TU WIEN Informatics

Designing a Software System for Finding Proper Timings of Health-Related Tips and Tasks

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Julian Thöndel, BSc
Matrikelnummer 11777732

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Doz. Mag.rer.nat. Dipl.-Ing. Dr.techn. Rudolf Freund

Wien, 3. Oktober 2024

Unterschrift Verfasser	Unterschrift Betreuung

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Designing a Software System for Finding Proper Timings of Health-Related Tips and Tasks

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Software Engineering and Internet Computing

by

Julian Thöndel, BSc
Registration Number 11777732

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Doz. Mag.rer.nat. Dipl.-Ing. Dr.techn. Rudolf Freund

Vienna, 3rd October, 2024

Signature Author	Signature Advisor

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Julian Thöndel, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 3. Oktober 2024

Julian Thöndel

Acknowledgements

First, I want to thank my advisor Rudolf Freund for his advice and support throughout the writing of this Master's thesis. I would also like to thank Christoph Aigner for his helpful input and support throughout this project.
Furthermore, I would like to thank the Austrian Institute of Technology for providing much of the necessary hardware, as well as friends and family for providing iOS devices to make the development of the software component featured in this paper possible.
Finally, I would like to thank my former colleagues at the Austrian Institute of Technology for their consultation during the planning of this project and the development of the software system.

Kurzfassung

Smartphones sind in den letzten zwei Jahrzehnten immer leistungsfähiger geworden. Ihr Nutzungsbereich geht weit über den ursprünglich vorgesehenen Zweck der Telefonie hinaus. Ein häufiger Anwendungsbereich von Smartphones ist die Nutzung von Fitness- und Gesundheitsanwendungen. Viele Anwendungen in diesem Bereich setzen Push-Benachrichtigungen ein, um ihre Benutzer zu positiven Verhaltensänderungen zu motivieren oder Erinnerungen zu schicken. Es ist allerdings entscheidend, dass diese Benachrichtigungen zum richtigen Zeitpunkt ankommen. Diese Arbeit untersucht die Antwortrate von Nutzern auf verschiedene Push-Benachrichtigungen basierend auf ihrer aktuellen Situation. Dazu wurde eine Smartphone-App entwickelt, die Push-Benachrichtigungen empfangen und anzeigen kann und das Antwortverhalten der Nutzer aufzeichnet. Darüber hinaus wurde ein Backend, welches aus mehreren Microservices besteht, entwickelt, um die Antworten der Nutzer zu speichern. Basierend auf historischen Daten entscheidet dieses, ob ein Nutzer basierend auf seiner aktuellen Situation auf eine Nachricht reagieren wird oder nicht. Die Situation eines Nutzers wird mithilfe von mehrere Variablen gemessen, wie zum Beispiel die Tageszeit, Bewegungsstatus und dem Aufenthaltsort eines Nutzers. Während einige dieser Variablen bereits von anderen Arbeiten für Planungsalgorithmen verwendet wurden, werden in dieser Arbeit zusätzliche Parameter verwendet, die von einem Garmin-Fitness-Tracker bereitgestellt werden: Stress- und Bewegungsdaten. Für diese Arbeit wurde eine erste Pilotstudie mit fünf Testpersonen und einer prototypischen Implementierung der mobilen Anwendung inklusive Backend durchgeführt.

Abstract

As mobile phones have become more and more capable over the past two decades, there have been more use cases for them way beyond the originally intended scope of telephony. One common use case for smartphones nowadays is using fitness and health-improvement applications. Many fitness and health applications use push notifications as a tool to invoke behavioral change in a user or to send reminders. However, it is important to time these notifications properly. This thesis explores the response rate of users to different push notification messages, depending on their current situation. For this research, a mobile app was built in order to receive these notification messages and track users' response behaviour. In addition, a backend consisting of several micro services was developed in order to persist the responses of users, and - based on them - decide on whether or not a user has a good enough chance to respond to a message according to their current situation. The situation of a user is measured by several variables such as the time of day, whether or not the user is currently moving and the current location of the user. While some of these variables were already used by other works for scheduling algorithms, this paper attempts to use additional parameters measured by a Garmin Fitness tracker: The stress and physical activity levels of a user. For this paper, a first pilot study is conducted using 5 test persons and a prototypical implementation of the mobile application and the backend. While it does not provide any definitive evidence regarding the precise impact of the Fitness Tracker parameters, it does provides some valuable insight for future studies with a greater sample size.

Keywords: *Notifications, Garmin, Behavioural change, Just-in-time adaptive intervention, Health*

Contents

CHAPTER 1

Introduction

To provide users with valuable health information throughout their day, many software products have been developed to display notifications on a user's smartphone throughout their day. Many practical applications for health-based smartphone notifications can be found in modern lifestyle apps, such as a motivational reminder to go for a jog or to drink enough water today. While most apps focus on what content could be displayed to the user, a more recent field of research is to also find the proper timings to display a message to a user.

1.1 Problem Statement

In 2014, Pielot et al. [1] found that participants of a study they were conducting at the time received more than 63 smartphone notifications a day. According to several new sources in 2023 [2], [3], teenagers receive more than 220 smartphone notifications a day in recent years, while a statistics report from 2024 suggests that an average US smartphone user receives 46 push notifications a day [4]. Improperly timed notifications might get lost in a sea of other notifications [5]. Timing notifications properly, for example by delaying news updates instead of sending them instantly [6], can increase the response rate of users.

The problem with notification message timings is that if a user is notified during an unfitting point in time, it is not very likely that they will respond to that particular notification.
While that alone is not a problem in itself, users can get annoyed by an app that repeatedly sends out messages.
To get proper responses from users, it is not only crucial to show users content that is relevant to them at the moment of the deployment of the message but also to send messages at a moment when a user is actually able to respond to act based on the notification. If a user is currently unavailable (for example if the user is currently in a

meeting or driving a car) the notification might be left unanswered for a longer period of time. Note that users might still respond while driving [7], but it is unsafe to encourage them to do so. If the user is only able to respond to the message at a later time, the message might become irrelevant. So in order to motivate positive behaviour from users, it is important to deploy notifications at a point in time where they can respond.
Nevertheless, it is detrimental for an application to send out many notifications in hopes of randomly guessing the perfect timing for one of them. As a result of an overload of poorly timed messages, a user might stop paying attention to the notification messages, mute them - or as an absolute worst case – uninstall the application, resulting in no further opportunities to enforce health-beneficial behavior in users.

1.2 Motivation

Since it can be problematic to send out notifications at random times, it would be advantageous to have a system determining *proper* timings and fitting content depending on the current situation of a user.

The *proper* timing of a notification refers to a point in time where a user is in a state where they can actually *perceive* the message within a short amount of time after the message deployment, and is also likely to *respond* to that message instead of dismissing it. For properly timing interventions, a JITAI [8], [9] (Just-In-Time Adaptive Intervention) must make an educated guess about the users' receptivity in order to determine the success rate of a message. Multiple papers [10], [11] exist investigating the detection of user receptivity. If a user is in a *perceptive* state is independent of the message contents which are sent, but whether or not the message will actually motivate the user to respond in their current situation is dependent on the contents of the notification.

The goal of this work is to develop a system that can pick fitting notification contents at the right time and send them out to users so that they are very likely to respond with that notification message. The system should send as many properly timed messages as possible, to increase user engagement with the application. Furthermore, the system should provide as few improperly timed notifications as possible, since too many of them can lead to annoyance and the user disliking the application.
This way, the maximum amount of health tasks can be sent to the user, without the user being annoyed or bothered. Ideally, this leads to as much healthy behavior invoked by the application as possible for one user.

1.3 Goals

In order to answer the research questions of this paper, a prototypical software system was developed. The system consists of a mobile app developed in Flutter, which will be

available for both Android and iOS smartphones. This application will track the so-called *context* of a user, therefore their current position, stance, and stress. The system will send this information to a backend application. The backend will then send out notifications with health-related tips to its users, and will track under which conditions a user was likely to respond, and under which conditions a response was unlikely.

At first, the timing when a user receives a message will be random. Using a user's current context, and their history under which conditions notifications were answered, the backend system will make a decision about whether or not a user will respond to a message or not. With every answer, the system is supposed to make a more accurate estimate of a user's response probability than previously.

The prototype will be used in an experimental test run with several users. Participants will get one week of randomized messages, followed by another week of messages with personalized message timings.
The goal is to improve the amount of times a user clicks on notifications, instead of just ignoring them. Ideally, the response rate of any user would improve with personalized message timings compared to just sending out randomly timed messages.
Furthermore, another important goal of this paper is to compare the stress values of user with their likeliness to respond to messages. As already mentioned, a customary Garmin fitness tracker will be handed out to every participant, and their stress values will be measured throughout the experiment. Using a user's tracked stress values, the system will ideally figure out whether or not users are less likely to respond to notifications when their Garmin watch measures a higher stress level. It is expected that a user's response rate will differ based on the stress value measured by their Garmin watch.

