercises that can be done at home and could, for example, be done with just a simple chair. It is important not to confuse exercising with brutal gym sessions but to understand that even these simple chair exercises can significantly improve health.

2.2 Chatbots

2.2.1 Overview

Chatbots are computer systems that simulate human-to-human conversations, where the human, using free text or some other input, gets a response from the system based on the input[14]. They have been extensively used to increase productivity, for example, on e-commerce or banking websites, to provide the user with the requested information quickly.

Earlier generations of chatbots often came with a set of predetermined conversation paths, where the user can choose from a set of conversations. These would then evolve into more sophisticated chatbots that use Natural Language Processing (NLP) to parse the user's intentions from free text. Recently, chatbots have taken NLP to the next level by using Large Language Model (LLM) technology and is used by the most popular chatbot application yet, ChatGPT. ChatGPT and similar chatbots are known for their surprising accuracy and ability to hold conversations in large contexts. These chatbots are also capable of conveying emotions and can be perceived as trustworthy.

2.2.2 Physical Activity Chatbots

Physical activity chatbots are chatbots that are applied to motivate the user to be more physically active. Not everyone can afford to hire professional help; some might feel uncomfortable doing so even if they could. It is assumed that most people would also like to get assistance quickly, at any time of the day, so there is also the problem of availability. PA chatbots attempt to fill this gap by always being available, allowing the users to freely interact, receive advice, and be motivated by them.

If the PA chatbot works as intended, there is a potential for users to increase the time spent being physically active. There are a number of features that a PA chatbot can have in order to motivate the user. Human likeness is assumed to be one of them, as it will build trust with the user and make the chatbot more engaging. There are also interventions such as allowing the user to set goals, sending the user notifications, and giving the user feedback on progress. Additionally, the chatbot can be programmed to provide exercise-focused recommendations.

2.3 Large Language Models

The Large Language Model (LLM) technology is one of the most significant advancements within NLP, and much of it can be attributed to the development of the transformer architecture[15]. This architecture has the ability to process sequential data such as text with great efficiency and scalability, which in turn allows for the deployment of models with vast parameter sizes. The most important element of the transformer is the self-attention module, which plays a crucial role in "relating different positions of a single sequence in order to compute a representation of the sequence"[15]. In other words, the transformer is able to evaluate how a single word in an input relates to every other word in the same input.

An LLM is trained on large volumes of text, giving it extensive knowledge across many domains. The depth of this knowledge can vary depending on the amount of training data available, which again is influenced by the specific period the model underwent training. When using an LLM in production, domain-specific knowledge is most often a requirement in order to reduce *hallucinations* and give more accurate outputs. When an LLM hallucinates, it produces an output that is factually incorrect and is "due to the model's ability to generate plausible-sounding text based on patterns it has learned from its training data"[16]. There are various ways for an LLM to gain knowledge about a domain, and depending on available resources, some solutions are more feasible than others.

For an LLM requiring high specialization, a process called transfer-learning, also called fine-tuning, could be suitable. In this approach, a pre-trained model serves as the starting point, and its parameters undergo further optimization using domain-specific data[17]. Another simpler approach to gaining domain-specific knowledge is injecting the required information directly into the conversation context, a process called Retreival Augmented Generation (RAG)[18]. The context window refers to the input message provided to the LLM, upon which the model bases its output. Figure 2.1 shows how the domain-specific knowledge, in this case, called documents, are placed in the context. The figure also shows how the model is given a behavior or a personality, as well as instructions. In the RAG approach, parameter values remain static, making it highly feasible to develop in practice while also giving accurate outputs. In most cases, not every piece of domain knowledge can fit into the context at once. Information can be fetched from a data store dynamically based on the user question before it is injected into the context.

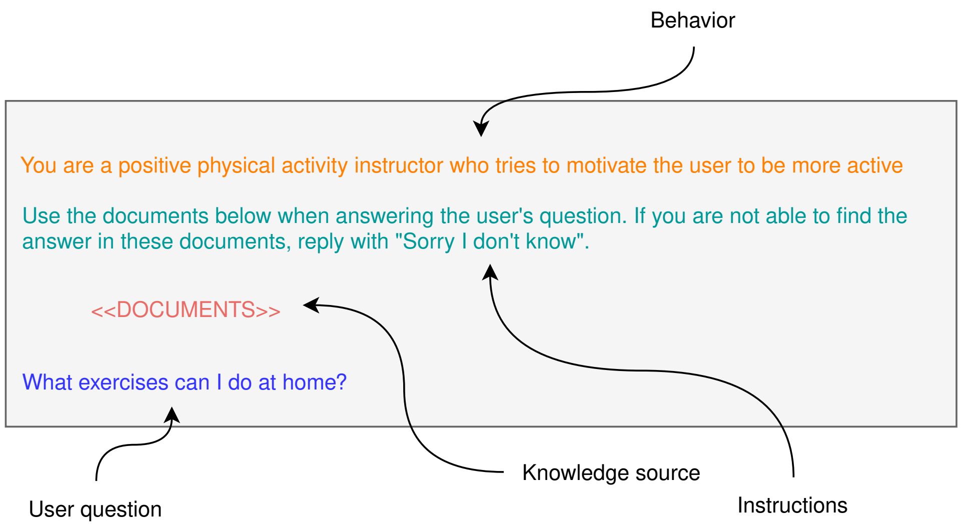

Behavior
You are a positive physical activity instructor who tries to motivate the user to be more active
Use the documents below when answering the user's question. If you are not able to find the answer in these documents, reply with "Sorry I don't know".
<<DOCUMENTS>>
What exercises can I do at home?
User question
Knowledge source
Instructions

Figure 2.1: Example of context given as input to LLM

In order to train a well-performing LLM, large amounts of computational power and data are required. It might not be feasible to train such a model independently, but there are other solutions, such as using open-sourced model parameters. Still, the computation needed to just run inference on such a model that serves multiple users might not be available to some. In this case, companies such as OpenAI provide an LLM as-a-service by providing paying customers with an API that allows the user to run inference on models that are run by OpenAI.

2.4 ChatGPT

2.4.1 Overview

ChatGPT, developed by OpenAI, is a set of LLM models accessible through the OpenAI API. To generate text based on a conversation history, the OpenAI API considers the conversation history as a parameter, which consists of a set of annotated messages. The annotations include which role is associated with a particular message as well as the content of the message. The role is one of "user", "assistant", or "system", where a message annotated with user is a query (for a user message, the content of the message could look like that of figure 2.1) and a message annotated with assistant is the model's response. The system annotation is used for a message that "helps set the behavior of the assistant"[19]. It was found, however, in the capstone[9] that placing the behavior in the context of the user message works better.

2.4.2 Function Calling

The system message has another use, which is in a feature OpenAI calls "function calling". This feature lets the model decide whether or not a function could be called based on the user message. One could then say in the API call to OpenAI that there exists a function that, given the name of a location, fetches the weather conditions for that location. The model will then evaluate the user message and determine if the user wants to know the weather and if the message includes a location. If so, the model will automatically parse out the location from the message and reply with a JSON object that contains the location. The JSON object could then be passed to weather function to fetch the weather conditions. The steps for completing this query are all described in the system message, which OpenAI handles automatically. Function calling also allows for setting *enums* for each parameter that defines which values the parameter can take. In the weather example, one could define legal locations as just Tromsø or Oslo. This is useful for handling typos in the user message. The model will be able to detect the typo and output the correctly spelled location, which can later be used in API requests or database queries. In addition, if the enums are written in English, the user could still write the equivalent word in a different language, and the model will automatically output the English equivalent if it exists among the enum values.

2.5 Security

2.5.1 Data Protection

Any entity processing the personal data of an EU individual is subject to the regulations outlined in the General Data Protection Regulation (GDPR)[20]. The GDPR aims to protect users' personal data and to penalize entities that collect and process data in non-compliant ways. In article 5 of the GDPR official legal text, seven principles for processing personal data are defined as follows[21][20]:

1. **Lawfulness, fairness and transparency**. Personal data must processed lawfully, fairly and in a transparent manner in relation to the data subject.

2. **Purpose limitation**. There must be a specific legitimate reason for collecting the personal data and the data subject must be explicitly informed of the data collection.

3. **Data minimisation**. Personal data must be limited to only what is necessary.

4. **Accuracy**. Personal data must be up-to-date and any inaccurate data must be updated without delay.

5. **Storage limitation**. If the personally identifying data is no longer needed for the specified purpose, it shall be deleted.

6. **Integrity and confidentiality**. Personal data must be processed with an emphasis on confidentiality and integrity.

7. **Accountability**. The data controller is responsible for, and shall demonstrate compliance with, 1-6.

2.5.2 Data Security

Data security is often defined in terms of confidentiality, integrity and availability[22]. Confidentiality pertains to safeguarding information, only allowing access to authorized users while making sure that unauthorized users are unable to read the information or even know about its existence. There are several tools that can be used towards this goal, some of them are[22]:

- **Encryption**: information can, with the use of encryption, be obfuscated in a way such that its content is unintelligible, and with the use of strong encryption, regaining the original content requires significant computational resources. There are two primary ways of doing encryption:

  - **Asymmetric encryption**: a piece of information is encrypted using an encryption key and can (without the use of significant computational resources) only be deciphered using the associated decryption key. The encryption key is often called the private key, which is not to be shared with anyone, while the decryption key is called the public key, which can be freely shared. In asymmetric encryption, it is possible to create digital signatures where the private key is used to generate a signature of a message. If the resulting output of applying the public key on this signature is valid, one can be confident about the sender's identity and that the message has not been tampered with.

  - **Symmetric encryption**: in contrast to asymmetric encryption, the same key is used for both encryption and decryption. Symmetric encryption algorithms are often faster to use compared to asymmetric algorithms but require a method of distributing the key to the user[22]. A solution to this is to use asymmetric encryption to communicate the shared key.

- **Access control**: determines who has access to what. In practice, access control is implemented using roles where a piece of information should only be accessible to entities with the appropriate role.

- **Authentication**: is used to grant access to information and determine the identity of the entity. Several options exist to authenticate an entity, some of which can also be combined to create multi-factor authentication for even stronger security guarantees. Authentication can be done using, for example, username and password, secret keys or biometric information.

- **Authorization**: determines if the entity should be allowed to access the information. This is done by looking at the defined access control policies.

Transport Layer Security

Transport Layer Security (TLS) is the most popular cryptographic protocol for securing communication on the web, ensuring that web traffic between two entities can not be read by unauthorized parties[23]. TLS uses strong cryptographic algorithms for authenticating the communication channel between the server and user, making sure that data can only be read at the endpoints, and makes it infeasible for attackers to alter the data without it being detected[24].

2.5.3 JSON Web Token

The JSON Web Token (JWT) is an internet standard (RFC7519) used to encode a set of claims (information about a subject formatted as key/value pairs)[25]. The integrity and authenticity of the JWT can be secured by digitally signing the JWT with a private key. In addition, the JWT can also be encrypted to achieve confidentiality.

2.6 Android App Development

2.6.1 Overview

Android app development broadly consists of creating a User Interface (UI) and implementing the underlying logic that enables interactions with it. The Google Play Store is the most common alternative to publish the application, allowing others to download and use it.

Installed apps can be in one of the following states:

- **Foreground**. When an app is in the foreground, it is currently open and visible to the user and can be interacted with. In this state, the app is prioritized by the Operating System (OS), allocating more system resources to it.

- **Background**. The app is running in the background but not visible to the user. This can occur when the user enters another app, goes back to the phone's home screen, or the phone is locked. While in this state, the user cannot interact with the app but background processes can still run. The app is, however, allocated limited resources.

- **Terminated**. A terminated app cannot be interacted with and does not perform any background tasks. While in this state, the app has no system resources allocated to it.

There are solutions that enable the periodic waking of terminated apps to run background tasks. Still, these solutions can be unreliable and cannot be run too often as they will be throttled by the OS.

Within an app, there are usually multiple *tabs*, which are screens with different content that the user can navigate. The content and corresponding logic within the tab can be written in the native language of the Software Development Kit (SDK) (Java or Kotlin for Android) and/or using other UI frameworks like React Native (RN). Often, a combination of the native language and other frameworks are used, as using a separate UI framework can speed up development and enable cross-platform functionalities. The native language is used to access platform-specific functionalities that are not available in the framework (e.g., device sensors).

2.6.2 React Native

React Native (RN) is a cross-platform UI framework. In RN, JSX is used to describe the UI and looks very much like regular HTML. React Native contains elements like <*Text*> for displaying text, <*View*> for creating a container, or <*Image*> for displaying an image and can be written in the same file as the JavaScript/TypeScript code.

Components

A core part of RN is the use of components which are units that contain the logic and the markup for the UI, both of which are written in the same file[26]. Components drive the reusability of UI elements and are implemented as JavaScript/TypeScript functions that take in a set of arguments called *props* and return what should be rendered on the screen[26]. A simple example is the implementation of a custom button that has a specific color, shape and size and takes in the text that should be rendered inside the button as a prop. The custom button component and how it can be used is shown in listing 2.1

Listing 2.1: Custom button component

```
// CustomButton.tsx
function CustomButton(props) {
    return (
        <Button
            style={{
                height: 10,
                width: 10,
                color: "black"
            }}
            title={props.title}
        />
    )
}

// App.tsx
function App {
    return (
        <CustomButton title="Press me"/>
    )
}
```

State

Variables that are defined in React components can be used as part of the UI and even updated by, for example, defining a button that increments a counter each time it is pressed. If the counter variable value is shown in the UI, one would, however, not see the updated value as the button is pressed; only its initial value will be shown. The counter value would also be reset whenever the component re-renders. React offers *state hooks* whenever data values has to persist across re-renders and triggers a re-render whenever the state value is updated to show the new value in the UI.

Sharing the same state among several components is often required in React, and passing them as props can lead to "prop drilling", where a state is passed down multiple layers of components, which makes it harder to maintain the code base. For example, an application needs to render a welcome message to a user if the user is logged in and display a login screen if not. For this, it creates a state value *isLoggedIn* and a setter for updating the state *setIsLoggedIn*. If the user is not logged in, the setter is passed to the login component. The login component itself can consist of a 'create new user' component and a 'log in existing user' component for which the setter has to be passed a second time.

This is illustrated in figure 2.2

An improved alternative to this is to use a *context* that holds the logged-in state, which can then be *consumed* by the components that need to read or update the state. This is illustrated in figure 2.3. One can imagine that the *LoggedInContext* could also contain a state value for the user's username, which could be consumed in other components. Another great reason for implementing contexts is that they can contain all the logic surrounding the state values that they provide. Instead of providing the *setIsLoggedIn* setter directly, it could provide a function that not only sets the state value but also makes the necessary API calls required to log in the user.

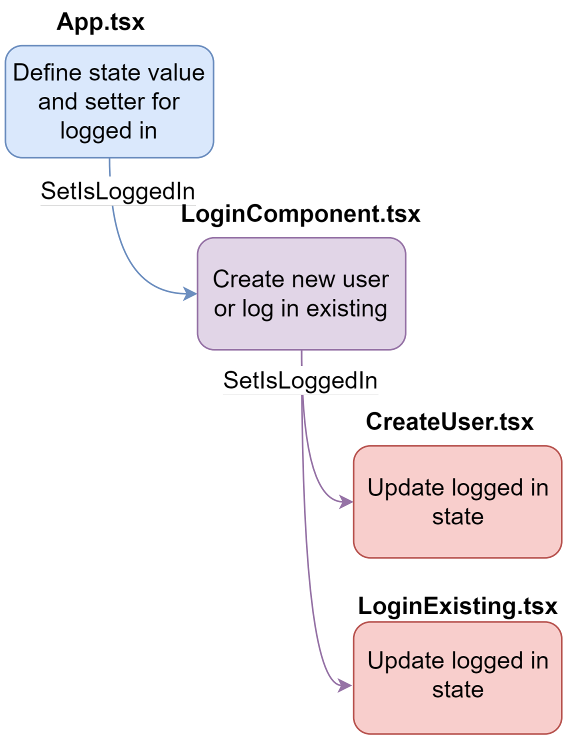


Figure 2.2: Prop drilling

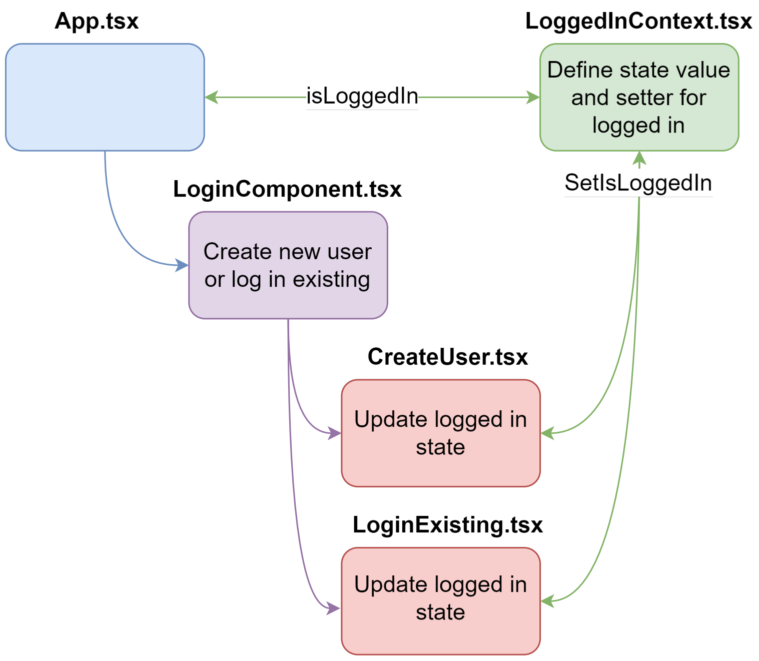


Figure 2.3: Consuming state from context.

3 Method

This chapter explains the methodologies that are used to implement and test FysBot. Section 3.1 presents the collaboration FysBot has with a PhD project. Section 3.2 introduces the methods used in the design of FysBot and is followed by the methods used for testing FysBot on actual users in section 3.3. Lastly, section 3.4 explains the ethical concerns relating to testing FysBot with users.

3.1 Collaboration

The system created in this master project is intended to be used in a PhD project, and as such, the PhD student could be considered a customer of the system. She has knowledge about the users and what they desire in a physical activity chatbot application based on her ongoing research. Her findings about potential users' preferences, as well as the findings in the literature review performed in the capstone[9], have motivated the implementation of the features that now exist in the application. Since this master project has a "customer", frequent meetings are important to give updates on the implementation progress in order to get feedback on what works and what does not. To facilitate this, we set up an entire day each week to work together, in addition to the regular supervision meetings.

3.2 Software Development

3.2.1 Agile Methodologies

Agile methodologies focus on collaboration, iterative development and adapting to frequent requirement changes[27]. This methodology is appropriate for this project due to the close collaboration with the PhD project and the user test that will be conducted. Both the collaboration and the test will lead to new or changing requirements as the project progresses, necessitating quick adaptation to these updates. To help define an initial set of requirements, personas and user stories has been used and are introduced in sections 3.2.2 and 3.2.3 respectively.

3.2.2 Personas

Personas are descriptions of fictional people based on the characteristics of the intended user base. The personas are used as a tool to derive the different features required in the system and evaluate whether the system design is user-friendly and delivers the intended value. Ideally, multiple personas should be defined with different background stories, genders, and skills. The following list contains the most important aspects of a persona that must be covered in the persona's description.

- Age
- Gender
- Occupation
- Location
- Education level
- Personal and professional details
- What is the user trying to achieve with the product
- Pain points that the user faces in their daily lives
- Motivations to use product
- How is the user currently addressing their problem

- Technology usage
- Proficiency level in relevant areas
- Personality

3.2.3 User Stories

User stories are short and informal descriptions of what a user would like to achieve while using a product. They are implemented using the following steps:

1. **Role**. The user of the functionality states the specific position they find themselves in. Roles always start with "As a".
2. **Action**. The action that the user wants to execute is the piece of functionality that must be implemented. Actions always start with "I want".
3. **Value**. What is the value the user gains from executing an action? Values always start with "So that".

User stories help in all stages of the development process, from designing the product, implementing the features, and finally, validating that the product matches the user's expectations. It is a tool used to guide the development into creating something that the user actually wants and avoids features that are "cool" but do not benefit the user. Since the target population is mostly individuals living with obesity as well as other related health condition, the roles were centered around this.

3.3 Usability Study

A usability test was conducted using the think-aloud method to test the developed physical activity app prototype. In the think-aloud method, participants were monitored while they performed a set of predefined tasks[28]. These tasks were centered around the different functions in the app in order to discover patterns in the users' decision-making process. Through this process, one can gain insight into areas the participants think need improvements, such as missing functionality or something that could be made more intuitive. Validating design choices in a test of this sort could be crucial for the success and effectiveness of the application.

In the think-aloud method, it is important that the participants continually verbalize their internal monologue and explain their decision-making process while performing the tasks. When a participant finds something to be unintuitive while solving a task, they should state their difficulties and, where necessary, make suggestions on what would make solving the task easier.

Friends and colleagues at UiT The Arctic University of Norway, and the Norwegian Centre for E-health Research were invited to participate in the think-aloud test. Participation was voluntary. Before the test, participants had to sign an information letter (appendix A) explaining the reason for the test and their rights relating to data privacy. Furthermore, participants were informed that the think-aloud test would be recorded. The set of tasks given to the participants is shown in table 3.1. The tasks were used in a schema by the observers of the think-aloud test where the following key points are noted: *Time*: the time to complete the task in minutes, *Completed*: whether or not the participant was able to complete the task (Y-Yes; N-No), and *Errors*: the severity of any identified errors were noted (C-critical; NC-Non-Critical). The tasks were executed with minimal intervention, but the participant could freely ask questions or ask for assistance. For participants who had never participated in this type of test, a demonstration using a different app was given to help them better understand what was expected of them.

Table 3.1: Think aloud tasks

Task
Download and install the FysBot app
Sign in to FysBot - create a username and password
Give the chatbot a name
Set step goal
Start chatting with the chatbot
Find and read information about FysBot
Ask the chatbot to recommend exercises
Change step goal
Make a physical activity plan for the day
Ask the chatbot to give feedback on steps

Ask/check for overview of physical activity

Ask the chatbot a physical activity related question

Ask the chatbot a general question

Check the overview of steps

Check how many badges you have received

Ask the chatbot to suggest a hike

The app was released in test mode for the usability test, requiring adding participants' Gmail accounts to a list of test users. These were collected orally. All tests were audio recorded on an offline device. Voice recordings were given anonymous identifiers, transcribed verbatim and then deleted. The data were then analyzed and used to improve the application's usability.

3.4 Ethics

3.4.1 Test Approval

Approval for executing the user testing was granted by Sikt - Norwegian Agency for Shared Services in Education and Research. This is required as the application could potentially collect and process personal information. To get the approval, a notification form is filled that explains the following:

- Usage of application
- Type of data collected
- How the data is collected
- How data is secured

In addition, an information letter explaining the goal of the test, the participation requirements, and the participants' data privacy rights was written for each user sample. The notification form, information letter and the Sikt assessment are in appendix A.

3.4.2 Data Privacy and Protection

An emphasis has been placed on protecting the privacy of the user data collected through app usage and user tests. This data includes, for example, location, message history and voice recordings.

In the application system, a specific area of concern is the usage of ChatGPT, which has also been highlighted by Sikt. OpenAI is primarily based in the US, and there is a lot of scepticism about transferring user data to a data processor or data storage outside of the EU. However, OpenAI provides ChatGPT models in different regions through the cloud provider Azure, one of them being Sweden, which is used by the system implemented in this project. While using ChatGPT in Azure, OpenAI states that they will not have access to the data, and it will not be used to further optimize OpenAI models[29]. OpenAI also states that they will store inputs and outputs for 30 days in the region where the ChatGPT model is used for the purpose of monitoring abuse[29]. By this, it can be concluded that no ChatGPT-related data will ever leave the region from which the model is used. More information on where and how data is stored is explained in section 4.2.8. Chapter 5 goes into more detail on how the user is protected while using the app, including which data is sent to ChatGPT, secure communication, authentication and authorization.

4 Requirement Specification and Design

This chapter is split into two parts. The first part, section 4.1, presents FysBot's functional and non-functional requirement specifications that were devised using personas and user stories. Section 4.1.1 presents the functional requirements and ends with defining a set of key focus areas as well as a mockup to guide the visual design. Section 4.1.2 presents the non-functional requirements that must be in place for the usability study.

The second part presents the design of FysBot, beginning with choosing the right LLM in section 4.2.1. The following section, 4.2.2, introduces the chatbot personas, which are entities implemented to handle different topics of conversation and are used to provide personalized responses. The different personas are further described in section 4.2.3. Sections 4.2.4, 4.2.5 and 4.2.6 present the design of the goal setting, badges, and daily schedule features, respectively. Section 4.2.7 presents the design and design obstacles related to collecting user steps. Section 4.2.8 presents FysBot's architecture with a graphical explanation of the backend server and database schema. Finally, section 4.2.9 presents a revised architecture that is not used in the current implementation of FysBot but could be used to increase FysBot's scalability.

4.1 Requirements

The functional requirements were developed using personas and user stories, with priority given to those that provide value to the user and were feasible to implement in the given time frame. The non-functional requirements were based on the need to protect users' data during the usability test and ensure the reliability of the server throughout the test.

4.1.1 Functional Requirements

Personas

The following describes two fictional personas undergoing treatment for obesity at a health clinic. These personas are representative subjects for the user base that will use the FysBot application and are based on interviews performed by the PhD student with patients at a rehabilitation clinic.

Persona 1 - John

John is a 33-year-old male living in southern Norway who struggles with obesity. He has a high school education and is currently receiving health benefits as he is unable to work because of his health problems. John is single and likes to talk to friends, primarily through gaming. He also enjoys watching movies and listening to music. Currently, John is a patient at a rehabilitation clinic for his obesity.

John does not enjoy physical activity and sits mostly at his computer. He lacks motivation and finds walking just for the sake of moving boring and pointless. He would like to be productive while getting his exercise. John does little physical activity at home, but he would like to be more motivated to lose weight and is addressing this by going to the rehabilitation clinic.

John is proficient with technology and confident using smartphones, computers, and wearables. He does not have much knowledge about physical activity but is enthusiastic about using a chatbot that can help him with physical activity and motivate him to be more active.

Persona 2 - Mary

Mary is a 61-year-old Norwegian who struggles with obesity and depression. She has a bachelor-level education and used to work as an advisor, but she is now out of work and receives disability benefits because of obesity-related problems. Mary is a single mother who lives alone, and her favorite activities are swimming, gardening and hiking.

Mary would like to be more physically active but lacks the motivation to start, especially in the winter months. She has bought resistance bands for working out and has hung them up in a place that makes her notice them every day in an attempt to remind herself to train. She also goes to a rehabilitation clinic to help manage the weight gain.

Mary is not proficient with technology and is hesitant to use new technology. She would like a chatbot that could be a companion and remind her to be physically active. However, it would need to be intuitive and should understand what she is saying. She would like to try the chatbot first to determine if it can help her be more active.

User Stories

In a published paper about the user base[30], a set of desired features were gathered from individual interviews and focus groups using a semi-structured interview guide. Using this information, along with the personas, several user stories have been defined and are presented in table 4.1 (US: User Story). These will be used to guide the design of FysBot.

Table 4.1: User stories

NO	User Story
US1	As an individual living with obesity, I want to get recommendations for exercises so that I can have more variety.
US2	As an individual living with obesity, I want to receive encouragement to work out so that I feel motivated.
US3	As an individual living with obesity, I want to be reminded to work out so that I don't forget.
US4	As an individual living with obesity, I want to be reminded to work out so that I get pushed to be more physically active.
US5	As an individual living with obesity, I want to create a structured plan for my day so that I can keep promises to myself to exercise and eat.
US6	As an individual living with depression, I want a companion to talk to so that I feel less lonely.
US7	As an individual trying to live healthier, I want to be rewarded for my efforts so that I feel motivated.

US8 As an individual trying to live healthier, I want to be able to talk to other people in similar situations so that their progress motivates me.

US9 As an individual trying to live healthier, I want to be able to work out with other people so that physical activity becomes more social and thus easier to accomplish.

US10 As an individual trying to live healthier, I want the chatbot to be aware of my physical activity so that I can get recommendations for how to progress.

US11 As an individual living with a disability, I want to be able to talk to a health professional if the chatbot does not know the answer so that I can get the right counseling.

US12 As a chatbot user, I would like information about how to use the chatbot so that I can have productive conversations.

US13 As an individual with sight problems, I would like text-to-speech so that the chatbot is more accessible.

US14 As an individual with sight problems, I would like speech-to-text so that I can easily interact with the chatbot.

Features for Motivating Physical Activity

Based on the user stories listed in 4.1, the following set of features have been decided as key focus areas for the application and are the functional requirements for FysBot. The decisions are based on their value to the user base and on what is realistically feasible.

- **Exercise recommendation**. The user should be able to receive personalized exercise recommendations through interactions with the chatbot (US1).

- **Outdoor activity recommendation**. The user should be able to receive activity suggestions based on the user's current location (US1).

- **Progression review**. The user should be able to review their progression and get suggestions on increasing daily activity (US2, US10).

- **Scheduling**. The user should be able to set a daily plan and have the option to be reminded of selected events (US3, US4, US5).
- **Open conversation chat**. The chatbot should be open to a multitude of conversation topics (US6).
- **Reward system**. The user should be able to set goals and be rewarded for their efforts (US7).
- **Guide for using chatbot**. The user should be able to receive information on how to use the chatbot (US12).

The remaining user stories, 8, 9, 11, 13 and 14, are not scheduled for this master thesis. Stories 8 and 9 relate to a feature incorporating a social platform into the application. With the given time frame, this feature is not prioritized as other features are thought to add more value. In user story 11, there is a desire to communicate with a health professional if the chatbot is not able to satisfy the user. This could be a great feature, but its value is thought to be first realized in a production setting, so it is not a priority in the prototype. User stories 13 and 14 express a desire to converse with the chatbot using voice. Speech-to-text and text-to-speech synthesis is, unfortunately, too expensive for this project.

User Interface

The previous sections have identified the features that could be included in FysBot, and from this, a mockup of FysBot's user interface is presented in figure 4.1. This mockup is used to help visualize the functional requirements.

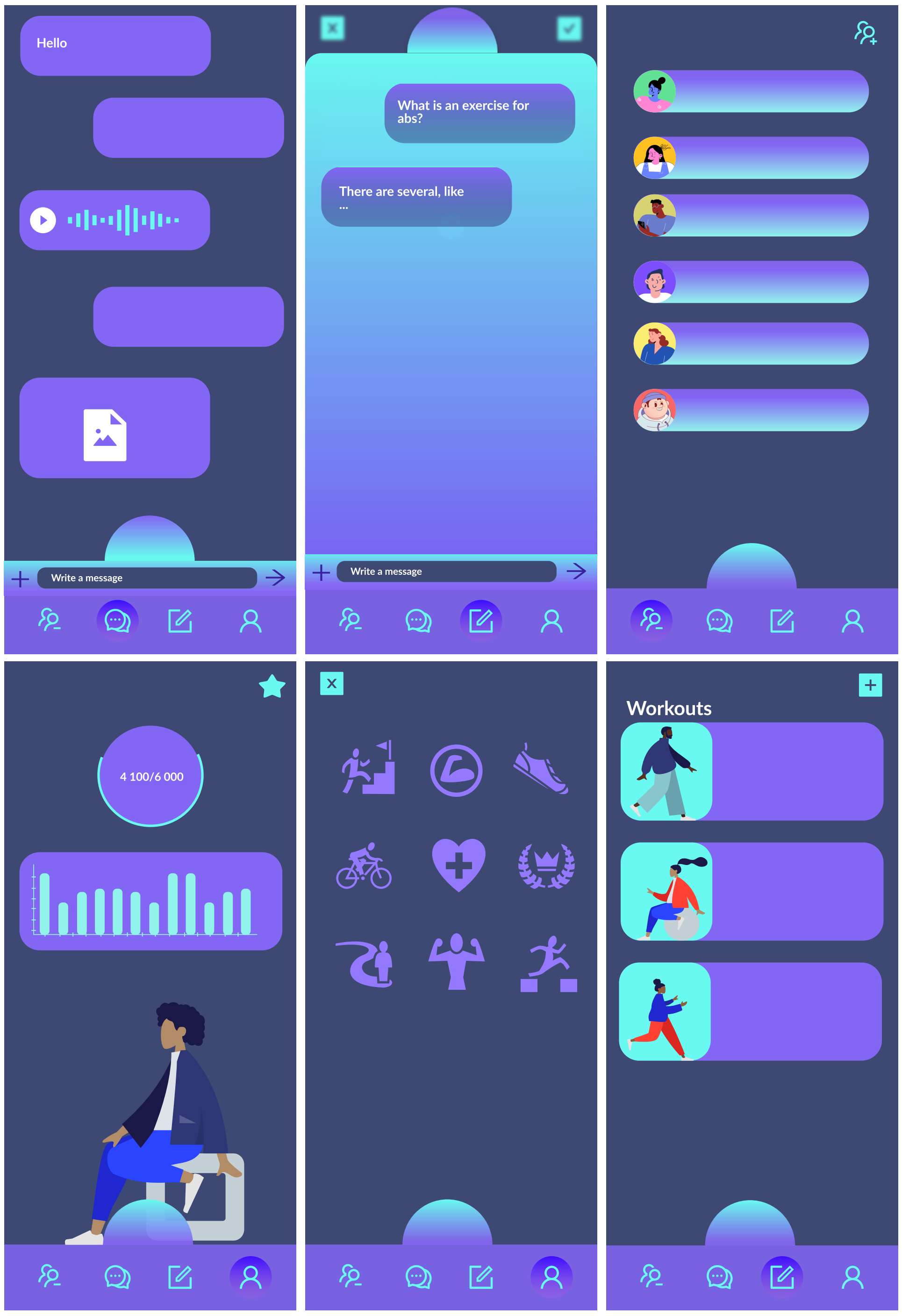


Figure 4.1: Mockup of app design.

4.1.2 Non-Functional Requirements

For the prototype of FysBot that will be tested in the usability study, there are three main non-functional requirements: security, reliability, and performance. Since the application could probably process sensitive data, a set of security mechanisms has to be in place to protect the users' data and privacy. For the usability study, the backend server hosting the AI model must be available during the study. This leads to the following non-functional requirements:

- **Secure traffic**. The traffic between the client and server, as well as between the server and other APIs must be secure (security).
- **Secure API endpoints**. It should not be possible for unauthorized users to retrieve data from the server. Additionally, users should not be able to retrieve personal data about other users (security).
- **High availability**. The system must be available throughout the usability study (reliability).
- **Adequate response time**. The system should consistently respond promptly without significant delays, ensuring that the quality of the usability study is maintained (performance).

4.2 Design

4.2.1 Choosing the Right LLM

The chatbot will be powered by an LLM as it will give the user the most human-like experience and make extracting information from free-text conversations easier. There are different design options to evaluate when choosing an LLM, and the most suitable solution will depend on the severity of inaccurate outputs, response time, response quality, time to implement and possibly most importantly, the cost of using the LLM.

With the dedicated hardware, a good option could be to host the LLM on-premise. There are several available open-sourced models in different sizes. Smaller-sized models are not considered an option for this application's purposes, as the chatbot responses will not be of high enough quality to be perceived as human-like. Larger models are also not an option as the dedicated hardware is not available to us on-premise, and renting cloud hardware is too expensive. Not considering the cost, the time and effort spent setting up a larger model like Metas' LLaMA[31] model will be extensive.

Since running the LLM locally is not an option, the best solution appears to be the ChatGPT models, accessed via the OpenAI API. This will provide a powerful model at an affordable price and several out-of-the-box features. The model chosen is the ChatGPT-3.5-1106 version, which will be referred to as just "the model" from here on.

4.2.2 Handling Different Topics of Conversation Using Chatbot Personas

To create a chatbot that works well for different topics, splitting the responsibilities of the chatbot into several entities, each specialized for a specific topic, could be one approach. These specialized entities have been given the name *personas* and are inspired by LangChain's *chain*[32] structure used to simplify the development of apps using LLMs. The topic could, for example, be exercise recommendation, for which the persona responsible for exercise recommendation will have the task of analyzing which exercise(s) the user is most likely interested in and providing the appropriate response. There could also be scenarios where one persona 'talks to' another persona in order to complete the user query.

The reasoning for this approach is that the context size of the model is limited, but also, the model would most likely become confused if given too many instructions at once. The instructions often include how to respond and/or parse a specific message. One can imagine that placing all the instructions for every topic in a single message, like the example message of figure 2.1, would make it difficult for the model to perform. Instead, one persona called the router persona, acts as a mediator between the user queries and specialized personas.

4.2.3 Design of Chatbot Personas

Router Persona

The Routing Persona (RP) is the persona that first receives a query and is responsible for routing the query to one of the other personas. As shown in figure 4.2, the RP is given a description of the capabilities of each persona and must evaluate the incoming query to find which description best matches the query. The RP will then forward the query to the selected persona.

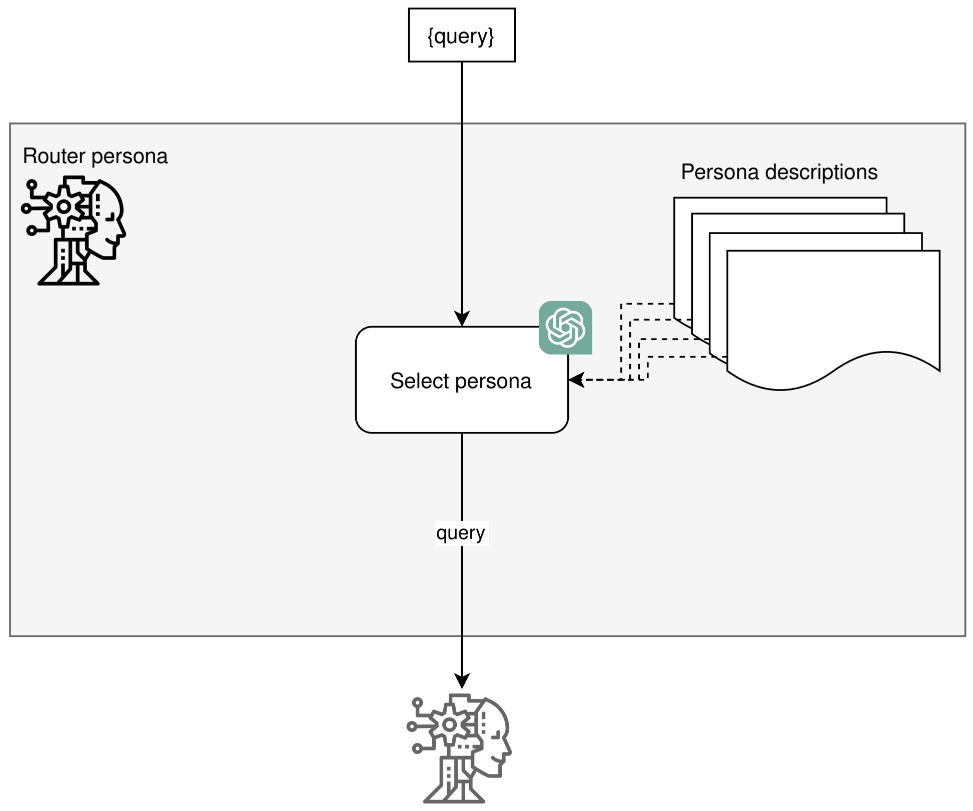


Figure 4.2: Router persona

Exercise Recommendation Persona

Exercise recommendation is a feature that allows the user to receive a selection of exercises from the chatbot based on the parameters: body part, type of exercise (ex., cardio or strength), difficulty level and available equipment. The data needed to accomplish this is taken from the Github repository "free-exercise-db"[33]. This repository contains over 800 annotated exercises with accompanying images and instructions. Since the user base for the application is primarily Norwegian speakers, the instructions (originally written in English) have been translated into Norwegian. The translation was done by implementing a small script that used the OpenAI API with ChatGPT as the translator.

Exercise recommendation is made via the Exercise Recommender Persona (ERP) after being forwarded the user message by the router persona. As shown in figure 4.3, the first step is to parse out the relevant keywords from the message. This is done via ChatGPT function calling, where the model is given a description of each keyword that could exist in the message, such as the exercise name, body group, exercise equipment or exercise level. An example of a possible message is "Could you recommend some easy exercises for biceps using dumbbells?", for which the ERP would parse out: {exercise: null, bodyGroup: bicep, level: beginner, equipment: dumbbells}. Questions with the exact semantic equivalence will result in the same parsed output.

The ERP is designed such that when the user has a specific exercise in mind, it will return the instructions as well as images for that exercise. Otherwise, it will list a set of suggestions based on the parameters the user has included. Any combination of parameters can be used.

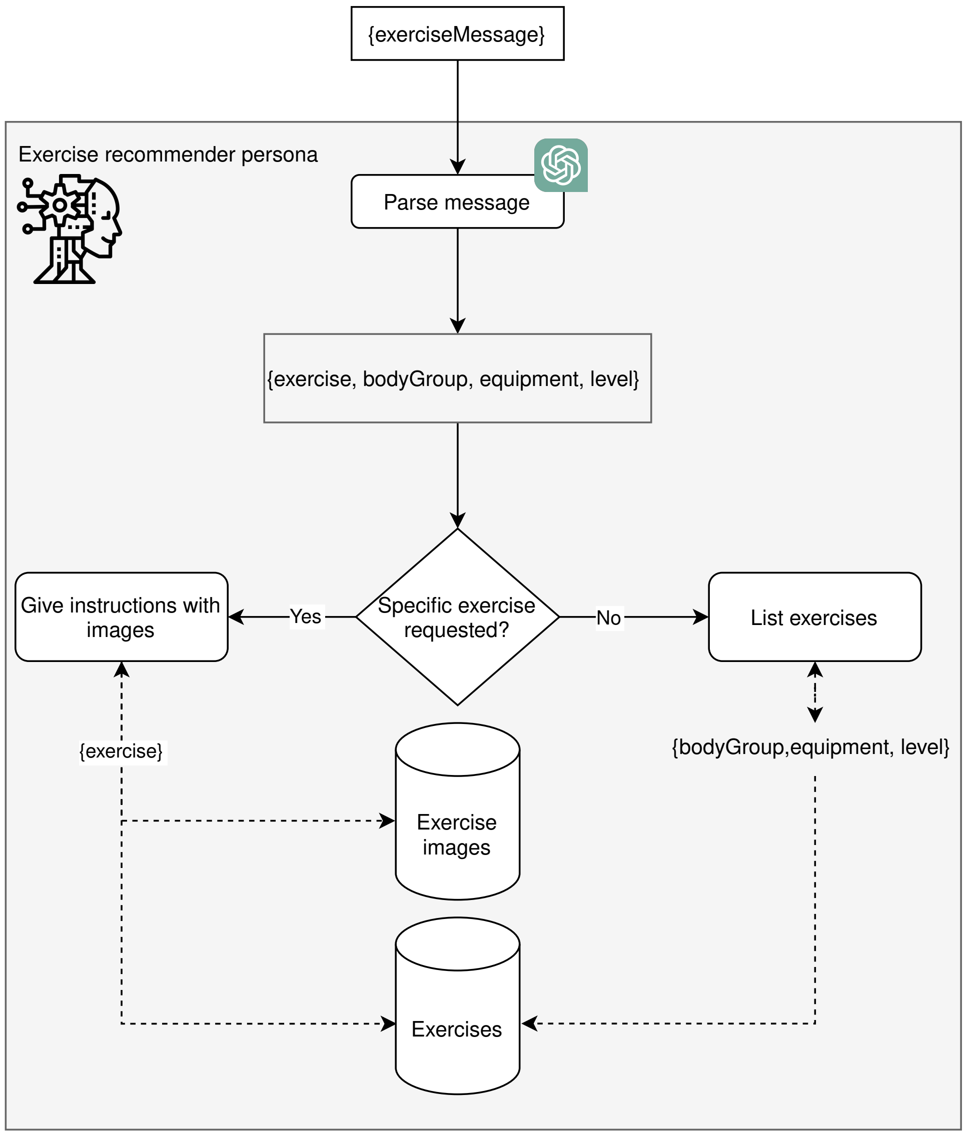

{exerciseMessage}
Exercise recommender persona
Parse message
{exercise, bodyGroup, equipment, level}
Give instructions with images
Yes
Specific exercise requested?
No
List exercises
{exercise}
{bodyGroup,equipment, level}
Exercise images
Exercises

Figure 4.3: Exercise recommender persona

Progress Review Persona

The intention of the Progress Review Persona (PRP) is to give the users an overview of their step history by comparing their daily steps to their weekly, monthly and yearly step averages.

When reviewing the steps, it is, of course, necessary to know the user's step history. With this system, the topic of the user query cannot be known until it is reviewed by the RP. Therefore, it is difficult to ascertain whether the steps should be included in the client's request. A solution is then needed to fetch the steps when the user query is directed to the PRP. This could be done using the periodically sent steps explained in section 4.2.7. However, more up-to-date steps are needed to get the most accurate response. A better approach for progress review is to do conditional fetching where the step records are only provided when needed.

When the user query is sent to RP, and it is determined that the query should be forwarded to the PRP, the PRP initially verifies if the necessary step records are included in the request. If they are absent, the server responds with a request to include this data. This is illustrated in figure 4.4. Subsequently, the client, awaiting the response, will conditionally send a new request with the step records if necessary.

Location Activity Persona

To motivate physical activity, finding ways to do PA should be as easy as possible. FysBot's solution is to create a Location Activity Persona (LAP), as shown in figure 4.5, which recommends activities based on the user's current location. For now, the activity the LAP will recommend is nearby hiking trips or walks, but several other possibilities could be implemented into this feature. The starting coordinates of a set of hikes and walks have been obtained from ut.no[34], which uses data from Kartverket and other sources and is stored in FysBot's backend database.

Whenever a user request is forwarded to the LAP by the Routing Persona, the same conditional fetching strategy used in the PRP is applied to resend the user message, but now also including the user's location. A mathematical function is then applied to determine the distance between the user and the activities, and the ones within a set radius are recommended to the user. This radius is currently set to five kilometers, but the feature could easily be extended to allow the user to specify the maximum distance in the request.

Another feature of the LAP is that it is programmed to give the user personalized

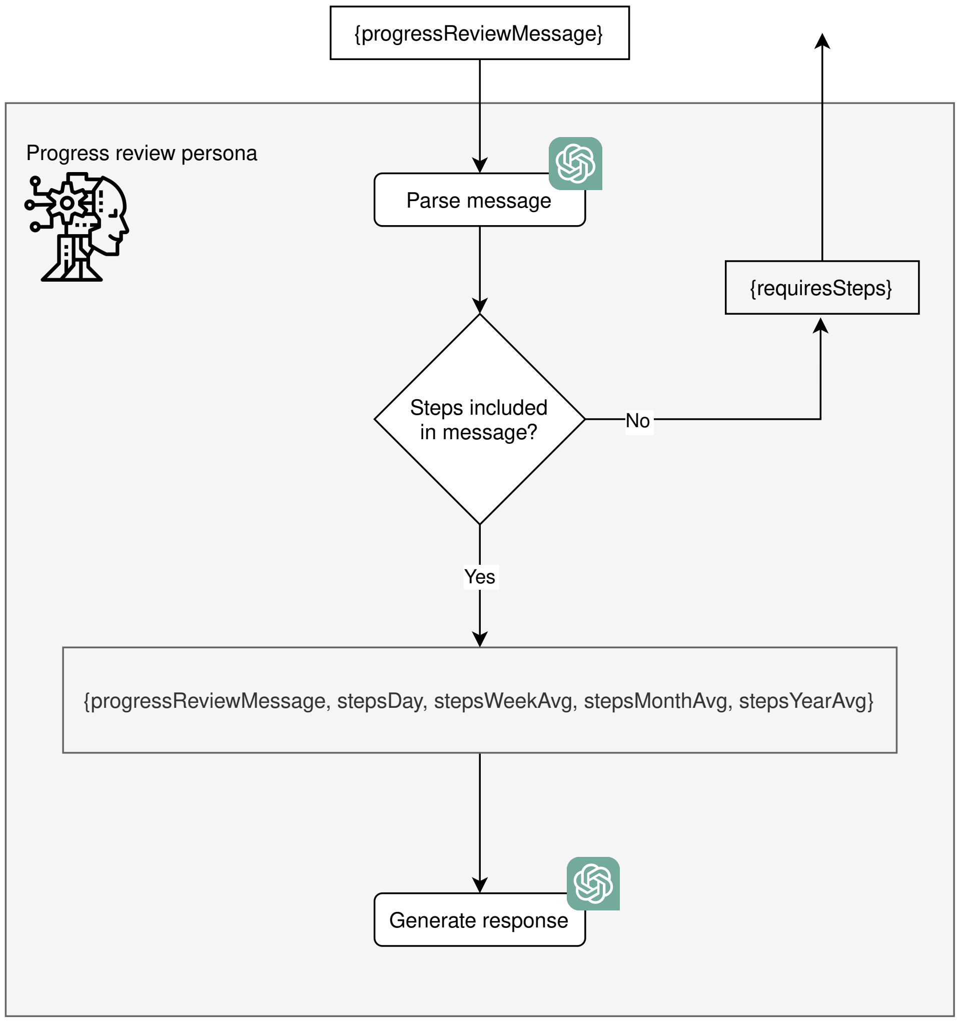

{progressReviewMessage}
Progress review persona
Parse message
{requiresSteps}
Steps included in message?
No
Yes
{progressReviewMessage, stepsDay, stepsWeekAvg, stepsMonthAvg, stepsYearAvg}
Generate response

Figure 4.4: Progress review persona

recommendations based on the weather conditions to help them prepare for outdoor activities. The weather conditions are fetched using an API provided by Yr[35], from which the current temperature, wind speed, and a summary of the weather conditions expected in the next hour are used. The next hour's summary could state, for example, "rain" or "cloudy", which is useful when crafting the recommendations. These properties are then fed to the LAP, and the model creates a weather-based physical activity recommendation and the necessary preparations for the user.