Research Questions

The wider goal of this thesis is to build a prototypical implementation of a notification scheduling system for health-related tips and tasks, to answer the following research questions:

> **Research Question 1:** Which parameters of a user's state should be considered when evaluating proper timings of notifications?

This question is addressed by conducting literature research on other similar studies, to figure out which parameters were commonly used in state-of-the-art experiments, and which ones influenced the likeliness to respond the most. Furthermore, a set of variables including stress and physical activity was picked for the prototype developed for this thesis based on the research. After a test run, it was evaluated which parameters had what impact on the likeliness to respond to a message. Moreover, the meaningfulness of including a user's physical activity and stress parameters of a Garmin watch in a user state was assessed.

Research Question 2: Can the notification response rate be improved with individualized message timings by tracking stress?

This was assessed by building the prototypical software system which includes the stress level of a user measured by a Garmin watch. After a test run, the individual response rate to the messages will be compared between randomly and individually timed messages, to see if any improvements can be seen.

Research Question 3: How should a prototype that implements adaptive notification scheduling look to increase the user response rate?

Literature research will be used to settle on a basic computer learning model which suits this task. A test run will be used to check if this model works and to identify if the trained model identifies points in time where notifications are likely to be answered. After the test run, the model will be re-evaluated and improved upon based on the results.

1.4 Methodology

The main methodology used for this thesis is multiple iterations of a software prototype with evaluations of the current state of the prototype in between.
To get a baseline for standards that should be met by the initial prototype, literature research was conducted to find the current state of the art of scheduling strategies, as well as the commonly used parameters to identify a user's current state. Based on this research, a design for the first prototype software of the scheduling system was carried out. Afterward, the first version of the prototype was developed and tested with a group of five test persons. Test persons were subjected to two phases, a phase in which they received messages randomly throughout the day, and a second phase where the system tried to deliberately decide whether or not it was a good moment to send a notification to a user. Once this first test run was completed, it was evaluated if the deliberate strategy improved response rates. Additionally - by analyzing the data manually - ways of improving the scheduling strategy for the second test run and improving the application further were identified.
Based on the informal analysis of the first iteration, decisions were made regarding further developments, and actions were taken accordingly. This lead to a final prototype, which was tested again. The results were then evaluated using interviews with the attendees about their personal experience with their application and by statistical analysis of the response rates of random messages and deliberate messages. A visual representation of the workflow can be seen in 4.1. During one test run, a user first goes through an *"Explorative Phase"*, where messages are sent randomly. This is used to gather data on the response behavior of users. After that follows a *"Deliberate Phase"*, where the gathered data is used to make educated decisions on when to send a message. The results of this are then evaluated to find out if the chosen message delivery strategy works.

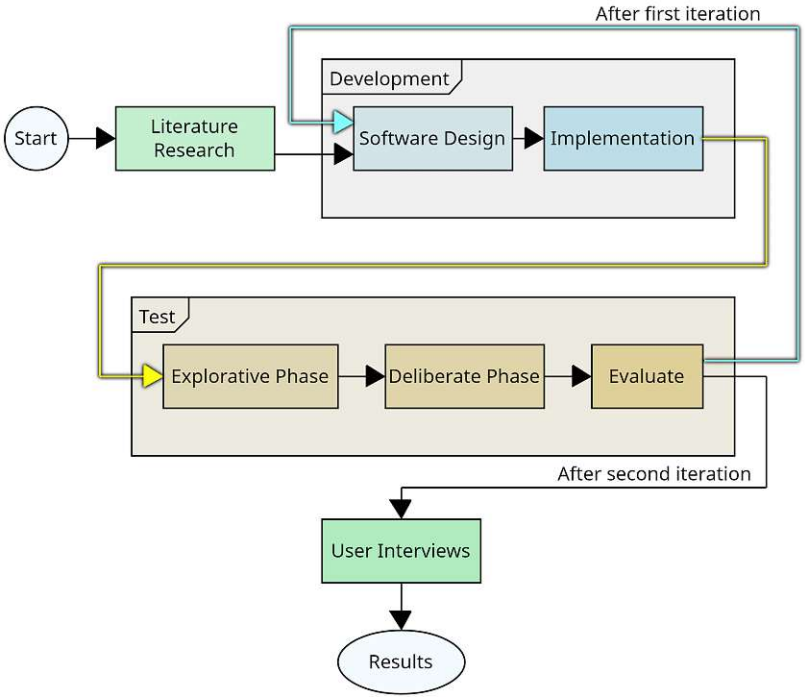


Figure 1.1: Flowchart of the Workflow

Acknowledgements

This research was initiated by the AIT - Austrian Institute of Technology. During the development of the software project made for this thesis, the author of this paper was employed at the AIT, and the AIT requested the development of a software system to time smartphone notifications for health-related tips and tasks, based on a user's smartphone and Garmin data. The hardware for the infrastructure and fitness trackers, as well as the iMac to develop the iOS version, were provided by the AIT. Therefore the choice of which fitness trackers were used was also made by AIT. The notification contents are based on another project from the AIT.
Besides the Garmin API setup and the used database for both Garmin and application-data, the backend software components, mobile applications, scheduling strategies, and software design were created for this thesis.

CHAPTER 2

Theoretical Background

2.0.1 Important terms

To describe certain aspects of the timing of notifications, several terms need to be explained.
First of all, messages that are supposed to be sent out at the right moment are commonly referred to as **"just-in-time adaptive interventions (JITAI)"** or just **"just-in-time interventions (JITIs)"**. JITAIs attempt to provide users with the right amount of messages at the right time and with the right context. The timing of JITAIs is decided using a user's **context**. The *context* of a user is the current situation a user is in. Section 3.1 will further elaborate on how a user's context can be measured [12].

A user's **receptivity** is defined as "the individual's transient ability and/or willingness to receive, process, and utilize just-in-time support". It depends on both internal (for example the user's current mood) and external factors (the user's location etc.). Sending out notifications while a user is in a non-receptive state will not be beneficial in any way, and might negatively impact a user's engagement with an application in the worst case [12].

2.1 Benefits and Problems of Push Notifications and Behaviour Change Apps

Push notifications provide a great opportunity to actively interact with users. "Classical" news, information, or social media apps are great at presenting information to the user on demand, but as long as there is no mechanism like push notifications in place, there is no way to interact with the user actively. Without them, the usage of an application is only dependent on the user's personal motivation to use the application. This can be a problem if using the app is not particularly fun or intrinsically motivating. This is

sometimes the case with applications that try to get a user to behave in a certain positive way. But it is often hard to self-motivate oneself into better life habits, and many times, it is not exactly pleasant to switch from junk food to a healthy meal. In 2016, a meta-study by Alkhaldi et al. [13] was performed analyzing different intervention-based approaches based on their duration, frequency, mode of delivery, and other factors. According to this meta-study, many of the included studies seemed to conclude that technology-based interventions can improve user-engagement.
Imagine a user downloading an application like *lifesum* [14], which is a lifestyle app to improve a user's dieting habits. While intrinsic motivation is present while the user is downloading the app, the willingness the use the app might decline over time.
However, with the help of push notifications, it is possible to form a habit of a person using an app continuously. When a user starts with a new behavioral change application, the application might send notifications with a specific action as a goal in mind. Over time, the goal-based notifications might become less important, as the user checks in regularly on the application they form a habit of engaging with the application due to push notifications [15]. In the case of lifesum, interventions like this can happen with a message like *"Did you drink enough water today? Enter how many glasses of water you drank today!"*. Not only does a message like this motivate the user to keep track of their water consumption, but it also gives them a reason to open the app again, and maybe also check their other statistic for today. By doing this regularly, more incentive is given to the user to open the self-improvement app on a regular basis, than there would be by just relying on the user consistently thinking to open the app. As a best-case scenario, the push notifications help the user form a habit of checking the app regularly[14].
As established, push notifications can play an important role in promoting positive behavior and improving the usage of an application aimed at improving positive behavior. Unfortunately, there can also be negative effects of using push notifications. If messages are spammed, or seen as irrelevant, this can lead to a user either muting the application - which takes any opportunity away from the application to ever invoke positive behavior again - or uninstalling the application - which is even more disastrous, since the application can not provide any "passive" information either. And even if the application stays not muted and can deliver notifications to a user, the messages might be dismissed or seen as annoying or stressful. Therefore, the timings, frequency, and content of such push notifications must be carefully picked, to avoid negative consequences, and maximize the positive effect of a behavior change application. For these parameters, there is unfortunately no one-size-fits-all solution, good values for these parameters can vary between people [16].

2.1.1 Social Acceptance of Behaviour Change Apps

One factor that can influence the effectiveness of push notifications and behavioral change applications is social acceptance. If a user is embarrassed to use an application or perform a beneficial action in their current situation, the effectiveness of the app is limited. According to Choi et al. [17], it is therefore important to not prompt users to do socially-deviant actions when users are under the assumption that this task can not

be performed at the moment.

Social Acceptance of Self-Improvement Apps

According to a qualitative study by Dennison et al., people can view self-improvement applications as embarrassing. Many want to keep insecurities about their weight and unhealthy lifestyle to themselves and might find it inconvenient if someone else sees they are using an application that tries to motivate them to exercise or to lose weight. This gets especially frustrating when external human beings have insight into the behavior change statistics. Some applications allow friends to see updates about a person's fitness or health data, supposedly to encourage the person to live healthier in order to not have friends and family see their low fitness statistics at the end of the month. This was seen as embarrassing and frustrating by one of Dennison's test subjects.
Manual sharing of fitness data (for example by a "Share on Facebook" button after a run) was seen as a motivating thing by some users, but automatic sharing of (non-) accomplishments was unwanted by the vast majority of users [16]. Other studies for health-related apps also reported that some users will find the competitive nature of a social media activity app motivating [18]–[20]. Also, multiple papers exist exploring the social and psychological aspects of users stopping (or continuing) to use health apps after some time and the barriers users face to continuously use the application. Some reports conclude that 25% of mobile health applications are only opened once after the installation [21]. One of the reasons why users might stop using the application is due to the absence of perceived usefulness [22]. Free applications with fewer features struggle with this especially [23]. If a mHealth application offers a variety of features that are perceived useful by a user, that user is more likely to keep using the application [24]. Peng et al. [25] investigate impediments users might face to continue or start using health applications. Multiple sources [26], [27] conclude that personalization, reminders, good app design, and gamification features can help to motivate users to use the application regularly and make interventions more appealing.

Social Deviance of Actions

Another important thing is the social acceptance of the prompted actions themselves. Whether or not an action is socially deviant, depends on the current situation of the user. When an intervention is sent out to the user, the user must have the impression that this action is performable when reading the message. When an action is not seen as socially acceptable right now, users will not perform it. This also depends on context. The prompt *"Stand up and walk around for a bit"* might lead the user to actually stand up and move around when they are home alone but will be seen as socially deviant while in the middle of an important meeting. Therefore it is important to pick socially neutral actions, or at least fitting for the situation a user is currently in [17].

2.1.2 Importance of individualizing message timings

Current research suggests, that poorly timed messages will be discarded or ignored. Notifications should be provided at moments in time when they do not disrupt a user's daily routine so that they can be acted upon immediately. One way to deliver messages which consider the daily routine of a user is to first ask the user to set time frames where they want to receive messages, and when they do not want to see any. The issue here is, that users themselves often have varying routines, or are not very good at anticipating when would be a good time to receive messages [28]. In the past decade, several experiments were carried out to explore the benefit of specifically timed messages when compared to randomly timed messages. It was shown that the response rate and response time of users can be greatly improved by tailoring the timings. [29], [30] In a study by Pham et al., the benefits of using notifications were explored. Their study found that the usage of notifications can help to improve the time users spend on their application. However, their research also showed that the user's attention and time are not infinite, free resources. The frequency at which messages are sent should be carefully considered. On the other hand, if timed correctly with the right frequency, notifications encouraged users to keep using the application even several days after the installation, compared to a control group which did not receive any messages at all. [31]

2.1.3 User engagement with wrongly timed messages

If push notifications do not hold any positive value for a user, they will be seen as annoying spam. It is important how many messages are sent out. Even if the timing is okay for a user, if they are overwhelmed by many notifications of an app in a short time frame, they might uninstall it, which is the worst-case scenario for an app provider. As a lighter consequence, a user might use the operating system's built-in feature to disable notifications for all or one specific app. In a study by Pham et al., the effects of different smartphone notification frequencies are explored. While the study shows that notifications can generally boost user engagement with an app, it also shows that notifications can have a negative effect if overused. For example, 58% of participants in their study stopped using the app within 24 Hours, if they received a reminder message every 3 hours. [31], [32] One thing to keep in mind is that a user "dismissing" a notification will not always result in the notification not impacting the user at all. Even if a notification is not tapped or interacted with, the user might have seen the notification and its contents without acting upon it [30]. But programmatically determining if a user actually read a message is quite hard since the mobile operating systems do not offer a feature to determine if and how long a notification appeared on a user's screen, and it can not be determined with certainty whether or not the user actually read the message. Therefore, notifications will be considered as "dismissed" if a message was sent and there was no action performed (meaning the notification was not opened) afterward.

2.1.4 Content of Notifications

If one aims to improve the response rate to notifications, it is not only important when a message is received, but also what kind of message. The content of a message will not change whether or not a user is in a perceptive state. The content of a message will not determine *if* a user sees a message, but only if the user will actually act upon a certain message, or if the message is perceived as useful. An important observation is that message content is generally seen as more meaningful if it is tied to new, communication, or some event. Notifications of social media or messenger apps are more likely to grab the user's information than other notifications. It suggested that users are more likely to respond and respond faster to social media and messenger prompts, due to the social pressure tied to them. People want to respond to other people as fast as possible, while notifications that simply include information or suggestions are easily seen as something that does not require immediate action and might be postponed or dismissed. Furthermore, users will quickly lose interest in the content of notifications, if their content is repetitive. Even if a pool of several different suggestion notifications is provided, users will quickly perceive them as repetitive if they are too similar. Ideally, content displayed to the user should be relevant in the here and now require immediate action (for example a message from a friend), and be different enough from the preceding notifications, in order to keep the notification content interesting and exciting enough to grab the user's attention [28], [30].

2.2 Garmin

As stated in 3.1.3, the stress level of a user can be an important part when considering a user's context. The issue is that measuring a user's stress subjectively takes time, can be biased or falsified by a user, and an accurate objective measurement requires special measuring tools, which can often be impractical to wear throughout the day. A pulse oximeter which is clipped to a person's finger or toe gives accurate measurements about a user's heart rate, but it is not feasible to constantly wear it on a daily basis. Therefore, a middle ground between comfort and accuracy has to be found to automatically measure a user's stress level in their daily lives. To accomplish this, a Garmin^{TM} wearable watch is used for this study.

2.2.1 Company and products

The Garmin company focuses its efforts on developing Technology for GPS navigation and wearable technology for automotive, aviation, marine, outdoor, and fitness applications. They develop various smart wristwatches which provide various features to monitor a user's health and fitness data.

Their different smartwatch models may have varying sensors and features, but most of

them incorporate a heart rate sensor, which is used to measure not only the pulse of a person but also their heart rate variability, to measure a user's stress level. According to Garmin, "The variable length of time in between each heartbeat is regulated by the body's autonomic nervous system." and "The less variability between beats equals higher stress levels, whereas the increase in variability indicates less stress.". Furthermore, the wearables determine the duration of a user's sleep phases, count a user's steps and activity, and use - depending on the model - different types of displays to show the user various information about their health and their phone.

The smartwatches connect via Bluetooth to a smartphone via the Garmin Connect™ application, which will constantly fetch data from the user's watch and synchronize them with Garmin's online database. The mobile application can also be used to gain more detailed information about a user's heart rate and stress throughout the day, as well as daily step count, workout information, and many other fitness-related information.

For this thesis, most users were given a Garmin vívosmart® 4 to track their biometric data. One user used their personal Garmin device, which was a different model. The devices were used in the user trials to measure the stress level of a user to grasp their context at a certain point in time, as well as measure their step count to get an understanding of their physical activities. This information is used to figure out under which conditions a notification should be sent to the user, as well as to decide which kind of content should be sent with a notification [33]–[36].

2.2.2 Garmin API

Features and downsides

For this paper, a backend application was developed that continuously checks a user's context and sends out notifications based on a user's context. To get information about a user's stress and physical activity state, the Garmin Developer API is used to get information from a user's Garmin Device.
While multiple different APIs are provided, this project uses the Health API to get daily reports of the heart rate, steps, and stress levels of a user. The API provides both daily summaries including stress levels, as well as shorter reports every 15 minutes, including a user's physical activity. Unfortunately, the shorter reports which are updated every quarter of an hour do not include information about the current stress level of a user, and even if they did, there would only be the stress level of 15 minutes ago in the worst-case, which might not be accurate enough to decide on whether or not a user is stressed this very moment. The daily reports on the other hand, which are sent at midnight every day, include accurate stress information throughout a user's day. While testing the Garmin Connect application with the test person in the field trials, it was noticeable that the Garmin Connect app did not always synchronize daily reports on its own. Most of the time daily reports reached the database of the backend at the right time, but sometimes

users had to be asked to manually press the sync button in their Garmin Connect app. Fortunately, this saved all daily reports missing from the database to the database [37].