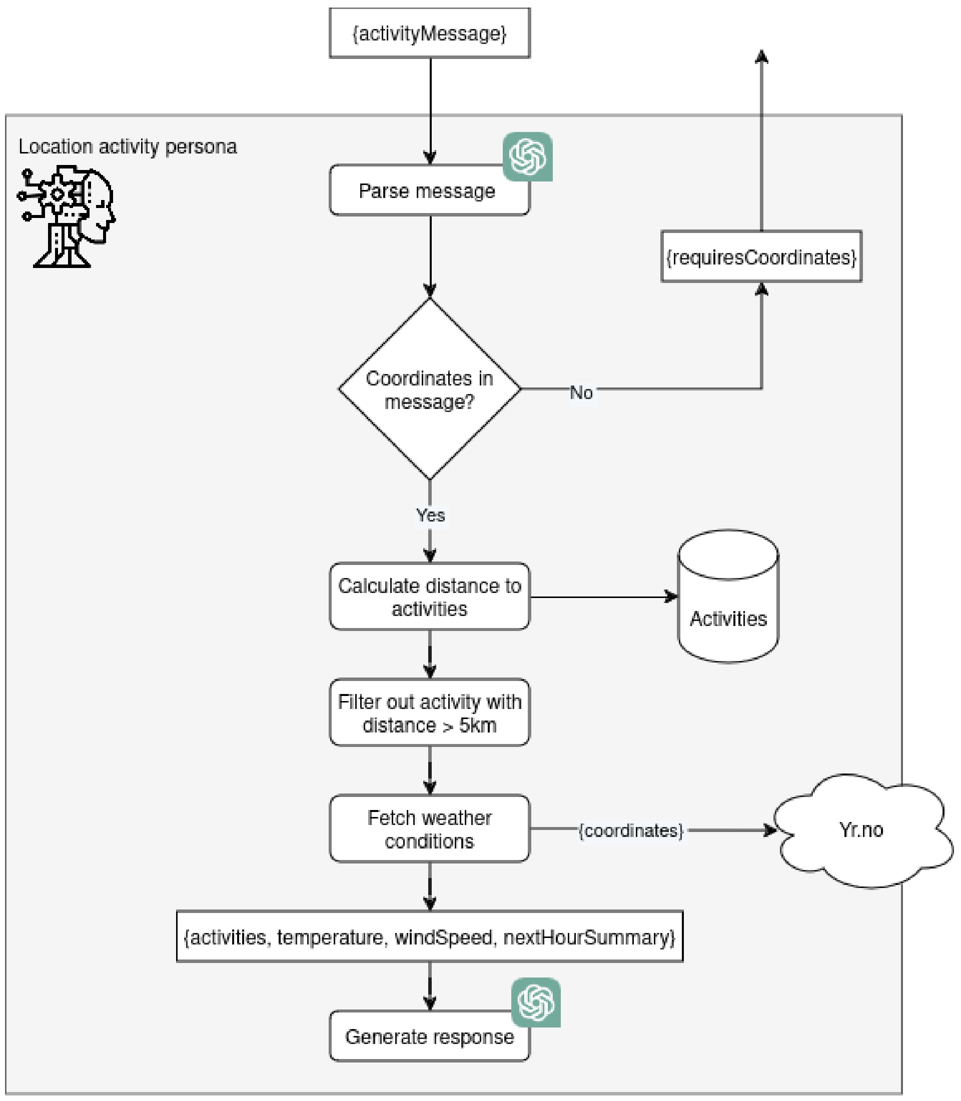

{activityMessage}
Location activity persona
Parse message
{requiresCoordinates}
Coordinates in message?
No
Yes
Calculate distance to activities
Activities
Filter out activity with distance > 5km
Fetch weather conditions
{coordinates}
Yr.no
{activities, temperature, windSpeed, nextHourSummary}
Generate response

Figure 4.5: Location based activity recommendations

Memory

Each persona has its own memory for storing conversations with the user. This is done to more efficiently manage the context in the conversations and attempt to give the user a more accurate response by providing continuity. The design challenge this presents is managing the conversation messages linked to a particular user and persona. One approach is to let the client manage the messages that should be sent to the server, which would require sending large message blocks to the server for each chat request. Instead, the approach used is to store the messages in the server database, where messages can be retrieved based on username and persona name.

4.2.4 Goal Setting

In the initial design of the goal-setting feature, the user was supposed to be able to set goals in free text by stating the goal directly to the chatbot. The chat model would then parse out the goal type, which could be, for example, a step goal or a weight-loss goal, and then parse the associated value (number of steps or number of kilos). The motivation for this approach is to enhance user experience and human-like interaction with the chatbot. Although feasible, getting it right requires a good amount of testing. Furthermore, it would be harmful to the user experience if the chat model could not parse a more unconventional goal setting. A solution would be to limit the user to a predefined set of goal types. However, it was concluded that the most suitable approach would be to give the user a menu of goals to choose from. This solution might make it easier for the user to set goals and, in addition, make it possible to include and obtain data on the goal types stated as outcomes in the PhD project.

4.2.5 Badges

Badges are earned by completing specific step goals, and the progress towards a goal is constantly monitored by the app, but only while the app is running. The badges earned by the user are stored locally on the user's phone. Whenever the application is opened after being closed, synchronization of the badge progress is required by reviewing the user's step history from previous days. The time of the last sync is also stored on the user's phone and will be used as the starting point. If, however, there has been no previous sync, as will occur the first time the user opens the app, the time of the app installation is used instead. The sync will run every 5 seconds.

The synchronization process found in figure 4.6 starts by retrieving the last sync date, as shown. Then, for each day up to the current day, the badges the user should have earned are found and matched against the badges the user has been rewarded. If they match, the last sync date is set to the current day, and the iteration jumps to the next day. If not, the user is first awarded the previously unrewarded badges. The badges are stored in a minimal SQL database on the user's phone.

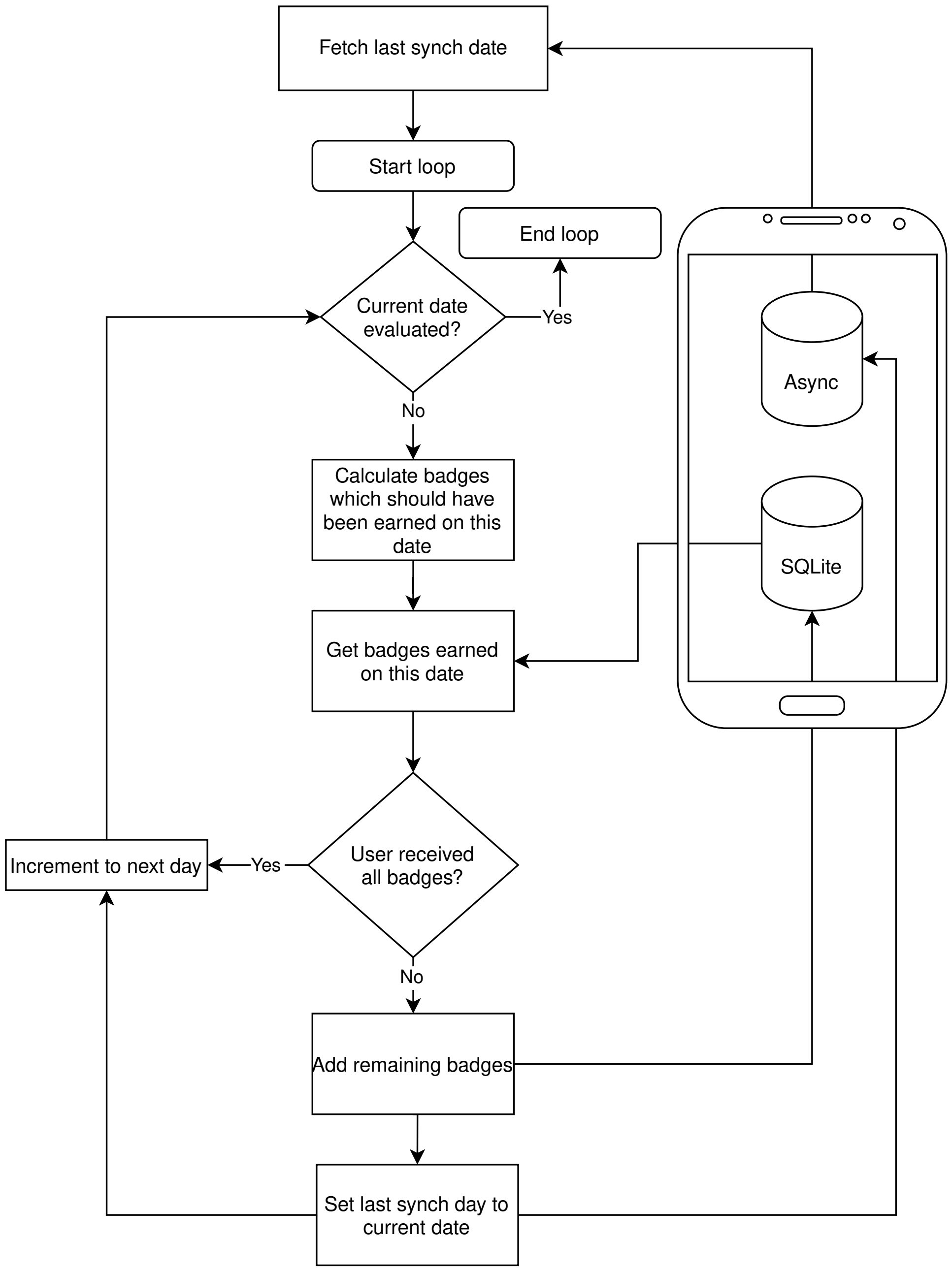

Fetch last synch date
Start loop
End loop
Current date evaluated?
Yes
No
Async
SQLite
Calculate badges which should have been earned on this date
Get badges earned on this date
User received all badges?
Increment to next day
Yes
No
Add remaining badges
Set last synch day to current date

Figure 4.6: Synchronizing badges

4.2.6 Daily Schedule

From section 4.1.1, it was identified that potential users might want to set a structured plan and get reminders for events like workouts. A scheduling feature is implemented for this purpose, where the user can create events and have the option to get reminders for specific events that are sent as push notifications.

A notification on an event is sent 30 minutes before the time of the event. For this to be feasible, each user's scheduled events must be stored in the backend server. At a frequent interval, the server then checks for any scheduled events in the next 30 minutes. A possible solution to this is to store the entire schedule of each user separately, then iterate over each user and their schedules. This is highly inefficient, so instead, each singular event of every user is stored in the same table. Sorting this table by event time makes it easy to check whether any events occur in the next 30 minutes. If events are identified, the notifications are sent, and the corresponding row is deleted. Figure 4.7 illustrates the process.

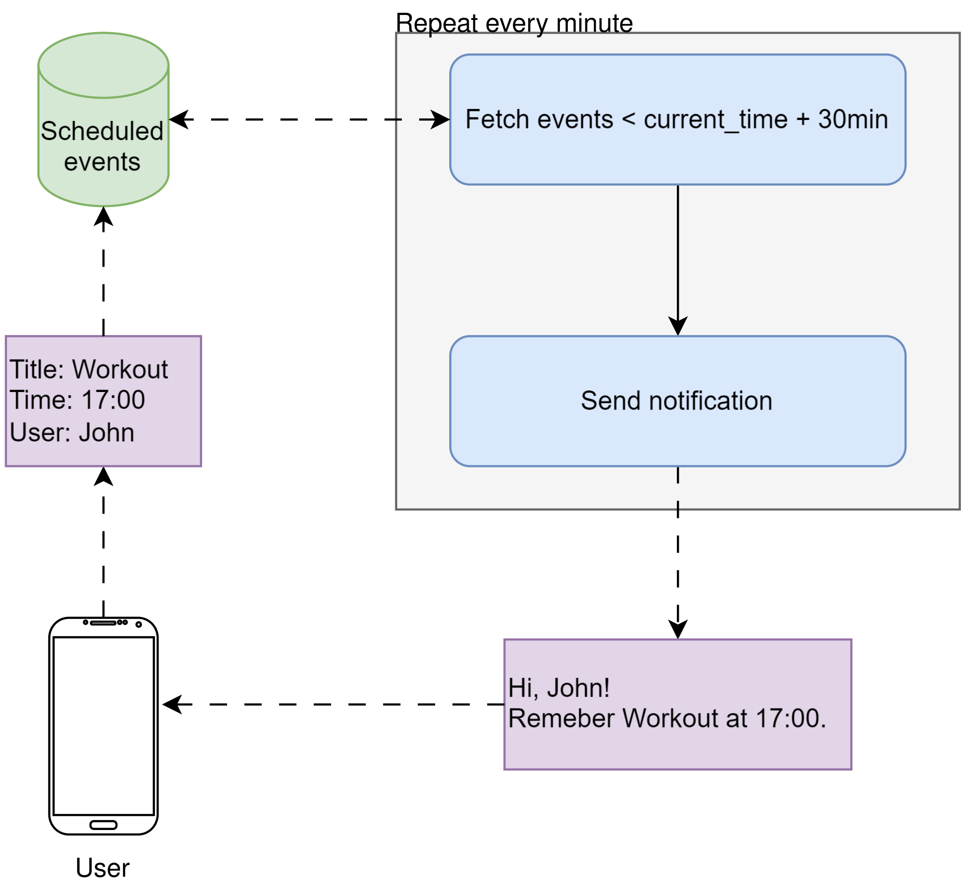


Figure 4.7: Sending notifications of scheduled events.

4.2.7 Step Collection

The FysBot application can fetch steps through Health Connect and Google Fit. To use Google Fit, the user must have it installed and log into their Google account when using FysBot. Google requires the login to authorize the user and give consent. Step data from Google Fit can be used without further verification while FysBot is released in the testing stage. For an open release of FysBot, the app must be verified by Google, which involves demonstrating the app and how the health data is being used.

Steps can also be accessed via Google's Health Connect application, which is a hub for different health apps to share data. The Health Connect application is installed by default starting from Android 14. This is currently the latest version of Android, and the usage of Android 14 is currently around 16% at the time of writing[36]. For users with versions below Android 14, the Health Connect application has to be manually installed, or the user could use Google Fit instead. Using the data from Health Connect requires a complete Google verification process that involves demonstrating the app and how the data is used. This applies to both test releases and open releases of FysBot.

Collecting steps is an integral part of further research on whether using FysBot increases a user's physical activity levels. It is also required to update the user on their progress and to give rewards and positive feedback. Feedback, given in the form of push notifications, can update the user on their progress towards achieving their goals or congratulate them when a goal has been reached. This feedback should be provided even if FysBot is in the background or closed. Mechanisms have been implemented for this that attempt to wake the application every 15 minutes to supply the current step count, which is then used to evaluate the user's progress. Due to operating system restrictions, a higher frequency is not possible.

4.2.8 Backend Architecture

Backend Server

The FysBot architecture consists of a virtual machine deployed in Azure in the Norwegian region and an Azure deployment of ChatGPT located in Sweden, as shown in figure 4.8. Several services are defined in the VM, and all contact one or several tables in the database. For simplification, a single dashed line shows the communication between the services and the database. The figure does not show communication between services, such as the scheduling service contacting the notification service to notify the user of upcoming events.

The backend server is designed using a proxy at the front, which terminates the TLS communication and forwards the request to the API controller after it has been authenticated. A TLS-certificate system is attached to the proxy for generating and renewing the TLS-certificates that enable secure communication. The API controller reads the request and uses one or several services to handle the processing of the request, which again might require accessing the database or external APIs. The AI chat service is responsible for forwarding the user query to the chatbot personas to process the query and generate a response.

Database Schema

An SQL (PostgreSQL) database is chosen as it provides a data model that fits the needs of the application, which almost exclusively executes simple read-and-write operations. In theory, a NoSQL database could be a better fit as the application does not require the strictness of the ACID properties found in SQL databases, which places an overhead on the transaction. Availability and transaction speed are valued more in this context, but with the low number of users expected, the difference between the choices is insignificant. The database schema is shown in figure 4.9. No normalization techniques have been applied.

4.2.9 Scalable Architecture

The system design that has been implemented is for a prototype. This section explores a design that is scalable, has a higher fault tolerance, is more resilient and has a stronger security. Only the most critical components are discussed.

In the revised architecture shown in figure 4.10, the API server and the different services are split into self-contained microservices. To provide a higher degree of availability and fault tolerance, these microservices are replicated and can be scaled either vertically or horizontally. Each service includes health check endpoints to verify the availability of the service. A load balancer sitting in front of the API server then uses health probes to route traffic to replicas that are operational. This ensures that users can still access the system through another replica should one of the replicas crash or overload.

The notification service handles notifications on scheduled events as well as goal completions. In the revised architecture, the notification service places pending notifications on a message queue like RabbitMQ. The notification consumer then handles sending these notifications to the user. Decoupling the

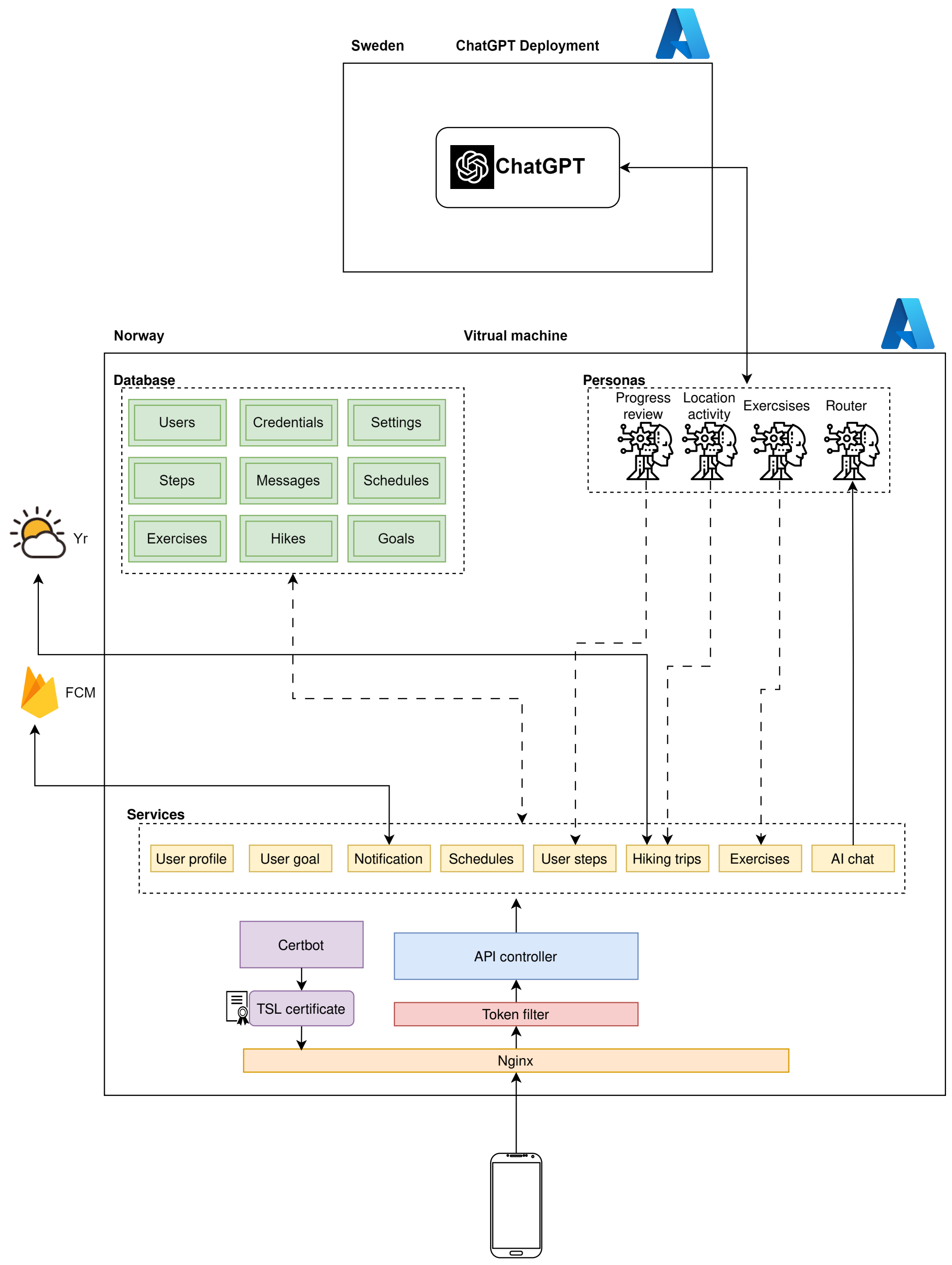


Figure 4.8: FysBot's backend architecure.

notification system in this manner makes it easier to develop and maintain, and the producers and consumers can scale independently. Using a message queue like RabbitMQ makes it easier to guarantee that messages reach the user successfully. It can also dynamically balance the load across the consumers.

The majority of the processing takes place in the ChatGPT service, which can

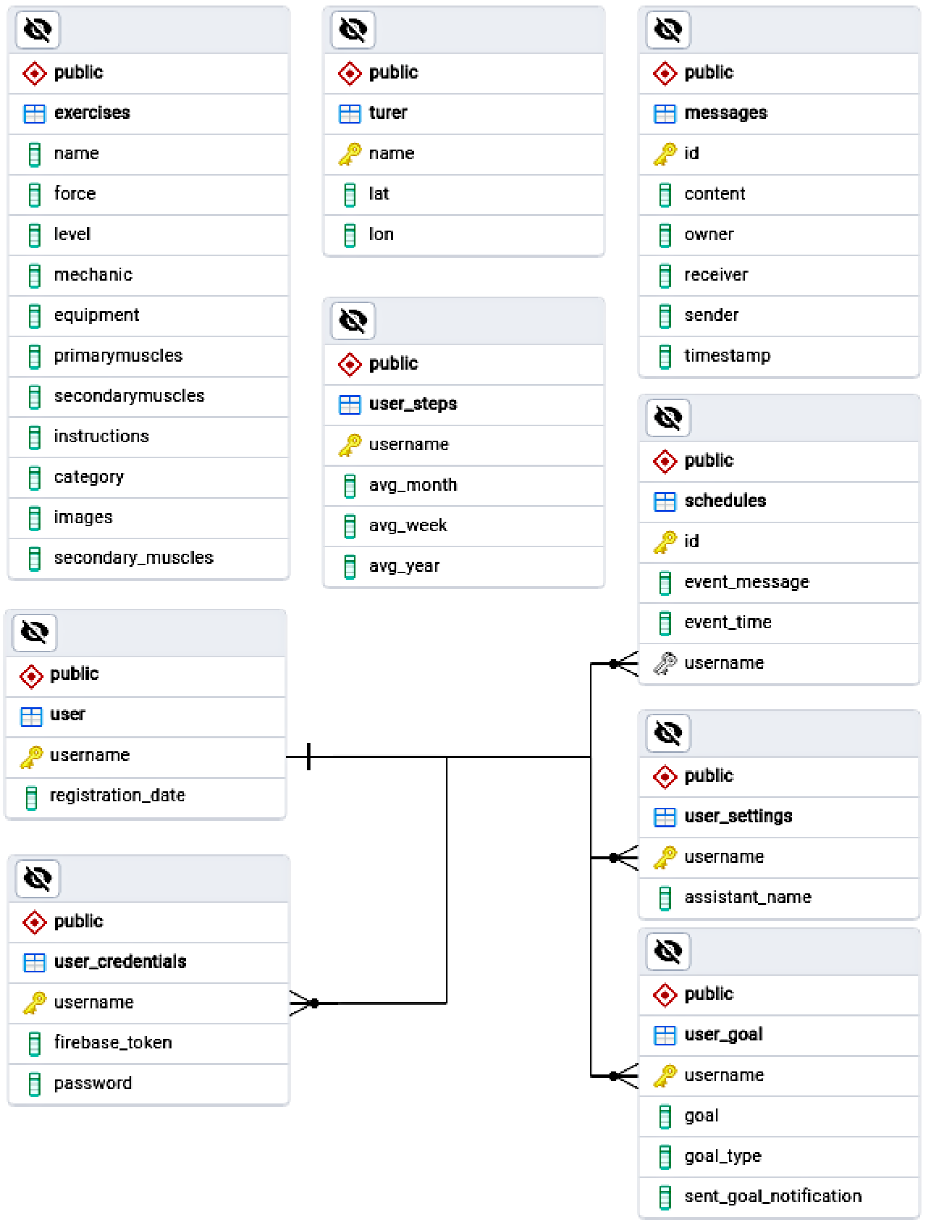


Figure 4.9: Database schema. Diagram is taken from pgAdmin using ERD tool.

scale independently from the rest of the system by being implemented as a microservice. This service will contain the personas and respond to messages sent to the chatbot.

Lastly, there is the authentication service implemented using OpenID Connect (OIDC) provided through Firebase. OIDC provides secure mechanisms for authentication and session management. Like the token authentication mechanisms used in the current FysBot design, Firebase also offers the use of JSON web tokens in their session management. Using a well-established system like OIDC is generally more secure and faster to implement.

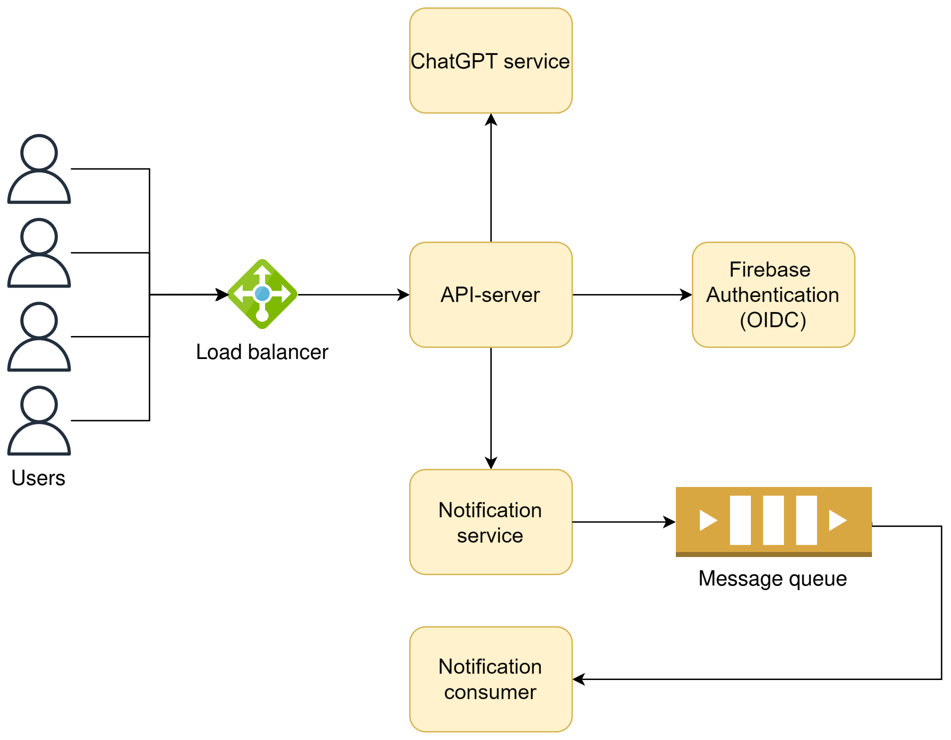


Figure 4.10: Scalable architecture that splits the backend concerns into separate microservices.