Usage for this project

Ideally, the backend of the software prototype built for this thesis would have live information about a user's stress level, to make a real-time decision on whether or not a user should be provided with a notification. While Garmin provides high-quality products with good heart rate measurements and an included pedometer, it is unfortunately currently not possible to programmatically gather live information about a user's stress level, due to above mentioned API restrictions. Therefore, the software prototype built for this Master thesis will not be able to access the Garmin data of a user in a real-time fashion to decide on the user's context. The prototype will, however, be able to figure out under which stress and physical activity conditions a message was sent in retrospect. This way the correlation between a user's stress level measured by a Garmin device and their likeliness to answer push notifications can be researched. Even though live data is not available, fitness and stress reports of previous days can be accessed at any time, and are used to make a decision about which message should be sent.

CHAPTER 3

State of the Art

Several researchers have already explored the topic of timing notifications properly.

3.1 Measuring the context

As already stated previously *context* of a user is the current situation a user is in. This could be the place a user is currently at, the mood of a user, the current weather, or if the user is currently stressed out. Ideally, everything that could influence a user's likeliness to answer a notification would be taken as a variable to measure the context of a user. In reality, it is of course impossible to consider and especially measure all factors which make up a user's context. Therefore, different projects that aim to find perfect conditions for JITAIs have to settle on a specific set of variables which are relevant to the type of interventions that should be sent out. Not only should the variables be relevant for the application, but they must be variables which can be properly measured as well. There are many different variables to include or exclude when measuring the context of a user, varying between studies and applications. In the following sections, a few commonly used parameters for measuring the context will be described.

3.1.1 Motion and Location

In a lot of cases, information about the user's context can be inferred by using the sensors of a smartphone. The accelerometer, gyroscope, and GPS module of a mobile phone can be used to gather information about the whereabouts of a user, as well as the movement status of the user (so whether a user is currently walking, sitting, driving a car, or otherwise). Both major phone operating system providers provide their own, OS-native solution for this, with Google's *Activity Recognition API* and Apple's *Core*

Motion library. These APIs enable almost every phone to easily track the movement status [38], [39].

There are also less direct ways of gathering information about a user's motion and location state. A paper by Visuri et al. investigates interaction with smartphone notifications and content while examining the user's context. While GPS is used as relevant sensor data, Visuri et al. also use the battery charging state and the Wi-Fi Network the phone is currently connected to as means to classify the current context of a user. This information can also be used to indirectly gather information about a user's whereabouts. [30]

3.1.2 Time of day

Many people follow a daily routine. Therefore, they wake up, go to work, and go back to sleep at roughly the same times during a workday. The current time of the day can have a great effect on the likeliness of responding to notifications. As a study by Bidergaddi et al. suggests, workers in a typical 9 to 5 job environment are most likely to respond to JITIs during their lunch break on workdays, and most likely to respond in the afternoon on a day off. Furthermore, research suggests that users are more likely to respond during non-work hours. The study by Bidergaddi et al. explored the effectiveness of notifications at 6 different fixed points in time during the day. While the points in this study were the same for all participants, it is also an option to let users define their daily routine schedules, as seen in a study by Morrison et al. This way, notification timings can be individualized to be sent out at time slots which might occur during different times of the day for different users but still have the same meaning to them. [28], [40]

3.1.3 Stress

Several JITI systems exist to reduce stress, but many also consider stress as an important parameter. Stress is more complex to measure than other parameters.
One way of measuring a user's stress level is by using questionnaires. This was done in a study by Leech et al. where the goal was to reduce the stress of caretakers using a smartphone application. The study used a questionnaire developed by Cohen et al. to figure out how stressful people perceived a certain situation. It uses several Likert-Scale questions to retroactively figure out how taxing a situation was perceived by a person. This method can only measure perceived stress, which might not always correlate with the actual stress level of a person. Furthermore, it can not provide live stress data and can be quite time-consuming to fill out several times.
Alternatively, stress can also be measured objectively. To accomplish this, specialized equipment is used for brain activity, heart rate variability, pupil dilation, or even keystroke and mouse dynamics. Many other factors which can be objectively measured exist, as described by Goyal et al. The downside to objectively measuring stress is that it can not be done by just using a smartphone and special equipment is needed. [41]–[43]

After work on this master thesis started, a meta-paper was released in 2023 [44], investigating several studies using common fitness trackers with the goal of stress reduction through the means of interventions. For the paper, over 40 studies aiming to reduce stress while tracking stress through a sensor were collected. One of the systems tracks different aspects of a user's stress level and then suggests one of three coping techniques[45]. Another one aimed at improving stress using yoga techniques [46]. One investigated the interplay between stress and physical activity [47].

Due to the rise in AI technology during the writing of this master thesis, several new papers investigating and detecting stress with the help of artificial intelligence and machine learning technology have been published. Approaches for this include stress detection with wearables and deep learning [48], classification of Electroencephalogram data [49], [50], mouse, keyboard and heart rate variability[51] or cortisol measurements [52] using a support vector machine or other classification strategies such as k-NN and Random Forest approaches. Other studys determine stress using AI for Facial recognition [53] or natural language processing [54].

3.1.4 Other Parameters

To evaluate the context of a user with a smartphone, many different aspects and parameters can be considered. While many studies use the location and motion of a user to decide on the context a user is currently in, there are also other, less often-used parameters. As seen in a study by Visuri et al., it can also be evaluated whether or not a user has currently headphones connected to their smartphone to settle on the context. Furthermore, the current ringer mode settings (silent, vibration, normal) of a smartphone can be taken into account. The screen state (off, on, locked, unlocked) of a user's smartphone can also be used as an indicator for a user's context, as well as the app which is currently in the foreground [30]. Other studies use the current weather [55], including the temperature and precipitation [56] to assess a user's context.

3.2 Current Systems

Over the past decade, several systems optimized the timing and content of just-in-time interventions. Different systems for notification timing were developed for different use cases, such as learning apps, fitness apps, or apps to improve the health state of their users. All of those apps vary in their implementation of the system. They all pursue different strategies on which content should be provided, how the context of a user is determined, what interactions should be triggered, how the application is designed, and how notifications should be scheduled to improve the response rate of a user. Furthermore, the way the applications were tested (which people were selected as test persons, amount of test persons, duration of the test, etc.) and when a notification is measured as "accepted" differ between studies.

This chapter will give an overview of several studies from the past years, what methodology they used, and how they differ from one another.

3.2.1 Using Notifications versus not using Notifications - The "JOOL" app

The Application

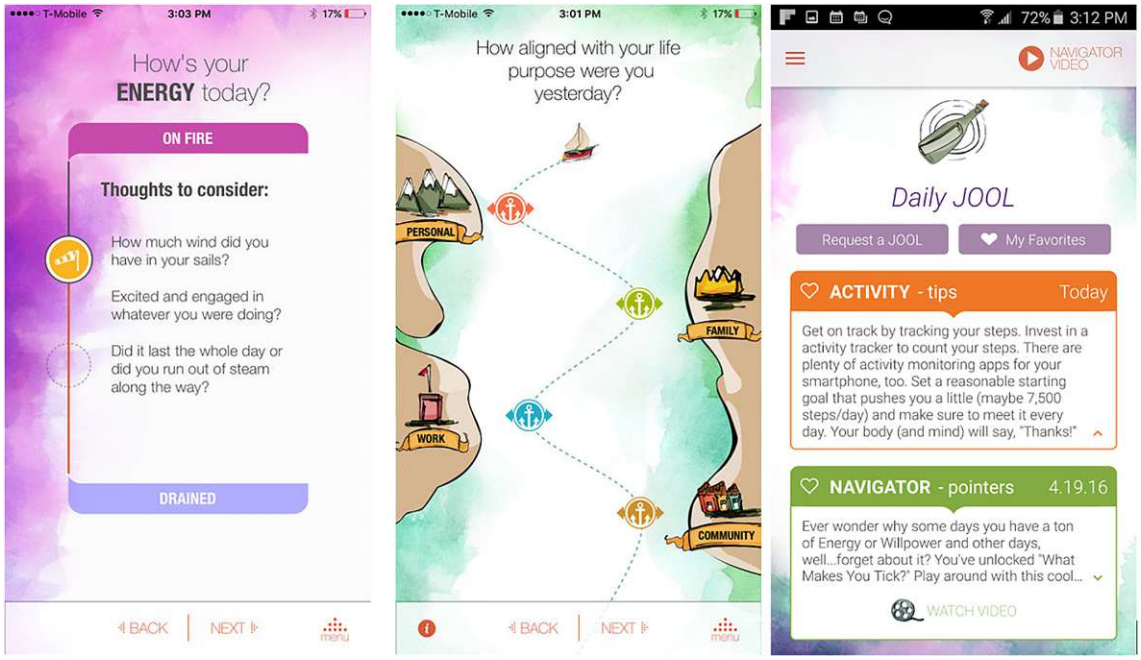


Figure 3.1: Screenshots of the JOOL application

"The JOOL app is a smartphone-based behavioral intervention using self-monitoring and feedback strategies to help users find their purpose in life and develop the energy and willpower they need to live in accordance with their purpose every day." The application uses push notifications with the aim of improving user engagement with the app. The notifications are not picked completely random - their content is dependent on the situation of a user. They evaluate the situation a user is in (e.g. the user's **"Context"**). "The context is determined by the user's current and past data from the self-monitoring, other app usage, and environmental data such as the time of day and day of week." Based on the setting of a user, a set of qualifying messages is generated, and one message is sent to the user as a notification [40].

Study

In 2018, a study which used the JOOL app was conducted by Bidargaddi et al. It explores the effects of sending a push notification with contextually tailored content

versus not sending a message at all. The study tries to assess which of those two strategies improves user engagement with the app more. It also explores if there is a difference in the effectiveness of the two strategies on weekends or weekdays.
For this study, 1255 users used the intervention app over the course of 89 days. Every day, the system randomly selected one of six different, predefined times of the day. Users were split into two groups, one that would receive a push notification with individualized message contents at the selected point in time and one that would not receive any notification at all. After that, it was examined which user group would be more likely to engage in the application within the next 24 hours.
Their study finds that users who did receive a message were 3.9% more likely to engage in the application in the next 24 hours than people who did not. The paper states that messages did seem to have a higher probability to improve user engagement on weekends (users were 8.7% more likely to engage with the app) as opposed to weekdays (only 2.5% of users were), but this observed effect was not statistically significant. Another conclusion this paper came to - which is also consistent with a lot of other research - is that on average, midday is the best time for sending a notification message. Users were 11.8% more likely to engage with the application after they received a message during the weekends at 12:30 pm. [40]

3.2.2 Evaluating notification intervals and content - “English Practice“

The Application

In 2016, Pham et al. conducted a study where notifications were used with the goal of improving user engagement with the Android application “English Practice“. As the name suggests, the application is used to practice the English language. Besides several grammar lessons and quiz questions, the application also includes a daily quiz, and “social media“-like features, like a chat room to practice English skills with other people. The application seemed to suffer from the issue that many users would install the app, open it once, but never open it ever again. To combat this, the developers tried to use push notifications to get users to open the application more frequently and improve their English learning process. To assess the effectiveness of this measurement, a study regarding the effectiveness of the newly implemented push notifications was conducted [31].

Study

Since the application was available to the public on the Google Play Store, the behavior of 12865 actual users was analyzed. While the study did not attempt to individualize message timings based on a user’s context, it gives useful insight about the frequency at which messages should be sent. The test persons were split into a control group of people not receiving any notifications, and an experimental group, which was again divided into 4 subgroups. Each member of those 4 subgroups received notifications in different

Figure 3.2: Main menu and different notifications of the English practice app

time intervals. The 4 time intervals were 3 hours, 12 hours, 1 day, or 2 days between notifications. It was then evaluated if the "app retention", therefore the number of people who keep actively using the app was higher for the experimental groups compared to the control group

The conclusion was that the notifications help to motivate users to keep using the app. In the experimental group, about 29% of the users were still actively using the application after one week, while only 24% were still using the app in the control group. But the study also suggests that if a too-high frequency is picked, worse results than in the group that did not receive any messages at all can be received. In the group where people did receive a notification every 3 hours, only 42% kept using the app after one day, whereas 44% did in the control group. The study also found that there were great differences between the effectiveness of different messages with different contents. Some messages are much more likely to get a user to tap on them than others. As previously mentioned in 2.1.4, social notifications are much more effective in getting the user to click on them

than others. This discovery also applies to this study, as notifications for social features like “New message“ or “New comment“ had a click rate of 39-50%, while simple reminder messages only had a conversion rate of 14-17%. [31]

3.2.3 Timing and Frequency of Health Interventions - “Healthy mind“

The Application

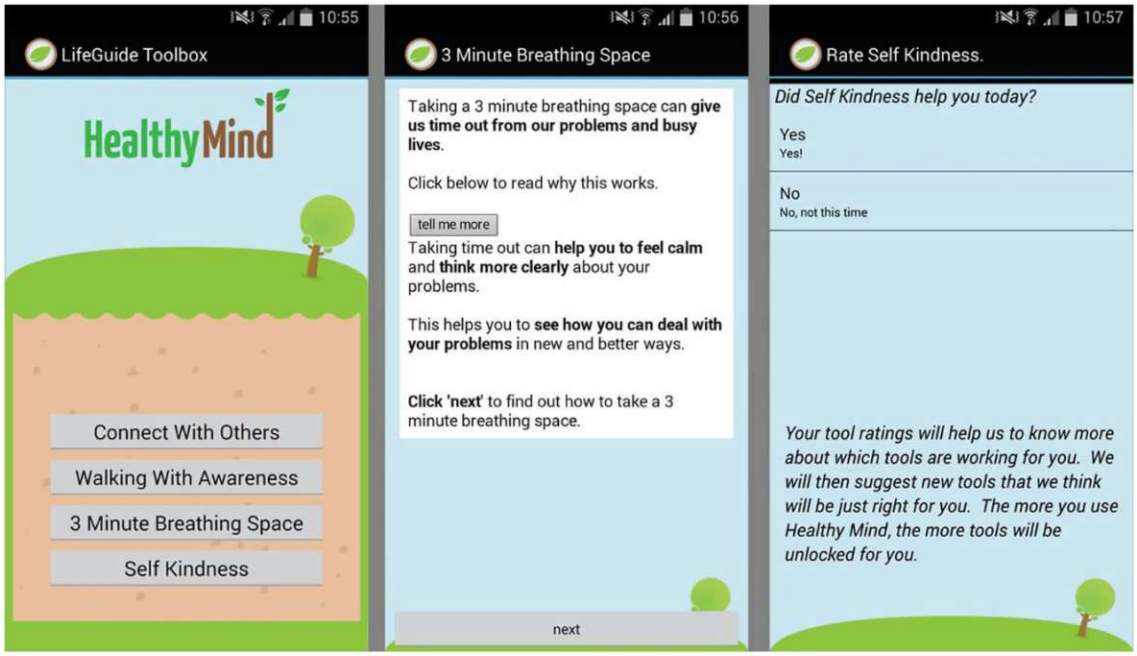


Figure 3.3: Healthy Mind tool menu-, description- and rating-screen

“Healthy mind“ is “an Android app-based stress management intervention disseminated in a UK-based public health setting“. It offers “evidence-based tools for managing stress and other negative emotions“. It aims to help with mindfulness, breathing exercises, and stress management. The application offered different features (called “tools“) to help the user combat stress. Not all of them are available from the start but can be unlocked. The notifications have different types, such as “tool announcements“, “tool suggestions“ and “general reminders“. “Tool announcements“ and “tool suggestions“ notify a user when a new tool is unlocked, or try to motivate a user to use a specific tool. “General reminders“ just try to remind the user to use the app at all. In a 2016 study conducted by Morrison et al., the app was used as a basis to research the effects of different notification scheduling strategies. [28]

Study

For this study, 3 different versions of the application were built. One with intelligent notifications, one with daily notifications, and another one within pre-defined time frames. For all three versions, the system automatically recorded the usage of the app and which notifications were answered. Additionally, telephone interviews with the participants were used to gain qualitative data. For the experiment, 162 test persons participated, but only 77 of them provided usable data due to a software error. Those were split into 3 groups for each version of the app respectively. To determine the context of a user, three parameters provided by the sensors of the phone were used, namely the *location*, the *time of day* determined by the phone's clock, and the current *movement status* as measured by the phone's accelerometer.
Users who were in the "intelligent notifications" group received notifications at times when the system predicted that the user would be likely to respond. While the system started out with two random notifications, it then adapted the notification timings based on the context of a user. This was done with a simple machine learning tool, the "Naive Bayesian classifier", which linked different contexts to a likelihood that a certain user will respond. After that, the system checked the user's context every 20 minutes to decide if it was an appropriate moment for a notification message. The weights of the different variables varied between users, based on their previous answers to notifications. Up to 3 messages were sent per day.
As previously mentioned, perceived repetitiveness can also dampen the effect of notifications. To avoid this, the system made sure that two consecutive notifications did not promote the same tool. The experiment collected data for 2 weeks for every user.
In this particular study, the intelligent scheduling of messages did not seem to provide any advantages over pre-determined notification delivery, at least when it comes to timing. The paper admits that this contradicts other papers, which did find individually tailored message timings to be better than static timings. The paper explains this by claiming other studies used artificial experimental settings, while their study used an application that was already used in a real work environment. Another observation by the paper is that if the system is allowed to send out too many "intelligent" messages, those will be perceived as random. The paper admits that it evaluated one specific implementation of tailored messages, and further testing has to be done if "intelligent" messages can improve message timings. [28]