5 Security and Privacy

Securing the user's data is a large focus area in this project. The data processed by FysBot is potentially sensitive information and includes messages between chatbot and user, location, steps and goals, and the prospect of using voice. This data must be protected against unauthorized access and should not be shared with others. Section 5.1 provides information learned from consulting a security expert. Section 5.2 explains how the API endpoints are secured. Section 5.3 explains how the users' credentials are secured. Section 5.4 explains how the communication between the client and the FysBot server is secured. Finally, section 5.5 explains how the users' privacy is protected when writing messages to the chatbot.

5.1 Consulting a Security Expert

An initial security design prototype was constructed that focused on the use of tokens (tokens are explained in section 2.5.3) for authenticating the user. A meeting with a security expert at UiT was then scheduled to have the prototype reviewed. The rest of this section goes into detail about several key points discussed in this meeting.

In addition to the token, username and password are essential parts of authentication. Without the ability to refresh a token using the username and password, access to it cannot be revoked should the user's phone be compromised. Without the username, there is no way to know which token is owned by which user, and without the username and password, there is no way to give the user a new token should the user reinstall the app.

The user needs to be informed that their data is being used in queries to OpenAI. Being aware of this, the user must decide whether or not to share sensitive data. Of course, much of the responsibility also falls on the server, which makes the actual request to OpenAI. Instead of providing the complete message history in the requests, a sliding window could be used to only take, for example, the last 20 messages. The size of the window could be tuned by the user, letting the user create a balance between privacy and quality with which the user is comfortable. The requests should not include the user's username unless the user explicitly states it in one of the messages.

To reduce the amount of sensitive information sent to OpenAI, an option could be to identify the sensitive information and run the query on a local LLM. Unfortunately, this is not feasible with the resources in this project. In a case where step counts for individual days are considered sensitive (could be used to track the user's location), another option is to aggregate the steps over several days, in which case it would be difficult to discern specific details about the number of steps. This could then be fed into the query instead of step count for individual days.

5.2 Protecting API Endpoints with JSON Web Tokens

The API server exposes endpoints to, for example, fetch the user's message log. It is crucial that the request to such an endpoint is authenticated, making sure a user is not able to fetch another user's data. A solution to this is to give each user an authentication token when the user starts using the app. This token is then stored locally on the user's phone and included in the request when the user wants to access a protected route. A common approach to this is to create a JSON Web Token (JWT). The JWT defines an access token and a refresh token, where the access token is a short-lived token used to authenticate routes while the refresh token has a longer lifetime and is used to re-authenticate the access token. By using the JWT, the server does not have to be concerned about a user's session, as the required information can be found in the user's token. The trade-off with this approach is that the user must create a username and password, which is required to create new tokens when the refresh token expires. Encrypting the token residing in the user's phone could also be considered for extra security. The trade-off for security is always accessibility and complexity, and additional complexity requires extra implementation time, though necessary to protect the user.

A simpler approach to protecting routes could be using tokens that do not expire in the demo period and are stored as plain text on the user's phone. By default, data stored by an application is not viewable by other applications[37]. This makes sure that malicious apps cannot retrieve data from other apps. The most prominent security risk with plain text tokens is other people getting access to the user's phone. In this scenario, an attacker would be able to view the data stored by the app and find the plain text token. The attacker would then be able to make a request to the API server with the user's token to retrieve the user's data.

Figure 5.1 illustrates the process of protecting the API routes. The security of this depends on the user never sharing their token or password with anyone.

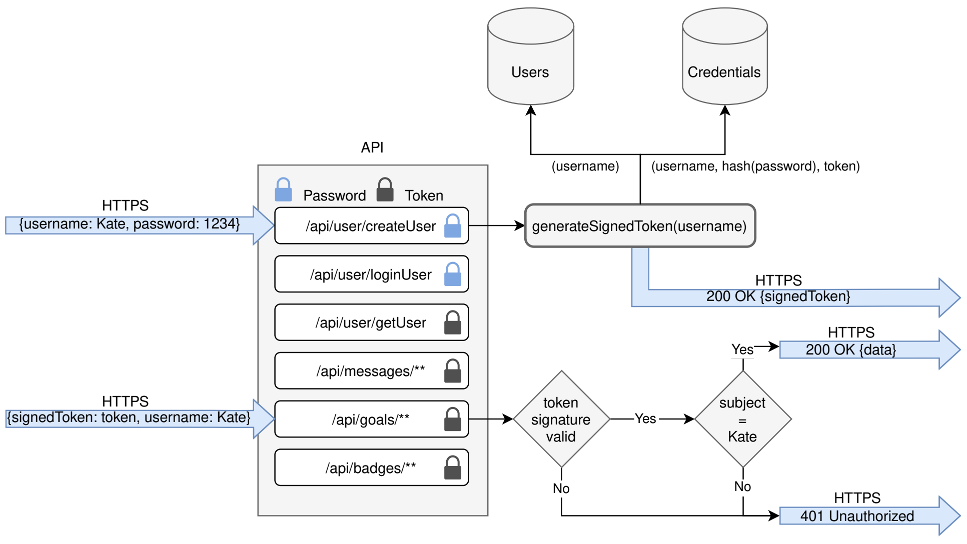


Figure 5.1: Authenticating users.

5.3 Securing Credentials

When the user first opens the app, the user is required to create a profile with a username and password. The server will then store the username and other metadata in a user table and the credentials in a separate credentials table. With this strategy, if the user table is leaked, the associated credentials will not be available to the attacker. There are also several cases where the user metadata is required for some functionality within the client. This strategy becomes a safety measure against a developer who could otherwise accidentally send the user credentials, too. In the latter case, although data transfer is encrypted and authenticated, passing on user credentials unnecessarily should be avoided.

The passwords within the credentials table are hashed, followed by salting. In this case, an attacker would be unable to retrieve meaningful information from the stored credentials easily. Even so, an extra security step could be ensuring users occasionally update their passwords. It should also be noted that this security measure is only efficient if the password is strong and, therefore, difficult to guess to begin with.

5.4 Securing Communication

Communication between the client and the server is encrypted using Transport Layer Security (TLS). A design concern when using encrypted communication is how the backend reads the encrypted requests or whether it should do so at all. Azure provides TLS offloading through a mechanism called Application Gateway[38] that terminates TLS at the gateway, providing the backend services with the decrypted request[39]. This could provide better performance as it removes the TLS processing overhead from the backend servers, and only a single certificate is needed at the gateway instead of one for each backend server[39].

The solution of using the Application Gateway resource was tested and worked as intended, securing the communication to the Azure VM gateway and providing the backend server with decrypted requests. However, the offloading feature provided by Azure comes at a price, and while testing, it was found that it is far more expensive than first anticipated. With limited funding, another solution is needed.

A different approach tested was to fetch the certificate from the Azure key vault and use it in the backend API server to decrypt the messages. The only problem with this is that no solution was found to effectively set and update the certificate used by the VM. Again, it was found that Azure provides pricey mechanisms for handling such cases, which is not an option in this project. This led to exploring options outside the confined spaces of the Azure Portal box and evaluating what could be done manually on the VM. It was then found that the open-source system NGINX has the ability to do TLS offloading at no cost. NGINX acts as a proxy that intercepts all traffic and collaborates well with the free-to-use certificate system Certbot, which provides automatic renewal of valid certificates.

Using NGINX, requests sent by the client will be picked up by the NGINX proxy that uses its configured certificates instead of those configured by Azure. With just a couple of lines in the configuration, TLS offloading is enabled, and the backend is served decrypted requests. Though several solutions were attempted, this ended up being the best solution. In the implemented system, NGINX interacts with the certificate generation system Certbot[1] that generates and automatically renews valid certificates used to enable TLS-enabled communication. Figure 5.2 illustrates how an API request is decrypted by NGINX and authenticated before accessing a resource.

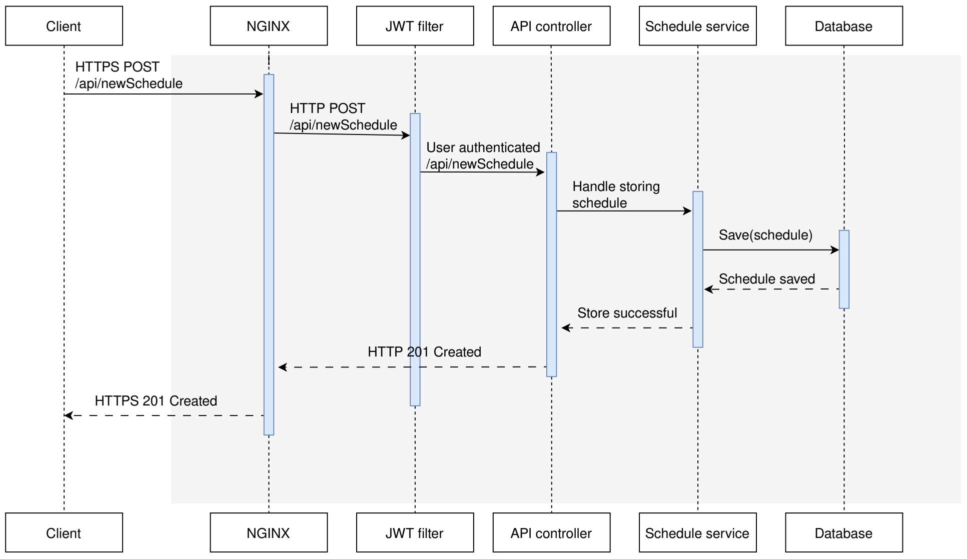


Figure 5.2: Call stack for an API request.

5.5 Protecting Privacy in Chatbot Messages

As discussed in section 3.4.2, queries to OpenAI are only processed and temporarily stored in Azure servers located in Sweden. The message history is stored for longer periods in the Azure VM located in Norway, as explained in section 4.2.8. This data is then subject to the laws of GDPR. To further secure the user's privacy, attempts are made to include as little information about the user as possible in the queries. The following lists data that could be used to personally identify a user and how it could be used in queries:

- **Username**: Never added to the user query.

1. https://certbot.eff.org/

- **Location**: Exact coordinates are never added to the user query. Location is used to fetch names of hiking trips in a large radius around the user and weather conditions, which could be added to the user query.

- **Steps**: Steps for the current day, as well as aggregates of steps in the current week/month/year could be added to the user query.

To reduce the risk of personally identifying a user based on the hiking trip recommendations, hiking trips in a large radius around the user is used. It should be noted that the risk increases for individuals living in a sparsely populated area.

6 Implementation

This chapter gives a description of how FysBot's design has been implemented, the tools used, and how FysBot is published to the Google Play Store. Section 6.1 explains the implementation of the backend server and how it is deployed to the Azure cloud. Section 6.2 explains how the database is set up and the injection of the exercise and hiking trip data. Section 6.3 explains the implementation for securing API endpoints and securing communication. Section 6.4 explains how the ChatGPT model is deployed in the Azure cloud. Section 6.5 explains the implementation of the features in the user interface. Finally, section 6.6 describes how the app is published to the Google Play Store.

6.1 Backend Server

The backend server was written in Kotlin using the Spring Boot framework with PostgreSQL as the database management system. The following sections describe the implementation of the backend system, how the virtual machine was deployed in Azure, how the ChatGPT model was deployed, data management and security.

6.1.1 Azure VM Deployment

A virtual machine hosted in Azure was used to implement the backend server. The entire deployment process was done via the Azure portal, which includes selecting the specifications for the VM, deployment region, storage type, virtual networks, reliability options and security. The specification is given in table 6.1. For the prototype, the amount of compute needed is very small; thus, one of the cheapest options was selected. The VM is deployed in Norway and uses Spot, which is a solution provided by Azure where the VM utilizes spare capacity in Azure's VM pool. This gives us a VM at a cheaper cost but at the risk of being evicted if the capacity in the VM pool is full. The latter is not desired in a production system but is not an issue for the prototype.

Table 6.1: Virtual Machine Specifications

Specification	**Value**
Operating system	Linux
Size	Standard DS1 V2
vCPUs	1
RAM	3.5GiB
Availability zone	None
Scale set	None
Security type	Standard
Disk	30GiB SSD
Spot eviction policy	Deallocate
Region	Norway east

6.2 Database Setup

The Postgres database was deployed in a docker container and was connected to the backend server through the Java Persistence API (JPA). Using the Object-Relational Mapping (ORM) system Hibernate as an abstraction layer to interface with the database, the creation of tables and updates to the tables were performed directly in the backend code. The tables in the database were created by annotating Kotlin classes as an "entity", for which Spring Boot scans through all the entities and constructs the tables when the server is run.

The exercise and hiking trip tables contain static data and require copying data from JSON and CSV sources. This is easiest to accomplish when initially building the docker image and involves injecting the SQL script shown in listing 6.1 into the docker image. The exercises are available in JSON format. To be able to copy this data to Postgres, it has to be converted to Newline Delimited JSON (NDJSON), which is done beforehand using jq[40]. Note that in the insert statement of "primaryMuscles", which is originally an array, only the first element is chosen as it is found that this array always contains a single element. By removing the array, future processing becomes easier. The hiking trips were manually written down in a CSV file and copied using this script.

Listing 6.1: SQL script for exercises and hiking trips

```
CREATE TABLE IF NOT EXISTS temp (data jsonb);

\COPY temp (data) FROM 'formatted\_file.nd.json' csv quote e'\
    ↪ x01' delimiter e'\x02';

CREATE TABLE IF NOT EXISTS exercises (
    name TEXT,
    force TEXT,
    level TEXT,
    mechanic TEXT,
    equipment TEXT,
    primaryMuscles TEXT,
    secondaryMuscles TEXT[],
    instructions TEXT,
    category TEXT,
    images TEXT[]
);

INSERT INTO exercises
SELECT
  data->>'name',
  data->>'force',
  data->>'level',
  data->>'mechanic',
  data->>'equipment',
  (ARRAY(SELECT jsonb\_array\_elements\_text(data->'
      ↪ primaryMuscles')))[1],
  ARRAY(SELECT jsonb\_array\_elements\_text(data->'
      ↪ secondaryMuscles')),
  data->>'instructions',
  data->>'category',
  ARRAY(SELECT jsonb\_array\_elements\_text(data->'images'))
FROM temp;

CREATE TABLE IF NOT EXISTS turer (
    name TEXT,
    lat DOUBLE PRECISION,
    lon DOUBLE PRECISION
);

\COPY turer(name, lat, lon) FROM 'turer.csv' DELIMITER ',' CSV
    ↪  HEADER;
```

6.3 Securing API Endpoints

6.3.1 Authentication

JSON Web Token (JWT) are used for authenticating API routes, and the backend server handles the generation of JWTs. This is done through a JWT class found in Java's IO package. When a user of the FysBot app creates a new profile or signs into an existing one, a new JWT is generated based on the user's username, known as the "subject" of the token. In addition, the token is annotated with an issue date and an expiration date. Using a public/private key pair generated for the server using OpenSSL[41], the last step of the token generation is signing it with the private key.

Table 6.2 shows every available API endpoint through the backend server. The endpoints that are *Authenticated* always require a JWT and the username. In Spring Boot, the implemented JWT filter class extends the native **OncePerRequestFilter** class, effectively intercepting all requests so that authentication can be performed before the request is passed to the API controller. When the server receives a request on an authenticated endpoint, the filter class checks that the token's signature is authentic using the server's public key and then extracts the subject from the token. The server can then authorize the request by verifying that the subject matches the username within the request.

Table 6.2: API Endpoints (TLS is enabled for all endpoints)

HTTP Method	Endpoint	Authenticated
GET	/	No
GET	/api/user/getUser	No
POST	/api/user/createUser	No
POST	/api/user/loginUser	No
POST	/api/user/stepsToday	Yes
POST	/api/user/setSettings	Yes
POST	/api/user/scheduleEvent	Yes
GET	/api/openai/chat	Yes
POST	/api/goal/getGoal	Yes
POST	/api/goal/setGoal	Yes

Since the request has to be read by both the JWT filter class (to extract the username) and by the destination endpoint (if the token validation process succeeds), caching the request is required as illustrated in figure 6.1. Typically, the request body can only be read once, as caching the body for each request can be expensive and introduce scalability issues. However, for the moment, this is the only identified solution to the problem. To read the request body multiple times, Java's **HttpServletRequest** and **ServletInputStream** classes are extended, and the input stream reader methods are overridden to cache each request.

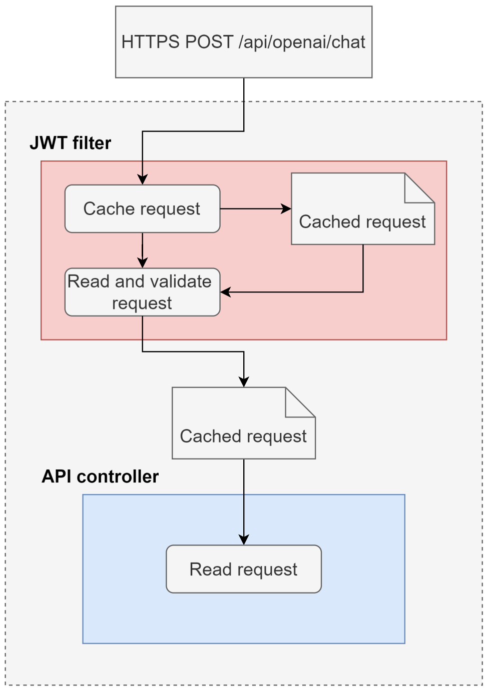


Figure 6.1: Caching a request to enable multiple reads.

6.3.2 NGINX and TLS

As explained in the security chapter, the backend server has enabled Transport Layer Security by using Certbot and NGINX. For this, the configuration shown in listing 6.2 is added to the NGINX configuration file. The *location* block specifies that incoming requests shall be proxied to the local Spring Boot server running on port 8080 using HTTP (thereby, TLS is terminated by the NGINX proxy).

Listing 6.2: NGINX TLS configuration

```
listen 443 ssl default_server;
listen [::]:443 ssl default_server;
ssl_certificate /etc/letsencrypt/live/fysbot.norwayeast.
    ↪ cloudapp.azure.com/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/fysbot.norwayeast
    ↪ .cloudapp.azure.com/privkey.pem;
server_name fysbot.norwayeast.cloudapp.azure.com;
location / {
    proxy_pass http://127.0.0.1:8080;
}
```

The certificate is created using Certbot with the command shown in listing 6.3.

Listing 6.3: Using Certbot to generate TLS certificate

```
$ sudo certbot --nginx -d fysbot.norwayeast.cloudapp.azure.
    ↪ com -d www.fysbot.norwayeast.cloudapp.azure.com
```

6.4 Azure ChatGPT Deployment

Like the Azure VM, the ChatGPT model was deployed by creating an OpenAI instance through the Azure portal. The model is deployed in Sweden within the same VNet used by the VM. The ChatGPT instance was configured to be accessible only from within the VNet, meaning that only the VM will have access to it. When created, the OpenAI API can be accessed through the URL of the ChatGPT deployment with the API keys generated by Azure.

6.5 User Interface

The user interface was written in TypeScript with the React Native (RN) framework, using Metro for building and testing the application while developing. The following sections discuss specific details of the implementation of the UI.

6.5.1 Contexts

The following contexts (introduced in section 2.6.2) are implemented in FysBot:

- **ChatbotContext**: Holds state for messages, chatbot name and avatar. This context will read messages, name and avatar from the user's local storage and inject them into the context state on launch of FysBot. It will also inject a welcome message as the first message in the message state. Whenever one of the states updates, the new values are also saved in the local storage.

- **BadgeContext**: Holds the state for earned badges. This context runs the badge synchronization process by repeatedly checking the user's step count, adding new badges to the state, and storing them in the user's local storage when earned. On launch of FysBot, the context will load the earned badges from the user's local storage.

- **GoalContext**: Holds the state of the user's current daily step goal. Set goals are stored in local storage and retrieved on launch.

- **StepContext**: Holds the state of the user's current steps and includes methods for retrieving steps. It also holds the state of whether the user has granted FysBot access to their Google Fit data. This context will continually request the current step count and update the step context if needed.

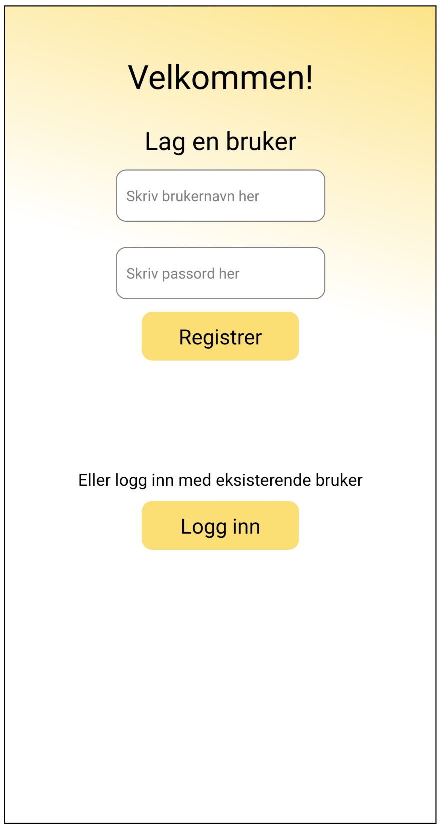
- **KeyboardContext**: Holds the state of whether the keyboard is currently in view.

- **ScheduleContext**: Holds the state of the scheduled events set by the user. Updates to the schedule state are stored in the user's local storage and loaded on launch.

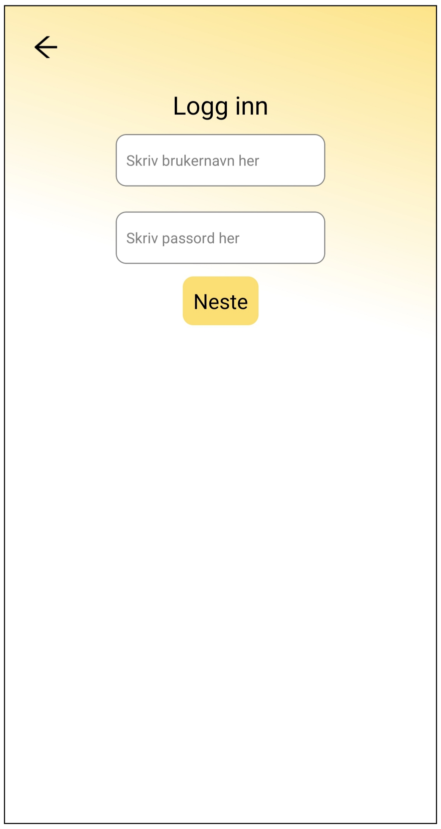
- **UserContext**: Holds the state of the user's username, avatar and logged in status, as well as the state for whether it is the first time the user has launched the app. If the user has previously created a profile, the username, avatar and first-time user states are retrieved from local storage.

6.5.2 Login

The login screen is the first content displayed to the user on the initial launch of FysBot. This screen is shown depending on whether the user owns a valid authentication token. If the user does not have a valid token, the user is able to either create a new account, as seen in figure 6.2a, or log in to an existing account, as seen in figure 6.2b, in order to gain a token. Since the token will be used most frequently for automatic authentication, the create user text input fields are shown first, while the log-in option requires an extra button press.



(a) Create new user.



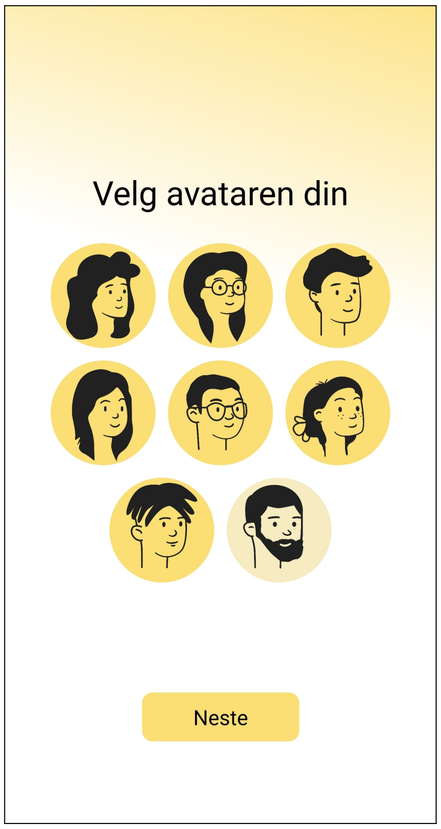
(b) Login existing user.

Figure 6.2: Create user and login tabs.

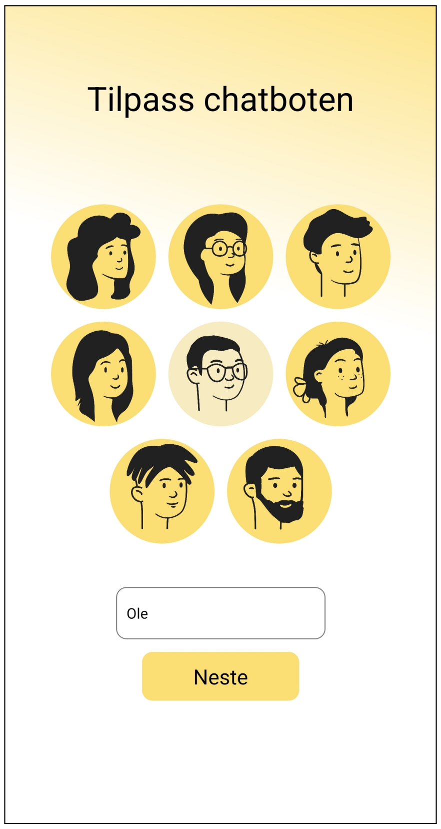
6.5.3 Create New User

When the user creates a new user, the server checks whether a user already exists with the same username; if it does, the user receives a descriptive error message. If not, the user profile is created. The creation process involves sending a request to the server to store the username, creation date, and hash + salt of the password in the backend database. In return, the user receives an authentication token, which is stored in the local storage of the user's phone. This token is then used for future authentication requests.