3.2.4 Breathing Interventions while driving - "AmbientBreath"

The Application

While many just-in-time intervention applications are built for mobile phones, a quite different approach was taken by Lee et al. Their system "AmbientBreath" focuses on reducing stress during driving. According to their paper, driving a car can lead to many stressful scenarios, like congestion or unexpected behavior of other drivers. If short-term stress from stressful driving situations on the daily commute stacks up, it can lead to chronic stress, which increases the risk of "cardiovascular problems, gastrointestinal and

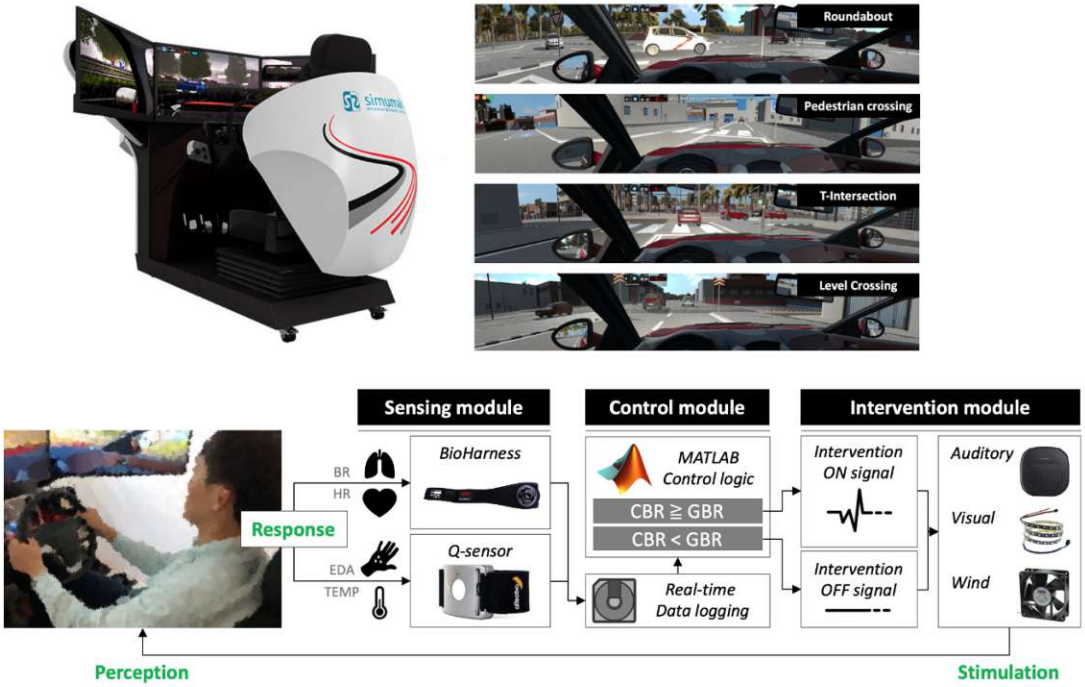


Figure 3.4: Driving simulator and system setup

musculoskeletal disorders, and post-traumatic stress“. Furthermore, a stressed-out driver is more prone to causing crashes. Therefore a just-in-time system for reducing stress was developed. An intervention system that attempts to remind the driver of something while actually driving a car faces the problem that the driver might be distracted by the intervention, which inherits a serious security risk. Generally, if a person is recognized as “driving“ by a mobile phone just-in-time intervention system, this is often seen as a KO criterion for sending a message, as this might distract the user, and even if it does not, the user is most likely busy with driving the car, and will not respond to the message. To get drivers to breathe more slowly and consciously, “subtle modulatory signals of auditory, visual, and wind (tactile + thermoceptive) stimuli“ were used, “with the goal of lowering the driver's breathing rate while providing minimal distraction“. The system was used in combination with a driving simulator. [57]

Study

Using the system, two user studies with a combined amount of 54 users were conducted in 2021. The participants were first introduced to three different ambient feedback methods already present in normal cars to help them normalize their breathing: rhythmic background noise, synchronized modulation of wind via dashboard fans, or ambient lights combined with auditory signals. They could then pick a preferred method or a

combination of them. The most popular choice was a combination of wind and audio signals. The study showed that these measurements could help improve the breathing rate, but whether or not this was successful depended on the driving context.
A specialty of this study was that, while in other studies using mobile phones the context of a user can not be predicted, this study happened in a driving simulator and gave the researchers full control over the situation a user will be in. "Timing" in this study mostly refers to what kind of driving situation a user was currently in. The study was performed for both manual and autonomous driving. When the breathing rates of the user group who were using the breathing guide system and the control group were compared, the results were mostly the same. However, the study did find that users were more engaged in breathing exercises when being at a traffic light. This effect was even more extreme when participants were given breathing exercise training beforehand. While the sample size of this experiment was rather small, the results point toward the conclusion that user engagement of a just-in-time breathing exercise while driving can be effective under some driving conditions and not be useful at all under others. [57]

3.2.5 Comparisons

The following table details the features of the previously mentioned intervention systems.

Table 3.1: State-of-the-art comparisons

Name	**Year**	**Goal**	**Hardware**	**Intervention Method**	**Strategy**
JOOL	2018	Mindfulness and finding purpose in life	Smartphone app	Push Notifications	Time-slots
English Practice	2016	Keep users engaged with language learning app	Android app	Push Notifications	Randomize notification frequencies
Healthy Mind	2017	Mindfulness, breathing exercises, stress management	Android app	Push Notifications	Random timings, machine learning
Ambient Breath	2020	Reducing stress during driving	Driving simulator	Visual signals, audio signals, wind	Remind driver to lower breathing rate

Compared to the applications mentioned in this chapter, the application developed for this thesis focuses on sending tips for physical activity and stress reduction to its users. Like many other behavior change applications, it uses iPhone and Android mobile devices

to gather information about a user's context and to send notifications to them. In addition, a Garmin wearable device is used to get additional parameters of the user's context. As a timing strategy, a mixture of the previously mentioned time-slot, random frequency, and random timings approach is used, as well as a simple machine learning approach.

CHAPTER 4

Design and Planning of Implementation

For this thesis, an adaptive notification system has been developed to handle individual user behaviors. The goal of this system is to send health-related tips and tasks out to users and to send out notifications at times that are convenient for the user.

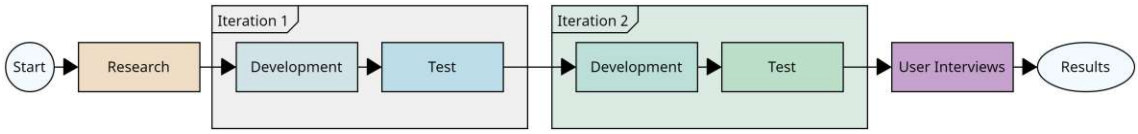


Figure 4.1: Flowchart of the Development Strategy

4.1 Design of the System

4.1.1 Development Strategy

As previously mentioned in 1.4, the strategy of developing and testing the application consists of a research phase, followed by two iterations of a development and testing phase, ending with a single phase of user interviews. In this section, the individual phases are described in further detail.

Research

In the research phase of the development process, the current state of the art of other notification scheduling and smart intervention systems was assessed. Using the information from other recent papers, a set of context variables to measure the user's current

situation was picked. Furthermore, the research was used as a foundation for scheduling mechanics and strategies used in the created software system.

Iteration 1

Development The goal of the development of the first iteration was to get an initial proof-of-concept version running, meaning that all components should already be present, including the database, backend, phone application, and Garmin wearable support. Furthermore, the scheduling algorithm and sending of the notifications were already meant to work in some form or capacity

Test During the testing phase of the first iteration, information was gathered to see how well the application works and how to improve its effectiveness and develop it further.

Iteration 2

Development Using the information gathered in iteration 1, the goal of the second development phase was to finalize the prototype using the information gathered in the first test iteration. This includes fine-tuning the scheduling algorithm and user variables, as well as refining the user experience. Another goal was to get the mobile app on both Android and iOS.

Test During the testing phase of the second iteration, the goal was to gather information about the effectiveness of the scheduling system. After performing tests with another test group, a statistic about which conditions lead to the most user responses was created.

User Interview

Users were interviewed about their test results. Users were shown their statistical tendencies to respond under certain conditions and asked to reason why they thought their response rate looked like it did. This is done to contextualize the gathered information. Furthermore, users were asked if they felt like their response rate improved if they felt like messages reached them at more convenient times, and what their general experience was.

4.1.2 Technical Choices

The system used for this thesis consists of a backend - which itself is divided into several micro-services - and a front-end application for smartphones, to receive notifications and send answers back to the backend. The application is available for both Android and Apple's iOS. Additionally, a Garmin smartwatch was used by each user to track the user's health and stress data during usage of the application. The wearable connects to the user's smartphone via Bluetooth and transmits health data to the Garmin Connect app. The app then sends the data to the Garmin Servers, which then proceed to mirror

the data onto the servers hosted for this project.
In the following, the chosen technologies for the backend, frontend, and wearables will be mentioned, together with the reasons why they were picked, and with a quick statement on whether or not they fulfilled their expectations.