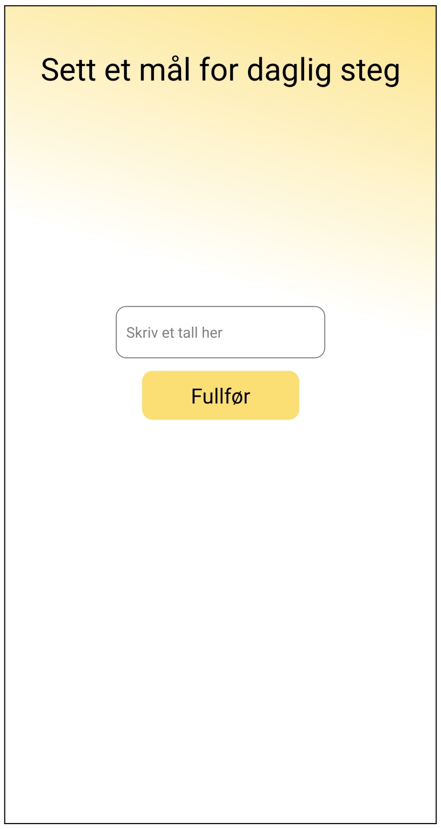
The user is then taken to the next part of the user creation process, which involves setting up their avatar, customizing the chatbot with a name and avatar and finally setting a daily step goal as shown in figures 6.3a, 6.3b and 6.3c, respectively. After a valid step goal is set, the user is taken to the main content of FysBot.



(a) Seleting the user's avatar.



(b) Customizing chatbot.



(c) Setting the daily step goal.

Figure 6.3: The next three steps required to complete the user profile creation after setting a username and password.

6.5.4 Log in Existing User

Logging in as an existing user requires a valid username and password combination. The server first verifies that a user with the given username exists, then checks if the hashed + salted password matches the credential stored in the database. If not, the user receives a description message of what went wrong. If the verification succeeds, the user is given an authentication token, which is stored in the local storage of the user's phone.

6.5.5 Tab Navigation Bar

An essential part of an application with multiple tabs is enabling simple navigation. In FysBot, a navigation bar is placed at the bottom of the screen, which follows the user in every tab they visit. Each option in the navigation bar has an associated image and descriptive text to recognize the tab's functionality easily. A state variable contains information about the currently selected tab, and by pressing one of the tab navigation options, the state value updates and the new content is rendered. This is achieved by pointing a specific state value to a particular functional component (see functional components in section 2.6.2). To easily recognize which tab is currently in view when looking at the navigation bar, the same state value is used to highlight the currently selected tab. The initial state value is set to the chatbot messaging tab, as this is the core part of the application and will be the first thing the user sees after logging in.

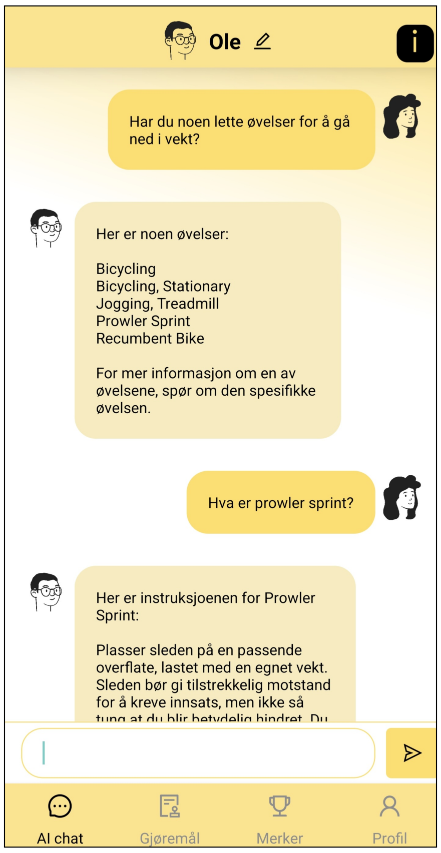
An issue with the navigation bar is that it will sit on top of the keyboard whenever the keyboard comes into view, cluttering the screen. Because of this, the navigation bar is conditionally rendered and only shown if the keyboard is not active. This is achieved by consuming the keyboard state from the *KeyboardContext*.

6.5.6 Chatbot Messaging

The chatbot messaging component is the core part of FysBot. It displays all messages between the user and the chatbot, as well as a text input field for writing a new message that can be sent by pressing the arrow button. Pressing the text input field will bring the keyboard into view, and the navigation bar will be hidden. When the user sends a new message, an API request is sent to the server to get the chatbot response. Based on the topic of the message, additional information like steps or location could be required. The client will then await the server's response and conditionally send the required information if needed. By consuming the *ChatbotContext*, the new user message and its response are added to the message state. Since this state is used in the JSX returned by the messaging component, any new additions to the message state cause the component to re-render, instantly showing the new messages.

The *ChatbotContext* is also used to display the name and avatar of the chatbot at the top of the tab, with the option of changing the name of the chatbot. In combination with the *UserContext*, each message is accompanied by the avatar of the user and chatbot to enhance the user experience.

Figure 6.4 is an example of asking for exercise recommendations where the Exercise Recommender Persona first lists a set of exercises based on the user query. It is then able to retrieve additional information if needed. Each exercise has an instruction as well as two images to illustrate the exercise.



Ole

Har du noen lette øvelser for å gå ned i vekt?

Her er noen øvelser:

Bicycling
Bicycling, Stationary
Jogging, Treadmill
Prowler Sprint
Recumbent Bike

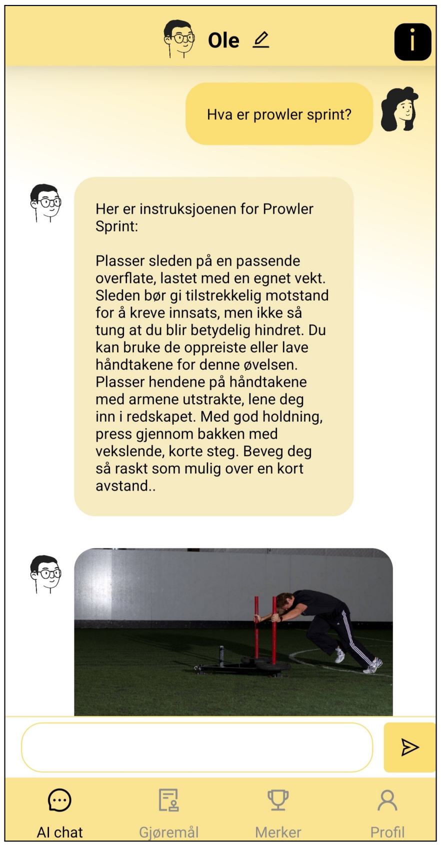
For mer informasjon om en av øvelsene, spør om den spesifikke øvelsen.

Hva er prowler sprint?

Her er instruksjonen for Prowler Sprint:

Plasser sleden på en passende overflate, lastet med en egnet vekt. Sleden bør gi tilstrekkelig motstand for å kreve innsats, men ikke så tung at du blir betydelig hindret. Du

(a) Getting exercise suggestions



Ole

Hva er prowler sprint?

Her er instruksjonen for Prowler Sprint:

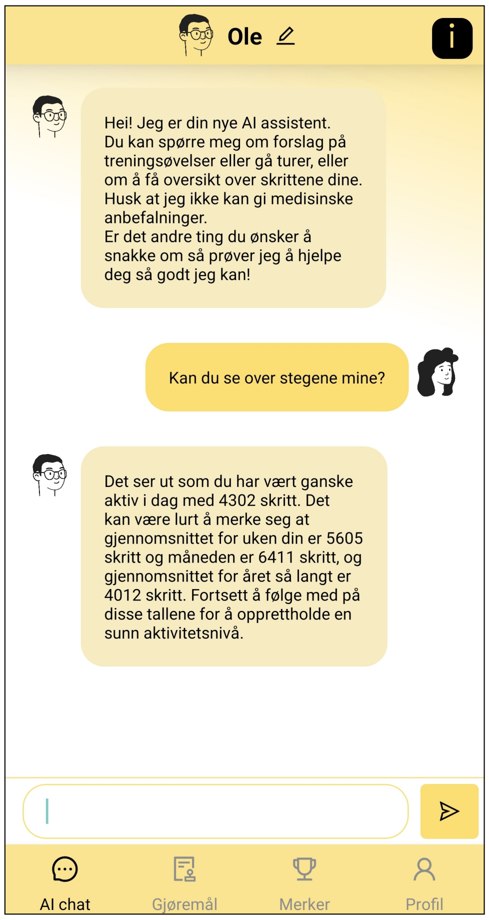
Plasser sleden på en passende overflate, lastet med en egnet vekt. Sleden bør gi tilstrekkelig motstand for å kreve innsats, men ikke så tung at du blir betydelig hindret. Du kan bruke de oppreiste eller lave håndtakene for denne øvelsen. Plasser hendene på håndtakene med armene utstrakte, lene deg inn i redskapet. Med god holdning, press gjennom bakken med vekslende, korte steg. Beveg deg så raskt som mulig over en kort avstand..

(b) Getting instructions and images

Figure 6.4: Requesting exercise recommendations. ERP found the following parameters in the first query: level=beginner, category=cardio.

Figure 6.5 shows the user asking for progress review within the app. The

Progress Review Persona will request the user's steps and craft a response based on the steps.


Ole

Hei! Jeg er din nye AI assistent.
Du kan spørre meg om forslag på treningsøvelser eller gå turer, eller om å få oversikt over skrittene dine.
Husk at jeg ikke kan gi medisinske anbefalninger.
Er det andre ting du ønsker å snakke om så prøver jeg å hjelpe deg så godt jeg kan!

Kan du se over stegene mine?

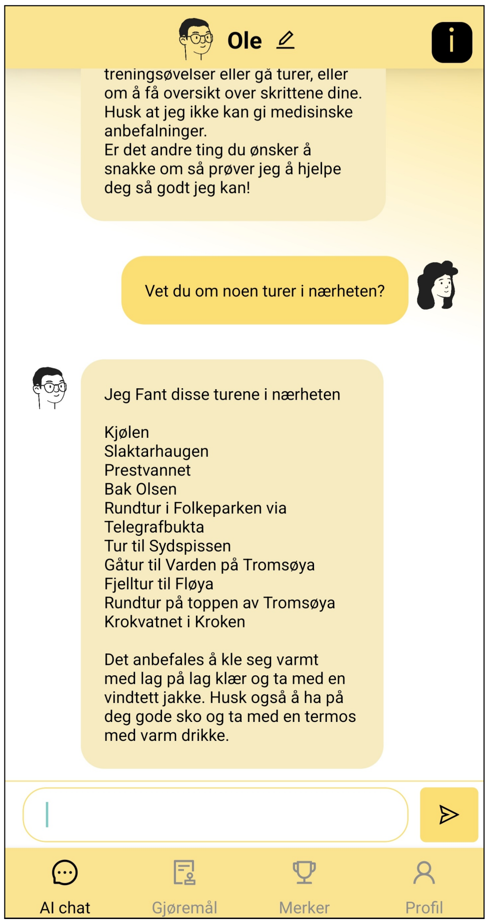
Det ser ut som du har vært ganske aktiv i dag med 4302 skritt. Det kan være lurt å merke seg at gjennomsnittet for uken din er 5605 skritt og måneden er 6411 skritt, og gjennomsnittet for året så langt er 4012 skritt. Fortsett å følge med på disse tallene for å opprettholde en sunn aktivitetsnivå.

Figure 6.5: Requesting progress review.

Figure 6.6 shows the user asking for hiking trips. The Location Activity Persona will request the user's location, fetch nearby hiking trips and the current weather conditions at that location, and use it in its recommendation.

6.5.7 Daily Schedule

The daily schedule feature allows the user to create events for the current day and have the option to get notifications on specific events. As shown in figure 6.7b, the user is able to add new events by pressing the plus icon, which brings up the event customization modal shown in figure 6.7a. Here, the user is able to set the hour and minute of the event and optionally press the notification symbol to receive a notification before the event takes place.



treningsøvelser eller gå turer, eller om å få oversikt over skrittene dine. Husk at jeg ikke kan gi medisinske anbefalninger.
Er det andre ting du ønsker å snakke om så prøver jeg å hjelpe deg så godt jeg kan!

Vet du om noen turer i nærheten?

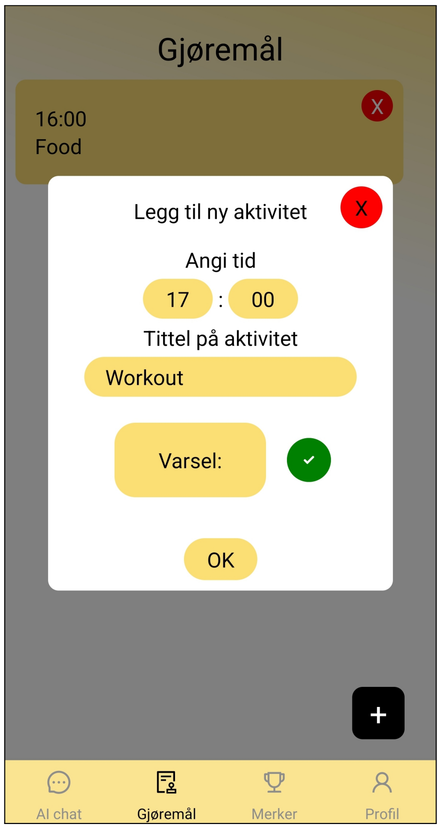
Jeg Fant disse turene i nærheten

Kjølen
Slaktarhaugen
Prestvannet
Bak Olsen
Rundtur i Folkeparken via Telegrafbukta
Tur til Sydspissen
Gåtur til Varden på Tromsøya
Fjelltur til Fløya
Rundtur på toppen av Tromsøya
Krokvatnet i Kroken

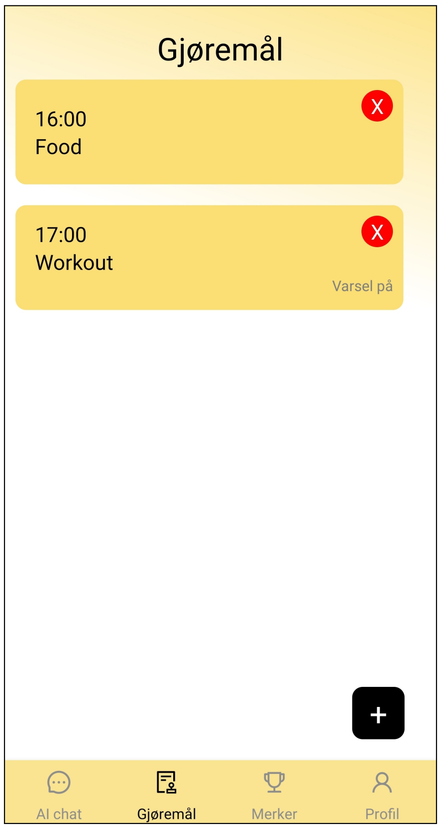
Det anbefales å kle seg varmt med lag på lag klær og ta med en vindtett jakke. Husk også å ha på deg gode sko og ta med en termos med varm drikke.

Figure 6.6: Requesting hiking trips.

To display the existing events in the schedule, the component consumes the *ScheduleContext* and retrieves the schedule state. Creating a new scheduled event will add the event to the state, which re-renders the list of scheduled items shown in figure 6.7b. Each item is sorted by time before rendering. Adding an item with notification enabled will issue an API request with username, time and event title to the server.



(a) Setting a new schedule entry.



(b) All events in schedule

Figure 6.7: Schedule feature in app.

6.5.8 Badges

The badges, shown in figure 6.8, are based on the user's daily step progress and can be earned by walking a certain amount of steps. It consumes the *BadgeContext* and aggregates the number of times each badge has been earned. This number is then displayed in the corner of each badge. Pressing a badge will display the requirements for how the user can earn it.

6.5.9 User Profile

The profile tab allows the user to set a new daily step goal and review their step count. The step count is fetched from the *StepContext* and displayed both in a circle that shows the step count for the current day and in a bar chart for the current week, as seen in figure 6.9. The circle can also display the step progress directly against the current goal, shown in the same figure.

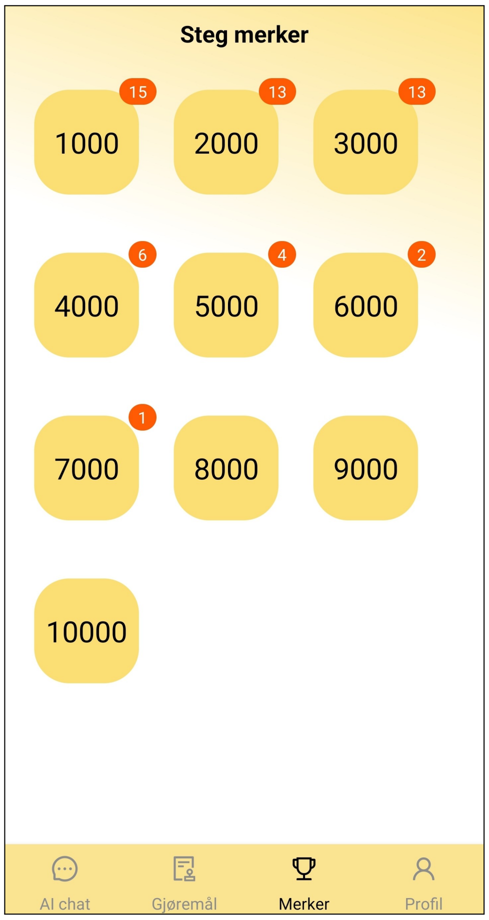


Figure 6.8: Step badges.

6.5.10 Firebase Push Notifications

Push notifications are provided through the Google service Firebase Messaging[42]. The user may get notifications on goals reached or for upcoming scheduled events. To enable Firebase Messaging, the FysBot application is registered in the Firebase console through their web portal. From the web portal, a JSON file is downloaded containing the API key for the messaging service along with other metadata such as project identifiers. This JSON file must be placed under */app* within the */android* root folder to enable Firebase Messaging on the client. The Firebase React Native API can then be used to handle incoming notifications and fetch the Firebase token associated with the user's mobile device. This token is sent to the server when the user logs in or creates a new profile in the FysBot app.

In the backend server, a different JSON file is required. This file is downloaded

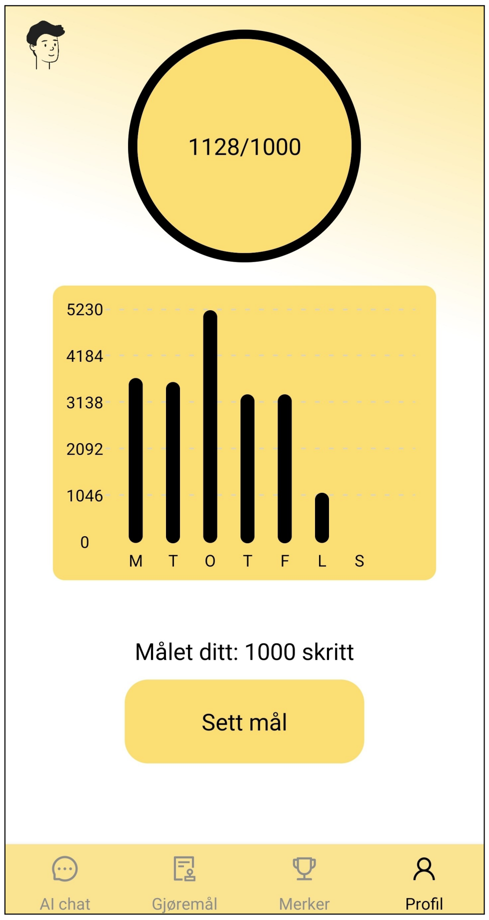


Figure 6.9: The user profile which displays step counts and the step goal, which can be updated by pressing the "Set goal" button.

from the Google Cloud Console and contains metadata such as the ID of the client sending the push notifications. It is used to retrieve the server's Firebase token, which is necessary for authenticating the server against the Firebase API. The backend server can then send notifications to registered users by including the user's Firebase token in the API request.

6.6 Publishing to Google Play Store

To publish the FysBot Android application, the React Native Command-Line Interface (CLI) is used to build a signed Android App Bundle (AAB). The signing key is generated using the Keytool CLI provided by Java. Navigating to "internal test" in the Google Play console[43], a new version of FysBot is published by uploading the signed AAB. Google will then perform a verification process to check the validity of the signature and make sure that the application does not request unauthorized permissions. The permissions are, for example, reading Google Fit steps, which is allowed in the internal testing stage. If the verification process is successful, the application can then be published. For users to be able to download FysBot, their Gmail accounts must be added to a list of test users within the Google Play Console.

An important thing to note is that Google will have its own certificate for signing the published build. This certificate has to be used to authenticate FysBot against other Google and third-party APIs like Google Fit. To fix this, the SHA-1 fingerprint of the publishing certificate is added to the credentials of the Google account used to publish the application.

7 Evaluation and Result

In this chapter, an evaluation, and the results, of the usability study is provided in section 7.1. This includes the time used for the different tasks as well as the most notable comments from the participants. In the following section, 7.2, an evaluation of the system performance and cost is given.

7.1 Usability Study

FysBot was evaluated using the think-aloud method as explained in section 3.3. This section will focus on how well the chatbot and the application performed as a whole and present the participant's feedback, which was given during the completion of tasks. The tasks are presented again in table 7.1 to make it easier to follow the evaluation process. The time participants used to complete specific tasks is also shown and was determined using the voice recording.

Table 7.1: Tasks evaluated using the think aloud method

NO	Task
1	Download and install the FysBot app
2	Sign in to FysBot - create a username and password
3	Give the chatbot a name

4	Set step goal
5	Start chatting with the chatbot
6	Find and read information about FysBot
7	Ask the chatbot to recommend exercises
8	Change step goal
9	Make a physical activity plan for the day
10	Ask the chatbot to give feedback on steps
11	Ask/check for overview of physical activity
12	Ask the chatbot a physical activity related question
13	Ask the chatbot a general question
14	Check the overview of steps
15	Check how many badges you have received
16	Ask the chatbot to suggest a hike

7.1.1 The Participants

Participants were selected from among friends and colleagues. Five participants aged 20-60 were included in this test. All participants are considered highly proficient with technology. Participants were sent a link via their Gmail to download the app. Those who did not own Android devices could borrow one for the sake of the test.

7.1.2 Downloading and Installing FysBot

Since FysBot was released on an internal test stage, participants might not have necessarily found it in their Google Play store. In most cases, the participants had to be given the download link directly. This caused some complications with the time measurements, so they were not calculated for this task. In any case, none of the participants experienced any issues downloading and installing the app after receiving the link.

7.1.3 Logging into FysBot

The first four tasks involved downloading and installing the FysBot app, creating a user account, customizing the chatbot, and setting a step goal.

After opening FysBot for the first time, the participants are presented with a Google account login pop-up to enable Google Fit data collection. No problems were encountered in this step, although one participant stated that it would be nice to know why this login was necessary. After logging into their Google account, the next step was to create a new user profile. Here, the user first sets their username, selects their avatar, and customizes the chatbot with its own name and avatar.

Two participants did not find the chatbot customization intuitive, as they did not realize why they had to select an avatar a second time (not knowing it was for the chatbot). This can be seen by looking at the times for participants 2 and 3 to complete Task 3 in figure 7.1. It was also mentioned that the selected avatar could be made more clear, as people with vision problems might have difficulty seeing it. The last step of the 'create user' process was to set a step goal, where one of the participants noted that it would be nice to know what a good step goal for them would be. For tasks 2 and 4 in figure 7.1, some participants spent more time thinking about their username and step goal.

7.1.4 Chatbot

There are seven tasks that revolve around interactions with the chatbot (5, 7, 10, 11, 12, 13 and 16 in table 7.1). The time to complete these tasks is not included, as participants had no issues navigating to the chatbot tab and asking the chatbot the appropriate questions to get the information they wanted. However, it is also in these tasks that the most critical user experience issues were discovered and are discussed further in section 8.1.2.

Hiking Trip Suggestions

Overall, users were satisfied with the chatbot's responses to hiking trip suggestions, step reviews and exercise recommendations. However, they were not

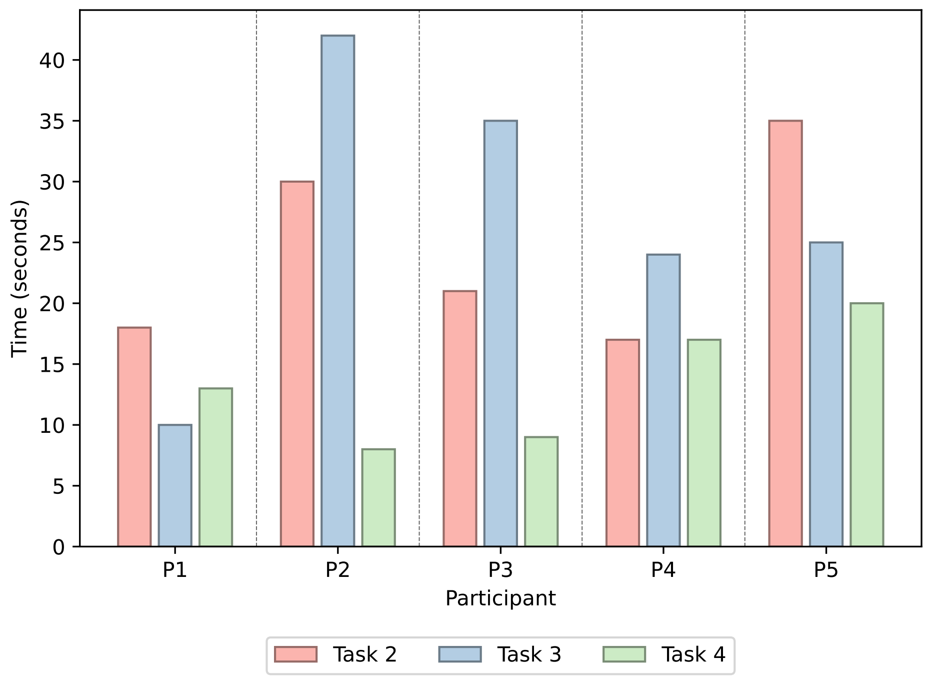


Figure 7.1: Time taken to complete task 2: Sign in to FysBot, task 3: Give the chatbot a name, and 4: Set a step goal.

entirely satisfied with the responses to the follow-up questions. Two participants tried to ask for hiking trips near a specified location, unaware that the chatbot only uses the user's current position. One of the participants tried to ask for more information about a particular hiking trip using the trip name. The chatbot did not understand the query since it's not programmed to give additional details on specific hiking trips.

Exercise Recommendations

For the exercise recommendations, participants were satisfied with getting a list of exercises based on their exercise goals. They also tried to ask the chatbot for additional information on particular exercises and were pleased to receive instructions and images on how to perform them. However, one participant suggested that the exercise names should be in Norwegian to make it easier to identify the purpose of the exercise. There was also a suggestion to press one of the recommended exercises to get instructions and images instead of having to type the exercise name.

Steps Review

All participants successfully got a step review response on their first try, although the response content was not satisfactory due to a bug in the code which recorded the wrong step count. This is a critical error which must be fixed. One participant did not have Google Fit installed but still tried to complete the step review task, although the chatbot responded by stating that it did not have access to the steps. Another participant did have Google Fit installed, but FysBot was unable to collect the step data. Additionally, one participant felt that the chatbot's response lacked sufficient focus on the number of steps, desiring more concrete statistics.

7.1.5 Step Goal and Step Overview

Task 8 involved going to the user's profile tab and changing the step goal the user initially set when creating their profile. In figure 7.2, it can be seen that Participant 5, in particular, took a long time to complete this task. Like Participant 4, they tried asking the chatbot to change the step goal before realizing it had to be done manually in the profile tab. Participant 2 was initially prevented from setting the step goal because they mistyped it and tried to set a goal over 100 000 steps, which the app did not allow.

All participants successfully found the step overview of task 14 in the user profile tab. As seen in figure 7.2, there were no issues related to this task.

7.1.6 Finding Information About FysBot

Task 6 involved finding and reading information about FysBot, for which most participants had no problems, as seen in figure 7.3. However, Participant 4 tried to ask the chatbot first before identifying the information tab in the top right corner of the application. The information button and the information tab within the app is shown in figure 7.4. They then suggested that the information tab should have been presented right after logging into FysBot.

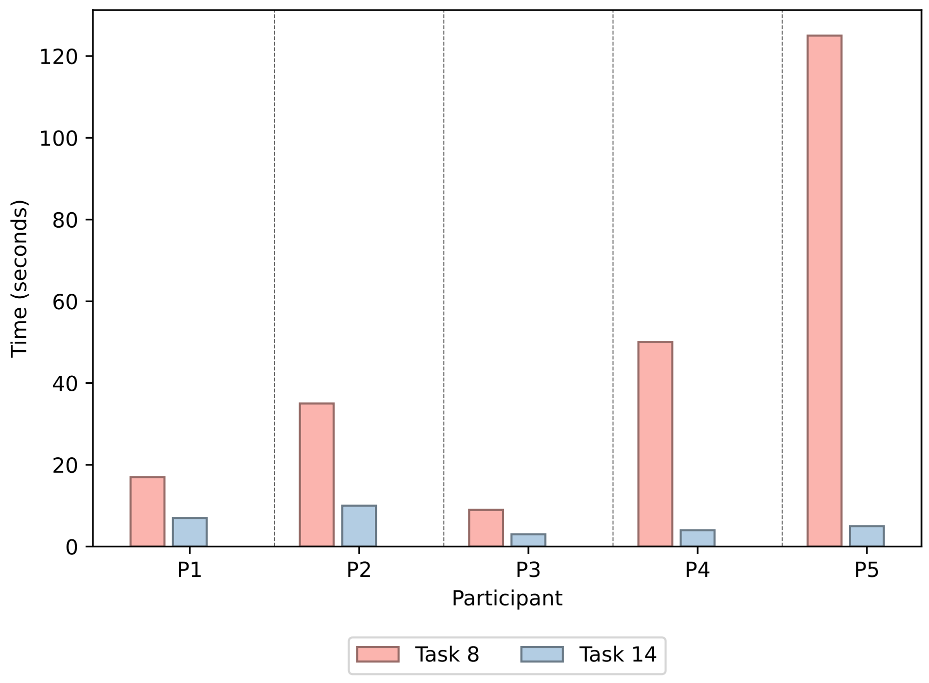
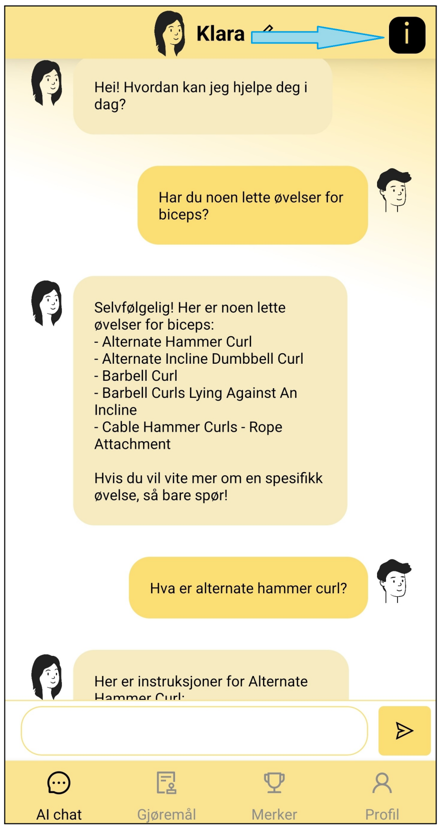


Figure 7.2: Time taken to complete task 8: Change step goal and task 14: Check the overview of steps.



Hei! Hvordan kan jeg hjelpe deg i dag?

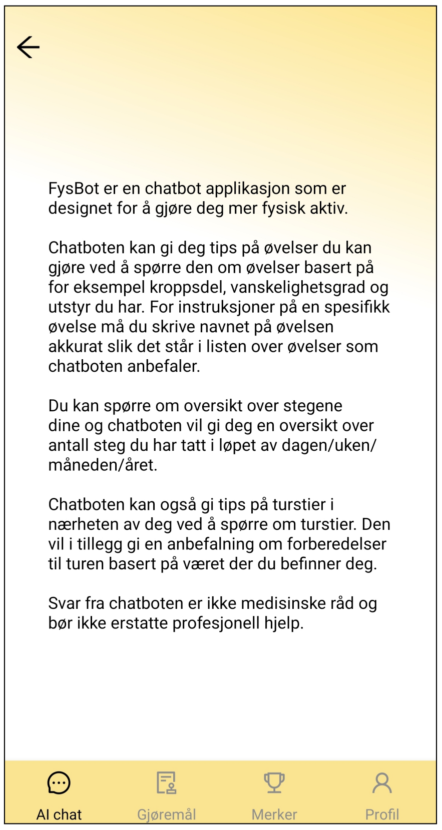
Har du noen lette øvelser for biceps?

Selvfølgelig! Her er noen lette øvelser for biceps:
- Alternate Hammer Curl
- Alternate Incline Dumbbell Curl
- Barbell Curl
- Barbell Curls Lying Against An Incline
- Cable Hammer Curls - Rope Attachment

Hvis du vil vite mer om en spesifikk øvelse, så bare spør!

Hva er alternate hammer curl?

Her er instruksjoner for Alternate Hammer Curl:



FysBot er en chatbot applikasjon som er designet for å gjøre deg mer fysisk aktiv.

Chatboten kan gi deg tips på øvelser du kan gjøre ved å spørre den om øvelser basert på for eksempel kroppsdel, vanskelighetsgrad og utstyr du har. For instruksjoner på en spesifikk øvelse må du skrive navnet på øvelsen akkurat slik det står i listen over øvelser som chatboten anbefaler.

Du kan spørre om oversikt over stegene dine og chatboten vil gi deg en oversikt over antall steg du har tatt i løpet av dagen/uken/måneden/året.

Chatboten kan også gi tips på turstier i nærheten av deg ved å spørre om turstier. Den vil i tillegg gi en anbefalning om forberedelser til turen basert på været der du befinner deg.

Svar fra chatboten er ikke medisinske råd og bør ikke erstatte profesjonell hjelp.

(a) The information button (indicated by the arrow).

(b) The information displayed after pressing the information button.

Figure 7.4: Schedule feature in app.

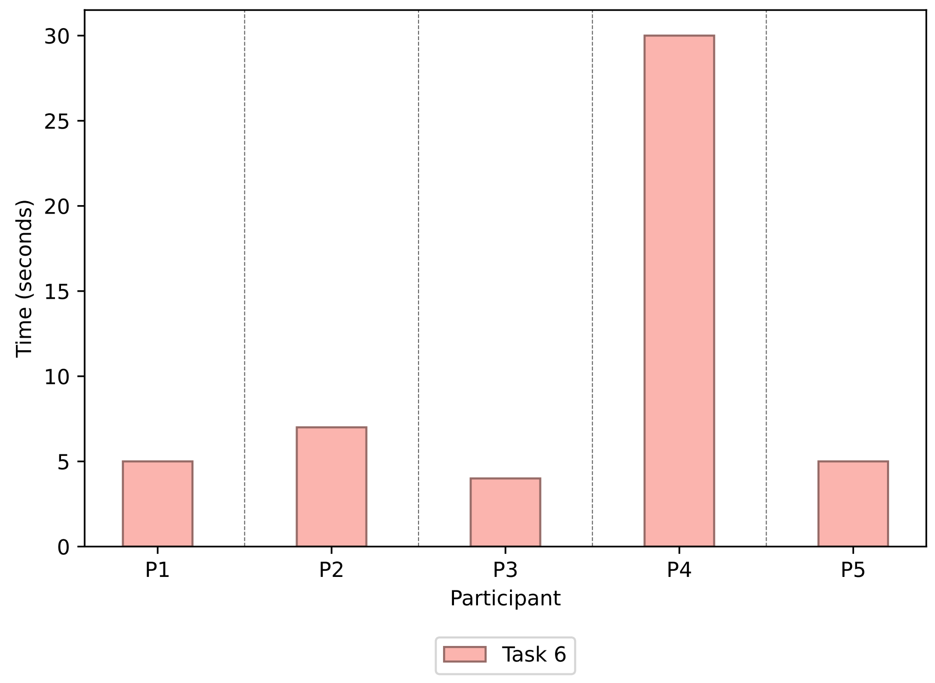


Figure 7.3: Time taken to complete task 6: Find and read information about FysBot.

7.1.7 Badges and Daily Schedule

In task 9, participants were asked to set a physical activity plan for the day in the schedule tab. Three of the participants assumed that they were to set up an exercise routine, creating schedule entries with exercise names and the duration of that exercise. Because of this, they were unsure of the purpose of the notification option.

The time to complete task 9 is shown in figure 7.5 and was measured from when the user read or was given the task until the first scheduled event was created. Participant 1 spent a little time looking for the correct tab to create the events but had no problems after that. Participant 2 initially did not set the hour of the event as it was assumed the scheduled events were exercises in a workout plan (and thus did not need the hour component). Participant 3 initially asked the chatbot to create the plan but had no problems after finding the correct tab. Participant 5 used the longest time and thought the schedule was for creating a workout routine for the day. In addition, this participant did not set the hour component for the same reason as Participant 2 and created a title that was too long. Some participants preferred longer titles in the event as this was limited to 20 characters.

Task 15, checking how many badges the user has received in the badges tab, was done with no problems, as evident in figure 7.5. Four of the five participants did, however, have some comments about the design of the badge tab. They found it unintuitive to know whether or not a badge had been earned because of the color options used. Two of them also thought they should already have received some of the badges based on their current step count (in the current version, badge progress is only recorded after installation of FysBot).

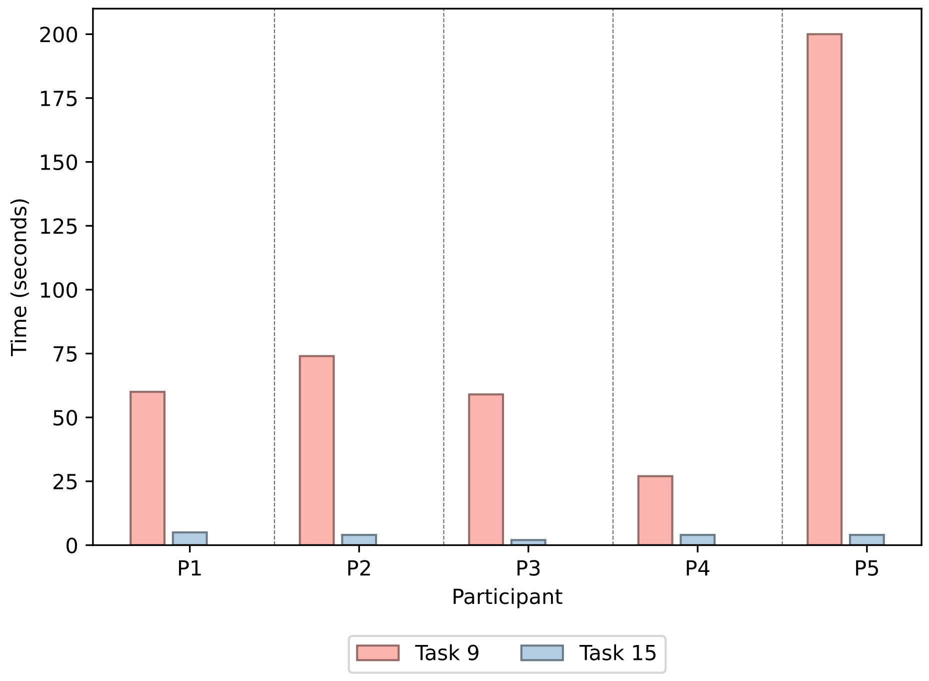


Figure 7.5: Time taken to complete task 9: Make a physical activity plan for the day and task 15: Check how many badges you have have received.

7.1.8 Overall Thoughts

Each test ended with asking the participants to give general feedback about FysBot, and they were encouraged to comment on the design, too. The following lists some of their suggestions for improvement that have not been stated previously:

- **Edit font size**. It should be possible to increase the size of the font.
- **Edit avatars**. It should be possible to change the user's and chatbot's avatar.

- **Chatbot name editor**. The chatbot name edit feature was not intuitive.
- **Emphasise chatbot tab**. Since the chatbot is the app's main feature, it would be nice if the tab option in the tab navigation bar was somehow more dominant than the other options.
- **Information**. When using the app for the first time, some information about the usage of the app and the abilities of the chatbot should be presented immediately.
- **Swiping between tabs**. It would be nice to have the ability to swipe between tabs instead of using the tab navigation bar.
- **Wearable sensors**. The total step count could take into account the steps registered on wearable devices.
- **Hiking trip difficulty**. The difficulty level of the hiking trip could be stated to get more appropriate suggestions.
- **Hiking trip location**. Could have a map to show the location of the hiking trip.

7.2 System Performance And Cost

The usability study encountered no errors that led to system failure. The study took approximately 6 hours, and the backend server ran continually without problems. The total cost of running the backend server and its associated resources on the day of the usability study amounted to 7.56 NOK. Considering the different time frames when the server was not in use, the cost of renting the VM is approximately 3.22 NOK per day. Therefore, the price of the added load was 4.34 NOK. For the ChatGPT deployment, a total of 2.39 NOK was spent on 387 messages and completions.

8 Discussion

The following sections discuss certain topics in design, ethics, implementation, and user testing to reflect on the parts that worked well and those that still require improvement.

8.1 Software Development Process

8.1.1 Solving a Problem

FysBot aims to motivate people to become more physically active. Each design and implementation decision strives to solve the challenge of motivating and guiding individuals towards improved physical activity. Therefore, the features of the system are influenced by this objective. By contemplating the problem that FysBot seeks to address, each decision is made to align with the user's needs.

In section 3.2, two fictive users and several user stories were presented. This is used to concretize the problem that needs to be addressed and from which a set of features can be outlined. The features are then realized in the prototype of FysBot. The prototype includes all the features that were scheduled for this project.

To ensure that there is value in the design decisions and that the implemented system works as intended, it is ideal that new prototypes are pushed out and tested constantly. Unfortunately, only one prototype of FysBot was created during this project. To some extent, this was because FysBot has a single developer, but even so, delivering multiple prototypes could have still been possible. This could have been done by accepting that parts of the application need not be finished or even functional for it to be evaluated. One concern is finding participants to test the prototypes, as having multiple usability tests with multiple users is time-consuming. For this project, if multiple prototypes were created, having the PhD student test each prototype would have been a great option.

8.1.2 Usability test

Testing the usability of a product is difficult to do in a vacuum, and the success of the product critically relies on testing it with real users. The think-aloud method was an excellent tool for discovering how users interact with the system. Many of these interactions were unexpected, where the user had a different perception of using a feature than originally intended.

System Performance and Cost

Section 7.2 provided an evaluation of the performance and cost of the system as a whole. With such a small group of users over a short period, the reliability of these results most likely does not transfer to a scenario with more users. However, one could use the cost from the usability study to project costs for many users.

Findings

Most users thought the daily schedule feature was a method for setting up a workout plan and inputting the exercise name and the duration of the exercise. This could partly be due to how the task was formulated: "Make a physical activity plan for the day". The title of the schedule tab could be updated to avoid future misunderstandings. In addition, a date should be displayed, possibly under the title. There was also some dissatisfaction about the legal length of the title on a scheduled event. This is set to a maximum of 20 characters and should be increased. Some users expressed a desire to set a weekly plan instead, which is a good idea and should be implemented in the future. Perhaps, users can have the option to choose the kind of plan they would like to set. There was also a desire to update scheduled events after creation. This can easily be fixed by implementing an *on press* listener for each event, enabling an editorial mode. Lastly, in the daily schedule feature, some users wondered what enabling notification on an event did for them (most likely due to thinking the daily schedule was a workout routine planner). The text beside the notification button could be more descriptive to prevent future confusion. Another option could be to enable an informational pop-up text box to appear and explain it when the user first enters the schedule tab.

For the tasks relating to the chatbot, the participants were able to ask for exercise recommendations, hiking trips and step progress reviews easily. However, they received unexpected responses to some of the follow-up questions. For the hiking trip recommendation task, users successfully got recommendations on the first try, but some users wanted more information about specific trips, like route information. The first issue to address with this is to inform the user that this type of information is not available at the moment, possibly through a response from the chatbot. Attempts have been made to make this sort of functionality available by using an open database provided by Kartverket that contains detailed information on hiking trips (route information, difficulty, etc.). However, due to several problems in using the database, implementing this functionality proved harder than anticipated. Kartverket was contacted to resolve the issues, but they could not help. The second issue is that the chatbot mistakenly recognized the request for additional information as a request for general hiking trip recommendations, for which it simply repeated the initial recommendation. This sort of problem also occurs with the Progress Review Persona as any follow-up questions, like "how many steps should I be walking?" caused the PRP to repeat the user's step progress.

There were problems experienced with the Exercise Recommender Persona as well. If it is found that a query should be processed by the ERP, it will try to extract information on which types of exercises to recommend and then statically list them. If no specific information is found in the query, it will ask for more information. In one particular case, Participant 2 asked: "Create a plan that makes me reach my goals", for which the chatbot just listed a set of random exercises, which was not really what the participant expected to get. The ERP has to be more clever in its reasoning, always asking the user what their goals are to prompt for more information before making its decision. An additional mechanic that can be implemented is not to list the exercises statically, but instead tell the ERP "Here are some exercises that match the user's request. Use them if needed to answer the question.". In this case, knowing the user's goals, the ERP could answer the question by even creating a complete workout plan. A proposed solution to mend these issues with the chatbot personas, as well as other design limitations, is discussed in section 8.8.

Limitations

For the usability test, all participants were quite proficient with technology and had previously used chatbots like ChatGPT. This is a limitation of the study, which could have included a more diverse group.

Exactrating time measurements from the audio recordings of the think-aloud test were challenging. The participants could have been focused on talking about their actions and thoughts and asking questions to get clarification while performing the tasks. Therefore, it can be assumed that the time users spent on certain tasks could be lower than what was measured in the study. There was an attempt, however, to cut out the time used to ask questions and provide clarification. Measuring the correct time can also be challenging since most users probably navigated between the tabs while searching for a solution for a particular task. They could then observe parts of the application associated with other tasks. Therefore, they knew exactly what to do when they started on the later tasks.

The study had no strict rules about the order in which participants completed the tasks, which also made time measurements difficult. It is thought that being more strict about following the exact order could give more accurate results. The time measurements were also taken by listening to the recordings of each participant, but ideally, this should have been done while observing the participant's actions within the application. The trade-off is that this could be disturbing to the participant if it is not done in a subtle manner. Optionally, accurate timings after the fact could be achieved using analytics tools, like the ones described in section 8.6.

8.2 Avoiding Misuse of FysBot

To use the app, a user profile has to be created using a username and password, and a token filter helps authenticate the API endpoints such that a user cannot retrieve another user's data. In a production system, additional authentication mechanisms should be implemented to secure the system further. This could be, for example, in the form of a verification message that is sent by email or SMS. One of the reasons for doing this is to be more guarded against exploit attacks. Hackers will often use bots that try to perform various exploits on open ports, which was experienced in the FysBot server during testing. Using web crawlers, they are also able to find the API endpoints of the server, which they also try in their exploits. Some bots are smarter than others, and it has been experienced that they are able to create user profiles by randomly filling in parameter values. This can, to a large extent, be avoided by using multi-factor authentication. A short-term solution could be to add a long static string at the end of the username when the user creates a profile. This string is also known by the server and removed before the user profile is stored in the database. If the username does not contain this string, the server rejects the request.

8.3 Complexities with Step Collection

Many of the largest mobile manufacturing companies implement their own health application with their own API, which usually comes preinstalled on the user's phone. The user also has the option to download and install other health applications as they see fit. With the abundance of possible health applications, it is difficult for an application like FysBot to integrate directly into a specific API. For the test release of the FysBot prototype, only the Google Fit API was used to obtain steps. FysBot was implemented to obtain steps via Health Connect as well, but could not be released in the Google Play store without going through a verification step as mentioned in 4.2.7 (unlike Google Fit that could be used as long as FysBot is released in a testing stage). This revelation came in the latter part of the project, and due to the time it takes for Google to approve an app, the use of Health Connect was disbanded.

Health Connect is used to share health data between applications installed on the user's phone. Although a few devices currently come with Health Connect preinstalled (approx. 16%), this number will rise in the coming years. Since Health Connect is the better solution for obtaining steps, users can avoid installing an additional app in order to use FysBot. More importantly, the Google Fit API will no longer be available after June 30, 2025. This makes sense as there should ideally be no need to use multiple APIs to provide health data related experiences in an app. In conclusion, the necessity for verification to use Health Connect should have been recognized earlier, serving as the only method of obtaining the user's step data.

8.4 Lacking IOS Implementation

One of the initial goals of FysBot was to release it for both Android and IOS, as this would make it available to more users and make finding test users easier. The plan was to get a finished version for Android first, then port it to IOS. Although the client interface is written in a cross-platform language, some modifications are still required for an optimal experience. There are also some features that necessitate implementing native code (code that is designed for a particular platform) that would also have to be rewritten for IOS. It is unclear how long this will take, but finishing a working version on Android with the backend server, security, and AI takes a substantial amount of time. It was, therefore, decided that improving the quality of the application is more valuable at this time.

8.5 Steps for Releasing FysBot to Production

Currently, the prototype of FysBot is released through the Google Play Console as an internal test release. In order to do a production release, some steps must be completed. First, assuming that only Health Connect is used, a form has to be filled out explaining why and how the user's health data is used. The app can then be released to a closed testing stage. Google requires that an app is tested in closed testing for 14 days with at least 20 test users. After this test, a second form has to be filled out explaining the application and how it has been tested. In the next step, the address to the application's homepage must be provided, including a link to the application's privacy policy and terms of service. When all of this is done, the application can be released to production and installed by the public.

8.6 Gaining Insight into App Use

As stated earlier, a big part of the success of an application is understanding how it is used and discovering patterns that can be used to improve the user experience. In this project, this was accomplished through a usability test, but it can also be done continually by implementing user analytics tools (assuming the application has users). There also exist free-to-use analytics tools like Google Cloud Analytics[44] that can be easily integrated. Using tools like this, usage patterns like how the user navigates between tabs or interacts with buttons and other visual components can be monitored. There may be some legal obstacles if a cloud provider is used, especially when the application contains sensitive data. However, great alternatives to Google Analytics like Matomo[45] offer a free on-premise solution such that the user analytics data can be stored on self-hosted servers. With more time, integrating user analytics tools would greatly benefit analyses from future application tests and possibly production releases.

8.7 Standalone App vs. Social Media Deployment

In related studies, physical activity chatbots are often deployed on social media platforms[7][8] instead of standalone apps like FysBot. There are many reasons for doing this, so the choice of using a standalone app in this project should be discussed in more detail.