Backend

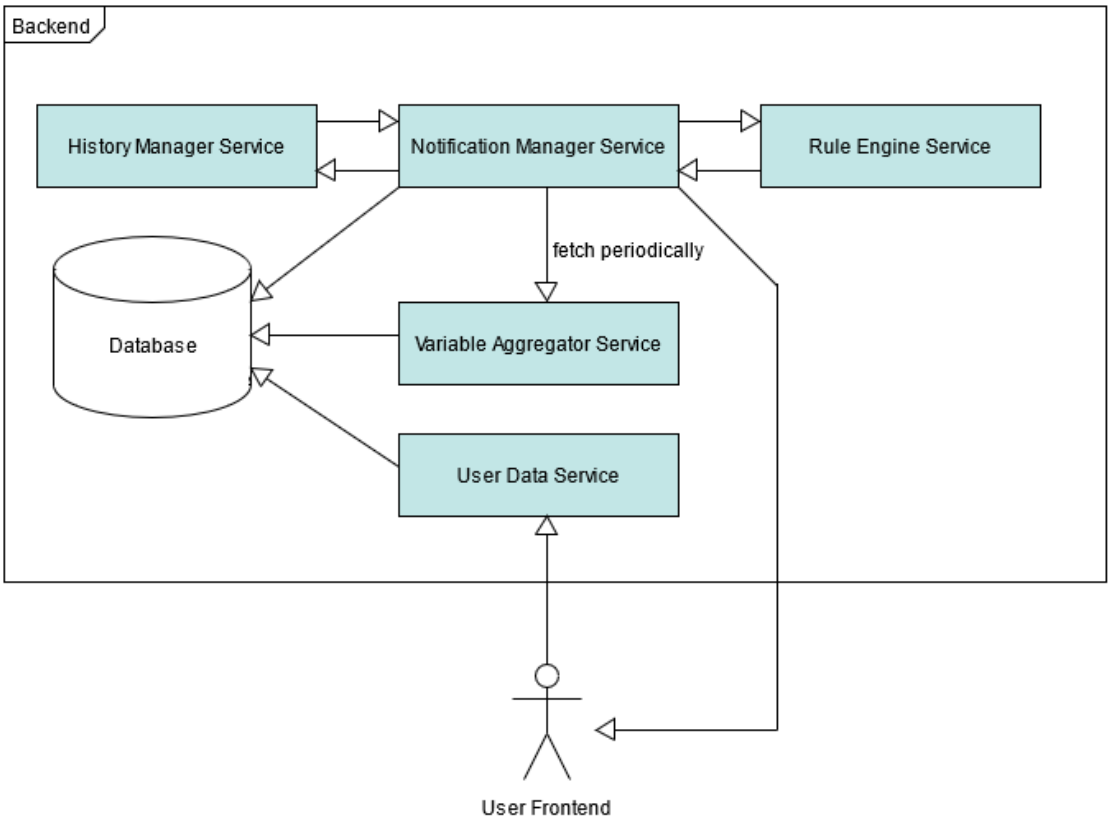


Figure 4.2: UML Diagram of the Backend

As seen in 4.2 The backend consists of 5 micro-services. All five of them use *NodeJS* and are written in *TypeScript*. For communication between the services, the popular NodeJS libraries *Express* and *Axios* are used. Only the *User Data Service's* REST interface is exposed to the internet and is accessed by the frontend applications of the user.

Frontend

For developing the fronted, *Flutter* was used, in combination with the *Dart* programming language. The reason why Flutter was chosen, is to have one common code base for both the Android and the iOS application. Unfortunately, many of the features of the

simplistic front-end application were platform-specific, such as receiving and displaying notifications, or running in the background and constantly tracking the user's movements.

Push notifications use quite different systems for Android and iOS, and always require work done to the code specific to one platform to make it work. While the functionality for tracking the movements and GPS coordinates of smartphones is provided by different APIs, there are flutter libraries that attempt to unify the interfaces and provide a common interface for both systems. One such library is Flutter Background Geolocation [58], which was used for this project. While it attempts to unite both interfaces, there are always some concepts that only exist on one platform. For example, if the application is closed, iOS will consistently reawaken your app after the user moves 200 meters from the location where the application was closed [59]. On Android, the reawakening of an application might not happen consistently, and can also vary between different Android phones and their system and battery-saving settings. Therefore, while both the iOS and the Android versions mostly use the same flutter code for location tracking, their exact behavior still deviates a bit when in use.
Since many of the Flutter application features are platform-specific, the benefits of using flutter mostly come down to using the same user interface for both platforms.

Wearable

Figure 4.3: Garmin vívosmart® 4 [34]

To assess the user's context (which will be elaborated on in 4.1.3), a wearable Garmin watch is used. As already discussed in 2.2, it provides a variety of different sensors to track a user's stress and fitness data. The reason Garmin Smartwatches were chosen

over other available products is their general good reputation and the previous positive experience of the Austrian Institute of Technology with their products. Furthermore, the Austrian Institute of Technology already had servers and databases set up to use the Garmin API, making further development with the API easier.
The tracked parameters and the tracking itself seemed good, the smartwatches were easy to use, and also the integration of the Garmin API was made easy as a database was already provided. However, the issue with using Garmin wearables in this case is that - while Garmin provides some real-time data to users through the Garmin app - the API offers most tracked variables only in a daily report generated at midnight, after each day. Some variables are provided in 15-minute intervals, but some - for example, stress measurements - can only be looked at in retrospect when the daily report is available, and not properly in real-time.

4.1.3 User context

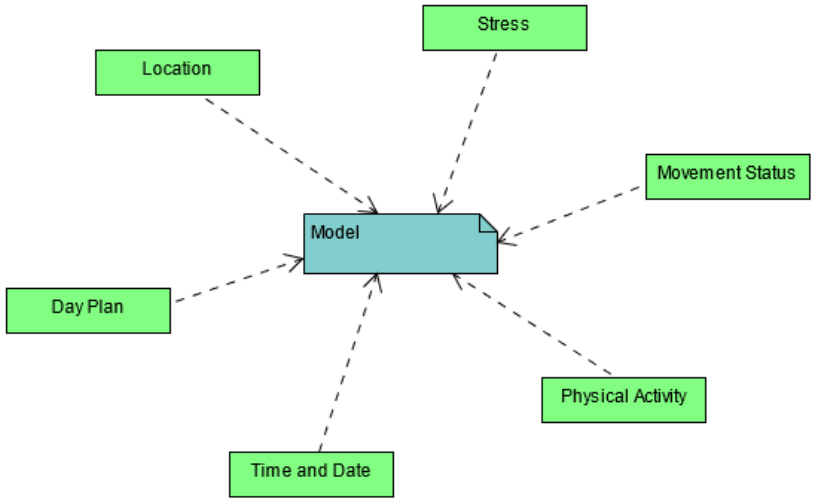


Figure 4.4: Context Variables [34]

Before the development of the backend started, a set of variables that will be tracked to assess the user's context was chosen. This was done based on the conducted state-of-the-art literature research. With this in mind, the model is based on six variables as seen in 4.1.3. During the literature research, location was a parameter commonly included in many studies [28], [55], [60], [61]. The same can be said for time as a parameter [28], [40], [55], [61]. Physical activity and motion status of a user were used in some studies [61], [62]. A major motivating factor to include physical activity and stress parameters is to explore the impact and usefulness of the information provided by the Garmin sensors.

4.2 Development and improvements

4.2.1 Initial Situation and Pre-Existing Infrastructure

This project was created in close cooperation with the Austrian Institute of Technology. Thankfully, the AIT granted access to pre-existing database infrastructure, as well as servers to host the backend of the system.

Garmin access

Furthermore, the AIT made it easier to access the Garmin API. First of all, Garmin does not give everyone access to their API, and it is not intended to be used by hobbyists or students. To use their API, access has to be requested via Garmin's "Developer Program Access Request Form", which is explicitly directed towards companies and not individuals, as it asks for the company name and intended use for the API [37].
Fortunately, the AIT already had access to the API, as well as several Garmin devices to distribute the potential test persons. Besides the sole permission to use the API, the AIT also already had a fully functional setup to use the Garmin API with. Unlike traditional APIs, where there is usually some REST backend provided by the API Provider, where any information can be accessed by a simple GET request, the Garmin API works the other way around. If an API user wants information about their Garmin Clients, the intended way to get that information is to host an API endpoint that can *receive* reports by Garmin. Garmin will send reports only once, if the receiving backend is available, it gets a chance to persist the report, otherwise, it will be lost to time. Therefore, an API user needs to keep such a receiving backend up and running 24/7 to receive all Garmin reports.
Theoretically, Garmin also provides a separate REST endpoint to get some user information via traditional GET requests, but the reports achieved this way are very limited compared to the other method, and do not include everything needed for this project. Even worse, this API only allows a limited amount of requests per user in a certain timeframe. Garmin is vague about how many requests can be performed before a user has to wait again. Considering these two points, the REST endpoint by Garmin was not an option for this project.
Luckily, the Austrian Institute of Technology already had a working server to receive Garmin requests, and also a web interface where users could log in with their Garmin account and then connect their Garmin account with their account for the notification application for this project. Upon login, the user receives an identifier which represents their Garmin account in the AIT database. This identification can be provided when setting up the notification application so that a user's notifications can be linked to data from a certain Garmin account.