One of the reasons for choosing a social media platform, like Messenger or Telegram, is that they already have large user ases, making it easier to reach a larger population. In this case, one can avoid having the user install another app to use the chatbot as they are most likely already using a popular social media messaging platform. It could also cause the chatbot to be thought of as another friend, which could have positive psychological effects.

Another big reason for using existing social media platforms is that implementation is much quicker and requires less software development knowledge. The social media messaging platform already has a messaging interface, it secures data in traffic, and runs on the social media providers servers. To host the chatbot itself, an independent server is still needed to run the backend Natural Language Processing system, which makes the necessary API calls to chat with the user. There could also be a need for installing an independent app even if a social media platform approach is used. This could be to, for example, collect the user's steps and send them to the chatbot server.

The limitation of using existing social media messaging platforms is that there are few opportunities to implement features outside the messaging interface. Most often, one is restricted to the existing features of the messaging platform. One is also heavily restricted to the messaging platform's policies and the belief that the user's data is secure and not misused for political agendas[46] or otherwise.

Selecting the right messaging platform can be a problem. Among the youth in Norway, in terms of usage, the top-ranking platforms where messaging is a core component are: Snapchat (95%), Instagram (90%), Messenger (62%) and Discord (25%)[47]. Other platforms like WhatsApp and Telegram see less than 10% of usage[47]. Of these, Snapchat and Instagram are not suitable for implementing chatbots, and it is unlikely that the user already uses WhatsApp or Telegram. Norway also prohibits Norwegian ministers and officials from using Telegram along with TikTok, so these are also not considered safe options[48].

Discord is predominantly used by young males, while Messenger is widely used across all age groups. Therefore, Messenger is one of the few viable options for integrating a chatbot. A chatbot in Messenger was explored in the capstone [9], and it was found that Messenger offers an easy-to-use API for conversing with users. The problem with the Messenger chat API is that its design is focused on marketing and has a few restrictions that make it hard to implement Behaviour Change Techniques. One of these policies implements a restriction on when the chatbot is allowed to send the user a message. Specifically, the chatbot is only allowed to send the user a message within 24 hours of receiving a message from the user. When the user sends a new message to the chatbot, this 24-hour timer resets. This restriction was explored in the capstone project [9] and heavily affected the study by To, Quyen G and Green, Chelsea and Vandelanotte, Corneel[49], as the policy was implemented during the test trial of the chatbot. This is evident that sudden policy changes in a social media platform can significantly affect chatbot use. The authors of [49] later decided to implement a standalone application.

At the time of writing, Meta has released a new policy allowing them to use the users' data to improve their AI models, including messages sent between the user and professional accounts[50]. If FysBot were to be released on the Messenger platform, it would be considered a professional account, and thus, there would be some concerns about the privacy of the users' data.

The reason for choosing a standalone application in this project sums up to the following: **(1)** a desire to have a rich and extensive set of functionalities that benefit the user, **(2)** avoiding strict functional policies on how the chatbot may be used, **(3)** making sure the user's data and privacy is secured and not misused.

8.8 Design Revision After Usability Study

As discussed in the findings from the usability test, several issues with the chatbot were encountered, and immediate attention was required. Three major focus points for improving it were identified:

1. **Improved query routing**. The Routing Persona must be more reliable in its query routing.

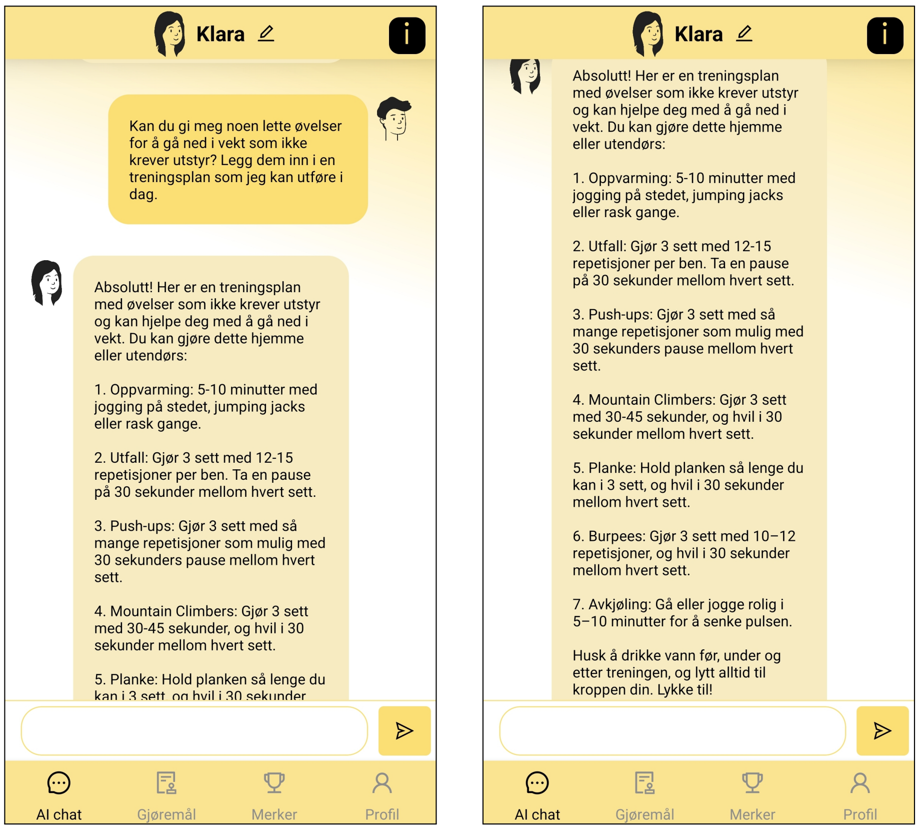
2. **More creativity**. The personas must be given the opportunity to be more creative in their responses.

3. **Global context**. A persona must be able to base its answer on previous conversations between the user and a different persona.

For the first point, a new design for routing queries has been implemented.

Previously, a message annotated with *user* was injected with the descriptions of all the other chatbot personas (figure 4.2), and the model was asked which of these descriptions best fit the given user query. In the new design, the Routing Persona instead utilizes OpenAIs function calling API (see function calling in section 2.4.2), where each of the other personas is given dummy functions that could be called. The query is then routed based on which function the model has decided to use. It is found that the RP now selects the correct persona much more frequently.

For the second point, the chatbot personas are now given much more freedom in their responses. Previously, their recommendations and suggestions were almost static, like the Exercise Recommender Persona who tries to find suitable exercises but then only gives the user a simple list (figure 6.4). In retrospect, this is not a good use of a powerful LLM like ChatGPT. In the newly implemented design, the personas are given the external data, but it is up to them to decide, based on the user's query, how to use this data in their response. This opens up the possibility for a much wider range of conversations. An example of this is shown in figure 8.1, where the ERP has to fetch the exercises from the database matching the user's specification and use them to satisfy the user's request.



Kan du gi meg noen lette øvelser for å gå ned i vekt som ikke krever utstyr? Legg dem inn i en treningsplan som jeg kan utføre i dag.

Absolutt! Her er en treningsplan med øvelser som ikke krever utstyr og kan hjelpe deg med å gå ned i vekt. Du kan gjøre dette hjemme eller utendørs:

1. Oppvarming: 5-10 minutter med jogging på stedet, jumping jacks eller rask gange.

2. Utfall: Gjør 3 sett med 12-15 repetisjoner per ben. Ta en pause på 30 sekunder mellom hvert sett.

3. Push-ups: Gjør 3 sett med så mange repetisjoner som mulig med 30 sekunders pause mellom hvert sett.

4. Mountain Climbers: Gjør 3 sett med 30-45 sekunder, og hvil i 30 sekunder mellom hvert sett.

5. Planke: Hold planken så lenge du kan i 3 sett, og hvil i 30 sekunder

Absolutt! Her er en treningsplan med øvelser som ikke krever utstyr og kan hjelpe deg med å gå ned i vekt. Du kan gjøre dette hjemme eller utendørs:

1. Oppvarming: 5-10 minutter med jogging på stedet, jumping jacks eller rask gange.

2. Utfall: Gjør 3 sett med 12-15 repetisjoner per ben. Ta en pause på 30 sekunder mellom hvert sett.

3. Push-ups: Gjør 3 sett med så mange repetisjoner som mulig med 30 sekunders pause mellom hvert sett.

4. Mountain Climbers: Gjør 3 sett med 30-45 sekunder, og hvil i 30 sekunder mellom hvert sett.

5. Planke: Hold planken så lenge du kan i 3 sett, og hvil i 30 sekunder mellom hvert sett.

6. Burpees: Gjør 3 sett med 10–12 repetisjoner, og hvil i 30 sekunder mellom hvert sett.

7. Avkjøling: Gå eller jogge rolig i 5–10 minutter for å senke pulsen.

Husk å drikke vann før, under og etter treningen, og lytt alltid til kroppen din. Lykke til!

(a) Asking a complex question. **(b)** ERP response.

Figure 8.1: Asking more complex questions using the new design.

Lastly, in the previous design, it was thought that a dedicated conversation memory for each chatbot persona would increase performance, but in practice, this was false. The usability study found that the user's follow-up questions to a previous conversation might not be destined to the same persona, thus breaking the continuity of the conversation. In the revised design, all personas now share the same memory, improving the conversation flow.

After implementing these three changes, the chatbot seems to have improved significantly, but ideally, a new usability study should be conducted to verify this.

After the completion of the usability study, some less critical issues were also fixed. Since many of the participants found the chatbot customization step in the create a user process unintuitive, the app's design has been altered with a new and more descriptive title. In addition, the daily schedule entries are now deleted after the event time has expired.

8.9 Facilitating Future Continuation of FysBot

FysBot will be used for further research on the effectiveness of physical activity chatbots, and thus, it should be easy to set up and run, and it necessitates handing over the Azure resources and source code. To enable quick setup of the system, the database and API server are placed in Docker[51] containers and can be run using Docker Compose. The containers are also set to start when the VM has booted, so all that is required to run the system is to press the "turn on" button in the Azure portal.

The Azure resources are currently located under the author's Azure subscription and has to be moved to the research group's subscription to avoid redoing the creation process. The necessary steps for this transfer has been researched, and the author will be available to assist once specific details about where the resources should be moved have been finalized. The author will also be available to assist in extracting data that is stored on the Azure VM after the transfer has been completed.

8.10 Research questions

In this section, the research questions are revisited and evaluated based on the findings of this thesis.

RQ1: How can a chatbot be developed using ChatGPT to help users become more physically active?

Through the use of the implemented chatbot personas, ChatGPT is instructed to provide information on different topics related to physical activity. Further, the use of OpenAIs function calling API makes it easy to extract relevant information from the user query. This information is then used to call other APIs or make database queries to give personalized responses. Whenever the topic of the user query does not match any of the persona descriptions, the chatbot still attempts to help the user by utilizing the vast knowledge already found in ChatGPT.

The chatbot personas are able to give exercise recommendations, hiking trip suggestions and review the users' step progress. They have been implemented with modularity in mind, allowing new personas to be added easily. The information they provide will help the user discover different options to be physically active and hopefully motivate them to maintain a more active lifestyle. Additionally, the chatbot is designed to avoid repetitive responses and, to a large degree, feel like a human. This helps retain user engagement and is made possible by using ChatGPT.

RQ2: How can a physical activity chatbot be implemented in a way that preserves the security and privacy of the users' data?

For a physical activity chatbot, there are cases where data that could be considered sensitive is transferred between the client and server as well as between the server and other APIs. Additionally, sensitive data, such as authentication credentials, messages, and health data, could be stored on the server. The FysBot system uses secure communication through TLS in all joints where communication over the internet is used. It also encrypts the users' credentials when stored on the server and protects against unauthorized access by securing API endpoints using JWTs.

Using the OpenAI API to use ChatGPT, assuring that the privacy of the users' data is maintained becomes an obstacle. FysBot handles this by using an Azure deployment of ChatGPT, located in Sweden, ensuring compliance with GDPR legislation and that the users' data is not used to improve future OpenAI models. It is also found that the data will not be moved from the location where the model is deployed. Even so, steps have been taken to ensure no personally identifiable data is included in the queries sent to ChatGPT.

RQ3: How can integrated interactive features enhance the effectiveness of a physical activity chatbot?

Although a chatbot has the potential to include several features just within the chat interface, it is thought that some are best suited in separate interfaces. FysBot provides badges for gamification and a scheduler to help the user plan and remember daily events. Having these features solely in the chatbot interface could be cumbersome for the user as it is challenging to find practical solutions that would not result in the user feeling overwhelmed, thus harming engagement. The features can still be bridged to the chatbot by enabling questions related to the badges or scheduled events.

9 Future Work

The work done in this project has led to a working prototype of FysBot, but there are still several improvements that can be made. The backend system can easily be extended to support new API endpoints if needed, and it is built on a robust foundation using Spring Boot, which offers a rich set of functionalities that can help the system scale. It also has a well-functioning security implementation that secures users' data both at rest and in traffic. However, some security measures should be implemented and evaluated before FysBot can be taken into the production stage. The frontend code has been implemented with a high degree of modularity, making it easy to implement new features.

The following discusses future work that can be done to further improve FysBot's user experience and system design. Section 9.1 gives a discussion of functionality that can be added to FysBot to improve user experience. Section 9.2 discusses adding a feature that allows health professionals to help users through FysBot. Section 9.3 discusses some of the system design changes that can be made to improve FysBot's backend system.

9.1 Improving User Experience

9.1.1 Voice Communication

OpenAI offers high-quality text-to-speech and speech-to-text models that could be used in FysBot if the budget allows. It is more expensive than text-only messages but could significantly improve the user experience for many users, especially those who struggle with text-based messaging. The implementation of this will cover user stories 13 and 14 of table 4.1.

9.1.2 Social Aspects

The users of FysBot could benefit from social interactions within the app to help them stay motivated in their goal of becoming more active. This could include chatting with friends and posting achievements on a timeline shared with all of the user's friends. By also having the means for creating workout plans, users could share these plans as well as coordinate group workouts so they can help each other be active. There could also be global events like challenges or competitions that are posted in which the user could compete with friends and have a leaderboard ranking. Implementing these social aspects will cover user stories 8 and 9 of table 4.1.

9.1.3 Gamification

Together with badges, avatars were first intended as another method of adding gamification to FysBot to increase the user's motivation to be active and have ties to the social aspects in section 9.1.2. The user's avatar was supposed to be customizable by applying customization options gained from reaching different step goals. Furthermore, the user could be rewarded with different avatars or receive other visual rewards to show their progress. Due to the lack of large sets of available open-use avatars of high quality and limited graphic design skills, this reward system has not been implemented. Attempts were made to use AI solutions like OpenAI's Dall-e image generator to create the avatars. However, this result did not align with the vision for the avatars, and no more time was spent on this topic due to not being a high-priority feature. This feature would be a great addition to FysBot for future work, especially in combination with the social aspects discussed in section 9.1.2.

9.1.4 Chatbot Personas

Through the usability study, it was found that the chatbot could, in some cases, give undesired responses. In most cases, these are related to mistakenly identifying a message as a request for step review or hiking trip suggestions, as discussed in section 8.1.2. A solution to this is explained in section 8.8 but should be further tested. Another solution that could be explored is using OpenAI's new Assistants API, which is currently in beta[52].

9.2 Professional Integration

Related to section 9.1.2, health professionals could be given access to review the progress of individual users to provide them with feedback or professional feedback on questions. There are some legal challenges related to this, and whether or not it should be included in FysBot depends on the direction in which it is taken. If used for the specific purpose of helping individuals who are undergoing treatment, this is a great feature to have and implements user story 11 from table 4.1.

9.3 System Reliability

Depending on the size of the user base, future work should consider the following properties of the system: scalability, availability, redundancy and resilience. Other authentication mechanisms should also be considered, like Firebase Authentication, which includes multi-factor authentication. Firebase also allows for different sign-in options by using, for example, Facebook or Google accounts. Ideally, a solution like Firebase's OpenID Connect solution should be used. Discussions on how the system could become more reliable are explored in section 4.2.9.

10 Conclusion

In this thesis, FysBot, an application aimed at increasing users' physical activity levels with a knowledgeable chatbot as its core component has been designed and implemented. In addition, an emphasis was placed on protecting the users' data through authentication and private communication. The use of ChatGPT to power the physical activity chatbot proved highly feasible through the use of chatbot personas that are specialized to handle different topics of conversation. While being open to any conversation, utilizing external datasets enabled the chatbot to provide personalized hiking trip suggestions, exercise recommendations and step progress reviews.

The design of FysBot is based on scientific evidence regarding what individuals seek to achieve by using a physical activity application with an integrated chatbot. A usability study conducted to test the implemented design using the think-aloud method gave valuable insight into how users interacted with the system. Through observations and feedback, several areas of improvement were identified, and updates were made to improve the application and chatbot. Overall, the participants found FysBot promising and effective in encouraging physical activity.

Bibliography

[1] Darren E.R. Warburton, Crystal Whitney Nicol, and Shannon S.D. Bredin. "Health benefits of physical activity: the evidence." In: *CMAJ* 174.6 (2006), pp. 801–809. ISSN: 0820-3946. DOI: 10.1503/cmaj.051351. eprint: https://www.cmaj.ca/content/174/6/801.full.pdf. URL: https://www.cmaj.ca/content/174/6/801.

[2] World Health Organization. "More Active People for a Healthier World." In: *Global action plan on physical activity 2018–2030: more active people for a healthier world* (2018).

[3] WHO. "*Age-standardized prevalence of obesity among adults*". Last accessed 15 April 2024. 2022. URL: https://data.who.int/indicators/i/BEFA58B.

[4] WHO. *Prevalence of overweight among adults*. Last accessed 15 April 2024. 2022. URL: https://www.who.int/data/gho/data/indicators/indicator-details/GHO/prevalence-of-overweight-among-adults-bmi-greaterequal-25-(crude-estimate)-(-).

[5] FHI. *Fysisk inaktivitet - voksne (indikator 7)*. Last accessed 2 April 2024. 2023. URL: https://www.fhi.no/is/ncd/fysisk-aktivitet/voksne/?term=.

[6] Corneel Vandelanotte et al. "Increasing physical activity using an just-in-time adaptive digital assistant supported by machine learning: A novel approach for hyper-personalised mHealth interventions." en. In: *J. Biomed. Inform.* 144.104435 (Aug. 2023), p. 104435.

[7] Wendy Wlasak, Sander Paul Zwanenburg, and Chris Paton. "Supporting autonomous motivation for physical activity with chatbots during the COVID-19 pandemic: Factorial experiment." en. In: *JMIR Form. Res.* 7 (Jan. 2023), e38500.

[8] Carol Ann Maher et al. "A physical activity and diet program delivered by artificially intelligent virtual health coach: Proof-of-concept study." en. In: *JMIR MHealth UHealth* 8.7 (July 2020), e17558.

[9] Sondre Elvebakken Løvås. *Exploring Chatbot Design and the Feasibility of securely implementing a chatbot using ChatGPT*. Unpublished work.

[10] C J Caspersen, K E Powell, and G M Christenson. "Physical activity, exercise, and physical fitness: definitions and distinctions for health-

related research." en. In: *Public Health Rep.* 100.2 (Mar. 1985), pp. 126–131.

[11] James O Hill, Holly R Wyatt, and John C Peters. "Energy balance and obesity." en. In: *Circulation* 126.1 (July 2012), pp. 126–132.

[12] Kevin D. Tipton and Robert R. Wolfe. "Exercise, Protein Metabolism, and Muscle Growth." In: *International Journal of Sport Nutrition and Exercise Metabolism* 11.1 (2001), pp. 109 –132. DOI: 10.1123/ijsnem.11.1.109. URL: https://journals.humankinetics.com/view/journals/ijsnem/11/1/article-p109.xml.

[13] Pedro A. Villablanca et al. "Nonexercise Activity Thermogenesis in Obesity Management." In: *Mayo Clinic Proceedings* 90.4 (2015), pp. 509–519. ISSN: 0025-6196. DOI: https://doi.org/10.1016/j.mayocp.2015.02.001. URL: https://www.sciencedirect.com/science/article/pii/S0025619615001238.

[14] Eleni Adamopoulou and Lefteris Moussiades. "Chatbots: History, technology, and applications." In: *Machine Learning with applications* 2 (2020), p. 100006.

[15] Ashish Vaswani et al. "Attention is All you Need." In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[16] Vipula Rawte, Amit Sheth, and Amitava Das. "A survey of hallucination in large foundation models." In: *arXiv preprint arXiv:2309.05922* (2023).

[17] Angie K Reyes, Juan C Caicedo, and Jorge E Camargo. "Fine-tuning Deep Convolutional Networks for Plant Recognition." In: *CLEF (Working Notes)* 1391 (2015), pp. 467–475.

[18] Inc Meta Platforms. *Retrieval Augmented Generation (RAG) for LLMs*. Last accessed 09 June 2024. 2024. URL: https://www.promptingguide.ai/research/rag.

[19] OpenAI. *Chat Completions API*. Last accessed 6 April 2024. 2024. URL: https://platform.openai.com/docs/guides/text-generation/chat-completions-api.

[20] GDPR EU Ben Wolford. *What is GDPR, the EU's new data protection law?* Last accessed 8 April 2024. 2024. URL: https://gdpr.eu/what-is-gdpr/.

[21] GDPR-info.eu. *Art. 5 GDPR: Principles relating to processing of personal data*. Last accessed 8 April 2024. 2024. URL: https://gdpr-info.eu/art-5-gdpr/.

[22] M. Goodrich and R. Tamassia. *Introduction to Computer Security*. Pearson, 2014.

[23] Eric Rescorla and Nagendra Modadugu. *Datagram transport layer security*. Tech. rep. 2006.

[24] Eric Rescorla. *The transport layer security (TLS) protocol version 1.3*. Tech. rep. 2018.

[25] Michael B. Jones, John Bradley, and Nat Sakimura. *JSON Web Token (JWT)*. RFC 7519. May 2015. DOI: 10.17487/RFC7519. URL: https://www.rfc-editor.org/info/rfc7519.

[26] Inc Meta Platforms. *Introducing JSX*. Last accessed 21 May 2024. 2024. URL: https://legacy.reactjs.org/docs/introducing-jsx.html.

[27] Pekka Abrahamsson et al. "Agile software development methods: Review and analysis." In: *arXiv preprint arXiv:1709.08439* (2017).

[28] Maarten Van Someren, Yvonne F Barnard, and J Sandberg. "The think aloud method: a practical approach to modelling cognitive." In: *London: AcademicPress* 11.6 (1994).

[29] Microsoft. *Data, privacy, and security for Azure OpenAI Service*. Last accessed 10 May 2024. 2024. URL: https://learn.microsoft.com/en-us/legal/cognitive-services/openai/data-privacy.

[30] Dillys Larbi et al. "What do adults living with obesity want from a chatbot for physical activity? – a qualitative study." In: *BMC Digital Health* 2.1 (2024), p. 15. ISSN: 2731-684X. DOI: 10.1186/s44247-024-00070-3. URL: https://doi.org/10.1186/s44247-024-00070-3.

[31] Inc Meta Platforms. *Discover the possibilities with Meta Llama*. Last accessed 09 June 2024. 2024. URL: https://llama.meta.com/.

[32] Inc. LangChain. *langchain 0.2.3*. Last accessed 13 June 2024. 2024. URL: https://api.python.langchain.com/en/latest/langchain_api_reference.html#module-langchain.chains.

[33] yuhonas. *free-exercise-db*. Last accessed 13 June 2024. 2024. URL: https://github.com/yuhonas/free-exercise-db.

[34] Den Norske Turistforening. *Finn din tur blant tusenvis av turforslag og hytter i hele Norge*. Last accessed 13 June 2024. 2024. URL: https://ut.no/.

[35] Norwegian Meteorological Institute (met.no) and the Norwegian Broadcasting Corporation (NRK). *Make something useful with data from the Meteorological Institute*. Last accessed 13 June 2024. 2024. URL: https://developer.yr.no/.

[36] Eugene et al. *Android API Levels*. Last accessed 13 June 2024. 2024. URL: https://apilevels.com/.

[37] Google. *Improve your app's security*. Last accessed 13 June 2024. 2024. URL: https://developer.android.com/privacy-and-security/security-best-practices#internal-storage.

[38] Microsoft. *What is Azure Application Gateway?* Last accessed 09 June 2024. 2024. URL: https://learn.microsoft.com/en-us/azure/application-gateway/overview.

[39] Microsoft. *What is Azure Application Gateway?* Last accessed 20 April 2024. 2023. URL: https://learn.microsoft.com/en-us/azure/application-gateway/ssl-overview.

[40] jq. *jq*. Last accessed 13 June 2024. 2024. URL: https://jqlang.github.io/jq/.

[41] OpenSSL Project Authors. *OpenSSL*. Last accessed 13 June 2024. 2024. URL: https://www.openssl.org/.

[42] Google. *Firebase*. Last accessed 13 June 2024. 2024. URL: https://firebase.google.com/.

[43] Google. *Google Play Console*. Last accessed 13 June 2024. 2024. URL: https://play.google.com/console/.

[44] Google. *Google Marketing Platform*. Last accessed 13 June 2024. 2024. URL: https://marketingplatform.google.com/about/analytics/.

[45] Matomo. *Matomo*. Last accessed 13 June 2024. 2024. URL: https://matomo.org/.

[46] Joanne Hinds, Emma J Williams, and Adam N Joinson. ""It wouldn't happen to me": Privacy concerns and perspectives following the Cambridge Analytica scandal." In: *International Journal of Human-Computer Studies* 143 (2020), p. 102498.

[47] Turi Reiten Finserås et al. "Reexploring problematic social media use and its relationship with adolescent mental health. Findings from the "LifeOnSoMe"-study." en. In: *Psychol. Res. Behav. Manag.* 16 (Dec. 2023), pp. 5101–5111.

[48] The Local. *Norwegian government bans ministers and officials from using TikTok*. Last accessed 1 June 2024. 2023. URL: https://www.thelocal.no/20230321/norwegian-government-bans-ministers-and-officials-from-using-tiktok.

[49] Quyen G To, Chelsea Green, and Corneel Vandelanotte. "Feasibility, Usability, and Effectiveness of a Machine Learning–Based Physical Activity Chatbot: Quasi-Experimental Study." In: *JMIR Mhealth Uhealth* 9.11 (2021), e28577. ISSN: 2291-5222. DOI: 10.2196/28577. URL: https://mhealth.jmir.org/2021/11/e28577.

[50] DAIR.AI. *Guidelines for privacy*. Last accessed 09 June 2024. 2024. URL: https://www.facebook.com/privacy/policy/version/25238980265745528.

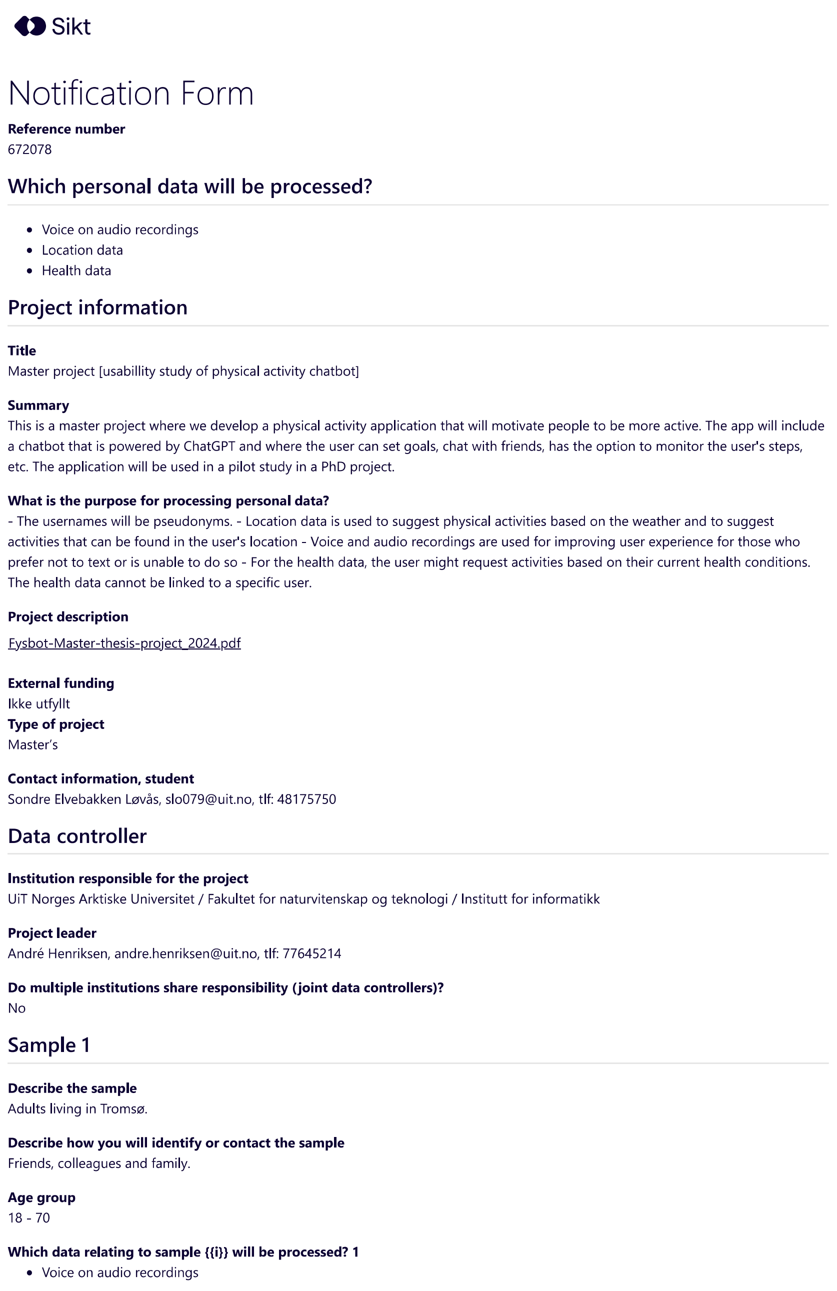
[51] Docker Inc. *Docker*. Last accessed 13 June 2024. 2024. URL: https://www.docker.com/.

[52] OpenAI. *Assistants API*. Last accessed 13 June 2024. 2024. URL: https://platform.openai.com/docs/assistants/overview.

A

User Testing Procedures

A.1 Sikt Notification Form



Sikt

Notification Form

Reference number
672078

Which personal data will be processed?

- Voice on audio recordings
- Location data
- Health data

Project information

Title
Master project [usabillity study of physical activity chatbot]

Summary
This is a master project where we develop a physical activity application that will motivate people to be more active. The app will include a chatbot that is powered by ChatGPT and where the user can set goals, chat with friends, has the option to monitor the user's steps, etc. The application will be used in a pilot study in a PhD project.

What is the purpose for processing personal data?
- The usernames will be pseudonyms. - Location data is used to suggest physical activities based on the weather and to suggest activities that can be found in the user's location - Voice and audio recordings are used for improving user experience for those who prefer not to text or is unable to do so - For the health data, the user might request activities based on their current health conditions. The health data cannot be linked to a specific user.

Project description

Fysbot-Master-thesis-project_2024.pdf

External funding
Ikke utfyllt
Type of project
Master's

Contact information, student
Sondre Elvebakken Løvås, slo079@uit.no, tlf: 48175750

Data controller

Institution responsible for the project
UiT Norges Arktiske Universitet / Fakultet for naturvitenskap og teknologi / Institutt for informatikk

Project leader
André Henriksen, andre.henriksen@uit.no, tlf: 77645214

Do multiple institutions share responsibility (joint data controllers)?
No

Sample 1

Describe the sample
Adults living in Tromsø.

Describe how you will identify or contact the sample
Friends, colleagues and family.

Age group
18 - 70

Which data relating to sample {{i}} will be processed? 1
- Voice on audio recordings

- Location data
- Health data

How will data relating to sample 1 be collected?

Participant observation

Legal basis for processing general personal data
Consent (General Data Protection Regulation art. 6 nr. 1 a)

Legal basis for processing special personal data
Explicit consent (General Data Protection Regulation art. 9 nr. 2 a)

Justify the choice of legal basis for processing

Information for sample 1

Will the sample receive information about the processing of personal data?
Yes

How does the sample receive information about the processing?
Written (on paper or electronically)

Information letter

infoskriv-gr1.pdf

Sample 2

Describe the sample
Individuals living with obesity.

Describe how you will identify or contact the sample
Advertisements and social media.

Age group
18 - 70

Are any of these groups included in the sample?
- Vulnerable groups

Which data relating to sample {(i)} will be processed? 2
- Voice on audio recordings
- Location data
- Health data

How will data relating to sample 2 be collected?

Online survey

Attachment

FysBot Spørreskjema.pdf

Legal basis for processing general personal data
Consent (General Data Protection Regulation art. 6 nr. 1 a)

Legal basis for processing special personal data
Explicit consent (General Data Protection Regulation art. 9 nr. 2 a)

Justify the choice of legal basis for processing

Information for sample 2

Will the sample receive information about the processing of personal data?
Yes

How does the sample receive information about the processing?
Written (on paper or electronically)

Information letter

infoskriv-gr2.pdf

Third persons

Will the project collect information about third persons?
No

Documentation

How will consent be documented?
- Electronically (email, e-form, digital signature)

How can consent be withdrawn?
An email or text message to the project leader can be sent to request withdrawal.

How can data subjects get access to their personal data or have their personal data corrected or deleted?
An email or text message to the project leader can be sent to request access, correction or deletion of their data.

Total number of data subjects in the project
1-99

Approvals

Will any of the following approvals or permits be obtained?
Ikke utfyllt

Security measures

Will the personal data be stored separately from other data?
Yes

Which technical and practical measures will be used to secure the personal data?
- Continuous anonymisation
- Encrypted transmission
- Encrypted storage
- Restricted access
- Other security measures

Indicate which measures
Users are required to create a profile with a pseudonym (username) and password to use the application. The server will then issue a signed token which includes the username and which will be used in any future requests unless the token is invalidated. The server protects users from other users by requiring valid token. Any tampering with the token will be noticed. The users are also protected from man-in-the-middle attacks by verifying the signature of the server token that is included in the server response. The token is signed using the servers' private key (keys are generated using RSA) and can be verified using the servers' public key. By using the username of the user, access to data located on the server can be revoked by invalidating the token. The server stored user credentials (password and token) is hashed and salted and is stored in a table separate to the user table. All traffic is done via HTTPS. Only Sondre Løvås and PhD co-supervisor Dillys Larbi will have access to the server which can only be accessed using SSH key-based authentication.

Where will the personal data be processed
- Hardware

Who has access to the personal data?
- Student (student project)
- Data processor
- Project leader

Which data processor will be processing/have access to the collected personal data?
Microsoft Azure, to set up a cloud virtual machine that is used as a server and to host the ChatGPT model.

Will personal data be transferred to a third country?
No

Closure

Project period
01.04.2024 - 10.06.2024

What happens to the data at the end of the project?
Personal data will be anonymised (deleting or rewriting identifiable data)

Which anonymisation measures will be taken?
- Personally identifiable information will be removed, re-written or categorized
- The identification key will be deleted
- Any sound or video recordings will be deleted

Will the data subjects be identifiable in publications?
No

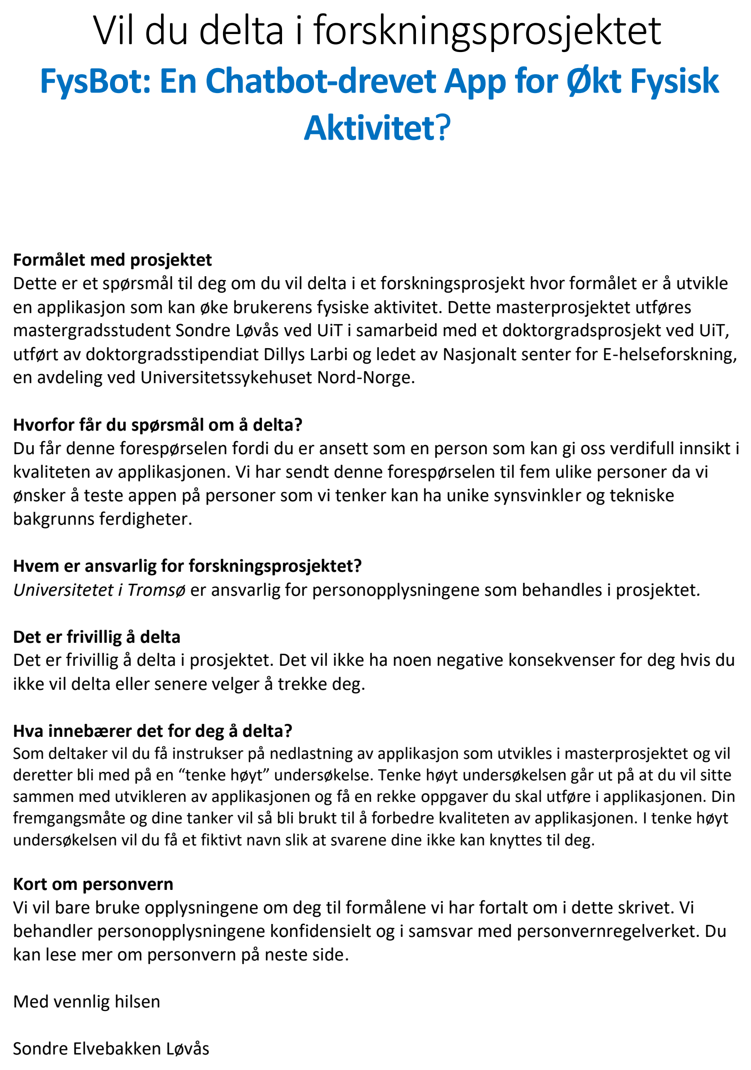
Additional information

The master project is part of a PhD project that has already sought approval from REC which was exempted. The REC decision for the main project is the one that will be added to this application. The decision is attached.

Other attachments

8) REK Decision_2021-12-07.pdf

A.2 Information Letter



Vil du delta i forskningsprosjektet

FysBot: En Chatbot-drevet App for Økt Fysisk Aktivitet?

Formålet med prosjektet
Dette er et spørsmål til deg om du vil delta i et forskningsprosjekt hvor formålet er å utvikle en applikasjon som kan øke brukerens fysiske aktivitet. Dette masterprosjektet utføres mastergradsstudent Sondre Løvås ved UiT i samarbeid med et doktorgradsprosjekt ved UiT, utført av doktorgradsstipendiat Dillys Larbi og ledet av Nasjonalt senter for E-helseforskning, en avdeling ved Universitetssykehuset Nord-Norge.

Hvorfor får du spørsmål om å delta?
Du får denne forespørselen fordi du er ansett som en person som kan gi oss verdifull innsikt i kvaliteten av applikasjonen. Vi har sendt denne forespørselen til fem ulike personer da vi ønsker å teste appen på personer som vi tenker kan ha unike synsvinkler og tekniske bakgrunns ferdigheter.

Hvem er ansvarlig for forskningsprosjektet?
Universitetet i Tromsø er ansvarlig for personopplysningene som behandles i prosjektet.

Det er frivillig å delta
Det er frivillig å delta i prosjektet. Det vil ikke ha noen negative konsekvenser for deg hvis du ikke vil delta eller senere velger å trekke deg.

Hva innebærer det for deg å delta?
Som deltaker vil du få instrukser på nedlastning av applikasjon som utvikles i masterprosjektet og vil deretter bli med på en “tenke høyt” undersøkelse. Tenke høyt undersøkelsen går ut på at du vil sitte sammen med utvikleren av applikasjonen og få en rekke oppgaver du skal utføre i applikasjonen. Din fremgangsmåte og dine tanker vil så bli brukt til å forbedre kvaliteten av applikasjonen. I tenke høyt undersøkelsen vil du få et fiktivt navn slik at svarene dine ikke kan knyttes til deg.

Kort om personvern
Vi vil bare bruke opplysningene om deg til formålene vi har fortalt om i dette skrivet. Vi behandler personopplysningene konfidensielt og i samsvar med personvernregelverket. Du kan lese mer om personvern på neste side.

Med vennlig hilsen

Sondre Elvebakken Løvås

Utdypende om personvern – hvordan vi oppbevarer og bruker dine opplysninger
Opplysningene som registreres om deg brukes kun for å kontakte deg om deltakelse av forskningsprosjektet. Dine personopplysninger vil ikke kobles til data som lagres eller bearbeides underveis ved bruk av applikasjonen. Applikasjonsdata som må overføres fra/til din mobil er alltid kryptert. Applikasjonsdata som må lagres på vår server vil også bli kryptert og vil ikke kunne knyttes til deg. Det er kun mastergradsstudent Sondre Løvås og doktorgradsstipendiat Dillys Larbi som vil ha tilgang til denne bruker dataen.

Etter prosjektslutt vil alle opplysningene om deg bli slettet. Du vil ikke bli gjenkjent i eventuelle publikasjoner.

Hva gir oss rett til å behandle personopplysninger om deg?
Vi behandler opplysninger om deg basert på ditt samtykke.

På oppdrag fra UiT har personverntjenestene ved Sikt – Kunnskapssektorens tjenesteleverandør, vurdert at behandlingen av personopplysninger i dette prosjektet er i samsvar med personvernregelverket.

Dine rettigheter
Så lenge du kan identifiseres i datamaterialet, har du rett til:
- å be om innsyn i hvilke opplysninger vi behandler om deg, og få utlevert en kopi av opplysningene,
- å få rettet opplysninger om deg som er feil eller misvisende,
- å få slettet personopplysninger om deg,
- å sende klage til Datatilsynet om behandlingen av dine personopplysninger.

Vi vil gi deg en begrunnelse hvis vi mener at du ikke kan identifiseres, eller at rettighetene ikke kan utøves.

Hva skjer med personopplysningene dine når forskningsprosjektet avsluttes?
Prosjektet vil etter planen avsluttes 10. juni 2024.

Kontaktinformasjon og andre direkte gjenkjennende opplysninger vil bli slettet umiddelbart etter prosjektslutt. Anonymisert data vil bli arkivert på en sikker måte.

Spørsmål
Hvis du har spørsmål eller vil utøve dine rettigheter, ta kontakt med:
- André Henriksen
  andre.henriksen@uit.no
  +47 77 64 52 14
- Vårt personvernombud: personvernombud@uit.no.

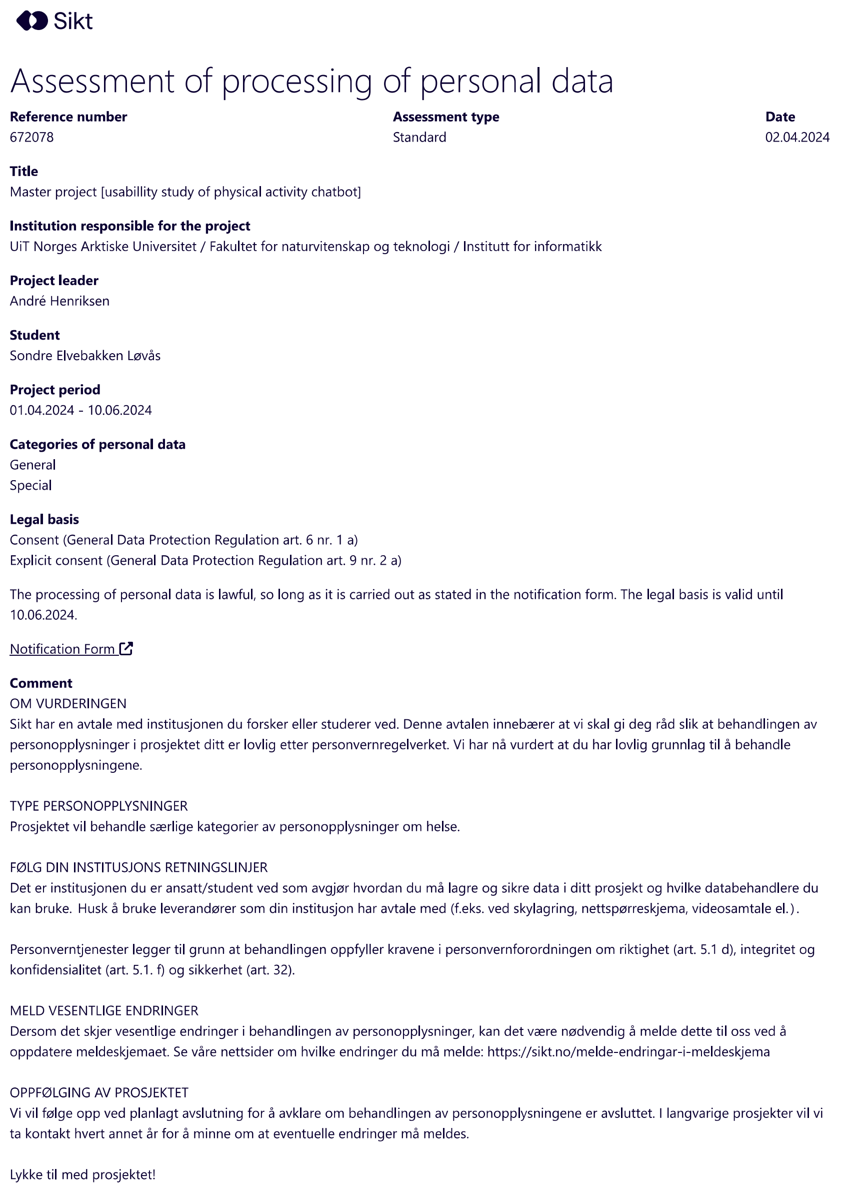
Hvis du har spørsmål knyttet til Sikts vurdering av prosjektet, kan du ta kontakt på e-post: personverntjenester@sikt.no, eller på telefon: 73 98 40 40.

Jeg har mottatt og forstått informasjon om prosjektet FysBot: En Chatbot-drevet App for Økt Fysisk Aktivitet, og har fått anledning til å stille spørsmål. Jeg samtykker til:

- ☐ å delta i applikasjons-testing
- ☐ å delta i tenke høyt undersøkelsen

Jeg samtykker til at mine opplysninger behandles frem til prosjektet er avsluttet

A.3 Sikt Assessment



Sikt

Assessment of processing of personal data

Reference number	**Assessment type**	**Date**
672078	Standard	02.04.2024

Title
Master project [usabillity study of physical activity chatbot]

Institution responsible for the project
UiT Norges Arktiske Universitet / Fakultet for naturvitenskap og teknologi / Institutt for informatikk

Project leader
André Henriksen

Student
Sondre Elvebakken Løvås

Project period
01.04.2024 - 10.06.2024

Categories of personal data
General
Special

Legal basis
Consent (General Data Protection Regulation art. 6 nr. 1 a)
Explicit consent (General Data Protection Regulation art. 9 nr. 2 a)

The processing of personal data is lawful, so long as it is carried out as stated in the notification form. The legal basis is valid until 10.06.2024.

Notification Form

Comment
OM VURDERINGEN
Sikt har en avtale med institusjonen du forsker eller studerer ved. Denne avtalen innebærer at vi skal gi deg råd slik at behandlingen av personopplysninger i prosjektet ditt er lovlig etter personvernregelverket. Vi har nå vurdert at du har lovlig grunnlag til å behandle personopplysningene.

TYPE PERSONOPPLYSNINGER
Prosjektet vil behandle særlige kategorier av personopplysninger om helse.

FØLG DIN INSTITUSJONS RETNINGSLINJER
Det er institusjonen du er ansatt/student ved som avgjør hvordan du må lagre og sikre data i ditt prosjekt og hvilke databehandlere du kan bruke. Husk å bruke leverandører som din institusjon har avtale med (f.eks. ved skylagring, nettspørreskjema, videosamtale el.).

Personverntjenester legger til grunn at behandlingen oppfyller kravene i personvernforordningen om riktighet (art. 5.1 d), integritet og konfidensialitet (art. 5.1. f) og sikkerhet (art. 32).

MELD VESENTLIGE ENDRINGER
Dersom det skjer vesentlige endringer i behandlingen av personopplysninger, kan det være nødvendig å melde dette til oss ved å oppdatere meldeskjemaet. Se våre nettsider om hvilke endringer du må melde: https://sikt.no/melde-endringar-i-meldeskjema

OPPFØLGING AV PROSJEKTET
Vi vil følge opp ved planlagt avslutning for å avklare om behandlingen av personopplysningene er avsluttet. I langvarige prosjekter vil vi ta kontakt hvert annet år for å minne om at eventuelle endringer må meldes.

Lykke til med prosjektet!

A.4 Think Aloud Protocol

FysBot Project

Think-aloud protocol – max 60 minutes

Description:

Concurrent think-aloud method – participants complete specific tasks and think aloud simultaneously. That is, participants voice out everything they see, do or think related to the assigned tasks.

Scope & Purpose

- Problem discovery – identify problems related to the following:
    - Terminology – terms/words used in the app and by the chatbot
    - Data entry – inputting data via free text, predefined text and voice
    - Layout & Structure – placement of icons and buttons in the app
    - Comprehension – chatbot's understanding and continuity of text/conversation
    - Relevance – how relevant are the features and functions to increasing physical activity
    - Feedback – suggestions on how to improve the application/chatbot

Metrics:

- Successful completion of tasks
- Time required to complete tasks (in minutes)
- Identified Errors: Critical – necessary to enhance the functioning of app/chatbot; Non-critical
- Satisfaction
- Ease of use/ Learnability
- Appearance (prompt to say something about this if necessary)

Procedure:

- Welcome participant
- Explain the reason the participant is there. Show example of think aloud.
- Ask if the participant has read, understood and signed the informed consent
- Take the informed consent from the participant and check if it has been signed
- Ask if they have any questions about the testing
- **Explain the process:**

You will be asked to perform certain tasks related to the app/chatbot. In the process of performing these tasks, you are required to think out loud. You do not need to think about what you are going to say first. Just say whatever comes to mind when you look at the screen and perform the tasks, including saying the things you are doing to perform the task. You can also comment on the appearance of the app, chatbot, buttons, etc. There is no wrong way to do this, so just feel free to say anything that comes to mind.

Thank you for agreeing to participate. Let us begin!

- Remember to prompt participants to keep talking, if necessary.

Tasks:

	Time to complete task (mins)	Task completed (Y- Yes; N- No)	Errors identified (C- critical; NC – Non-critical)
1. Download and install the FysBot app			
2. Sign in to FysBot – create a username and password			
3. Give the chatbot a name			
4. Set step goal			
5. Start chatting with the chatbot			
6. Find and read information about FysBot			
7. Ask the chatbot to recommend exercises			
8. Change step goal			
9. Make a physical activity plan for the day			
10. Ask the chatbot to give feedback on steps			
11. Ask/Check for overview of physical activity			
12. Ask the chatbot a physical activity-related question			
13. Ask the chatbot a general question			
14. Check the overview of steps			
15. Check how many badges you have received			
16. Ask the chatbot to suggest a hike			

Pre-test/Post-test checklist

- ☐ Preparation – print informed consent, find room for testing, recruit participants by invitation
- ☐ Scheduling – find out when participants, observers and a room are available
- ☐ Location – convenient for participants
- ☐ Prototype – FysBot app/chatbot
- ☐ Observers - 2 people observe and write down what participants say. Compare notes at the end of testing