Firebase access

As it was clear that the application would center a lot of its functionality around push notifications, there was the need to have a system for push notifications, which works on both iOS and Android. Fortunately, there were already other applications within the Austrian Institute of Technology that used Firebase Cloud Messaging as a solution for push notifications. Firebase gives developers convenient management for registering devices for receiving push notifications and offers an easy-to-use interface. Also, the interface acts as a proper abstraction layer so that backend developers do not have to worry about the target smartphone model number, operating system, or state (turned on or off, silent...). Furthermore, Firebase will take care of the delivery of the notifications, even if the notification cannot be delivered right away (for example if the user's smartphone is currently turned off or has no internet connection, a message will be delivered at a later time). This way robust notification delivery is established without too much work from the developer side.
Firebase offers many services besides Cloud Messaging. There is a free version, and a paid version offering more features, such as higher storage amounts for realtime databases and cloud storage. While Firebase Cloud Messaging (FCM) is generally free to use, access was provided by the Austrian Institute of Technology. [63]

Apple ecosystem and iOS developer access

Anyone can develop an Android application and distribute it privately to any smartphone which allows installing APKs without paying any licensing fees to Google, as all tools needed are openly and freely available. This is usually enough for creating applications for scientific experiments, but even if there is a reason to make an application widely available on the Google Play store, this can be easily done by paying a small one-time fee to Google.
Unfortunately, this is not as easy when publishing applications on the Apple App Store. Apple tries to operate a much more closed ecosystem compared to Google. Even testing an application on a developer's phone is rather restrictive, as Apple presumably wants to avoid users installing applications from any other sources as their app store.
First of all, developing iOS applications is not readily available for everyone, as development does require a Macintosh personal computer. While it is theoretically possible to write the code for both iOS and Android applications on a Windows, Linux, or Mac system, only Android applications can be compiled on all of them. When developing on iOS, Apple's "Xcode" is needed to compile, test, or publish any application. As XCode is only available for Mac, this greatly limits the access to iOS development tools, as Macs only make up approximately 15% of the global personal computers in 2022.
Thankfully, a Mac Computer was provided by the Austrian Institute of Technology to enable the development of the application for iOS devices. With a Mac and a free Apple User account, a Developer can start building Apps, and deploy them to their own iPhone for personal testing. But this does still come with a lot of restrictions. Applications "expire" after 7 seven days, and have to be rebuilt and redeployed to the testing device,

so a longer test run for 2 weeks is impossible without a paid developer account. Also, another missing feature that makes a test run with multiple persons infeasible is the ability to use Apple's "TestFlight", which is used to distribute the application as a beta test to iPhones of test persons. TestFlight is only available for users with a paid Apple Developer Account. Every app that could be sent out to a test person must be approved by Apple first, and this process must be repeated every time a user publishes an update. Not only does this force the developers of an application that is used for a small testing group to pay a licensing fee of 99$ a year to Apple, it also greatly limits the flexibility of the development process as updates and hotfixes can be held back for up to 24 hours until they are approved, and might be disapproved by Apple entirely. Fortunately, an Apple Developer Membership was already present at the Austrian Institute of Technology, and no extra membership was needed to be paid for this project [64]–[67].

Database technology

Considering this initial situation for access to Garmin devices and potential server hosting possibilities, it was clear that the existing Garmin server infrastructure would be used. The backend software was designed around the idea that it would be used with AIT infrastructure. Therefore the decision was made to use PostgreSQL for the backend, as this was the database technology already used for the existing user database.

4.2.2 Requirements

While there were no restrictions given by the Austrian Institute of Technology for developing this system, there were a few requirements given. The application should be available for both Android and iOS. Initially, the Idea was to implement the Android version inside of an already existing application for AIT. But soon after work on the project began, this requirement was lifted. Instead, a Flutter version for both iOS and Android was developed. Another requirement that was not lifted was that the application had to use Garmin smartwatches,

4.2.3 Planning Phase

Application Goals

When the planning phase of the application started, the idea was to improve users' stress levels and improve physical activity for users. While this remained true throughout the development process, the idea was much more concretized after several iteration steps. The application is supposed to send out prompts to users to improve their physical activity and reduce their stress. Furthermore, users would receive prompts at a certain point in time, where that specific prompt would improve their stress situation and their physical activity. As simultaneously reducing physical activity, improving stress rates, tailoring content, and timing messages were deemed to be rather imprecise goals for this thesis, which would also be quite hard to track the results of all 4 properly, the decision

was made to focus the project and its evaluation on the timing aspect. The application would still send out prompts encouraging sports and stress-coping techniques, and every possible predefined prompt would have a defined rule set under which situations they could appear, the set goal of the application would be to improve the timings of the messages, about the user's context.

First Prompt Designs

After the goal was defined, the first thing that was designed was the actual text prompts that users would receive during their day. Inspired by other papers, one idea was to encourage participants to intercept prolonged sedentary behavior by giving users tasks that were not initially apparent to fulfill this purpose. For example "stand up and take a glass of water". After several iterations, this idea was simplified to a "Stand up and walk around for a bit" task.
Also initially included in the draft for the application were several tasks for doing specific exercises at home, some of which would have even required certain exercising equipment, like a rowing machine or bicycle. The idea would have been to either give users the option to state "I do not own this sports equipment" if they received such a prompt or to ask them upon registration which ways of exercising are possible for them. After several iterations of the project plan, the application was much more streamlined to focus on one thing: the timing of the notifications. While prompts would still be tailored to a user's context, the situations in which a certain notification could pop up would be hard coded in the backend, and not customized for each individual. The purpose and focus of the application after the planning phase was to send users prompts randomly at first and to later only send them at times, when the backend considered prompts to be successful, due to probabilities based on the initial phase. The goal of the application was to get better response rates from users from deliberately sent messages, than from randomly timed prompts.
As initially mentioned in 4.1.3, in the beginning, a set of variables to asses the user's context was selected. The selection was based off the state-of-the-art research (for example, the prompts for combating prolonged sedentary behavior by Müller et al. [68]). The contents of the notifications were based off existing behaviour change projects of the Austrian Institute of Technology.
After the initial design phase, there were two major categories of prompts. Prompts that attempt to motivate a user to improve their physical activity, or at least to motivate them. On the other hand, stress reduction notifications, which either try to ask a user to take a few minutes off their current task in order to perform a breathing exercise or just take time to relax, or simply inform them about some benefits certain good habits can have on their mental health and stress levels.
In this draft for the notification messages (which would eventually be used for the first test run of the system), both tasks and tips were sent to the user. The idea was to assign each notification an expected **length**. For example, going for a walk would be considered

a *long* task, while just reading a notification message including some information about mental health would be considered *short*. Using this information, the system would pick a message that was suited for the situation a user was currently in. While standing up and going for an hour-long bicycle ride will probably not be something a user would be capable of midway through a workday, it might be a feasible task during the final hours of a day when the user is comfortable in their own home and still has time and energy for such a task. On the other hand, short text messages including information about the benefits of sports and stress management, might also be read during a lunch break, as this consumes much less time, and might even lead to a response of the user while they are occupied with another task. Using this system, tasks would be categorized into 4 different *"Task Lengths"*: Super short (up to 30 seconds), Short (up to 5 minutes), Medium (up to 15 Minutes), and Long (everything above 15 minutes).
Furthermore, notifications would have to be assigned a **location** they could appear in. For this, the 3 location states of the user context would be used. For every notification, a decision was made in which locations they would be appropriate. For many potential messages, this parameter would be set to *"Unimportant"*, and could therefore appear when a user was at home, at work, or at neither location. Some messages are only appropriate when the user is at home, such as "Listen to music, relax and perform a breathing exercise", as it would be quite inconvenient and socially deviant for a user to perform this task at work, even if the user currently had the time to do it at work. Also, tips on avoiding doing stressful tasks directly before going to sleep were adjusted to be only shown while the user was at *"home"*. While a user could most certainly read it while being at work, the just-in-time effect of that message would probably be greater if the user received the message at a place where they could go to sleep anytime soon after the message arrived.
Another parameter that was used to tailor the notification selection to the current situation of a user was to make use of the current **movement status** of a test person. There were three tiers of the movement status, ordered by their movement speed, starting with the slowest - *"Stationary"* - which occurs when a user is sitting or standing still. This category is followed by the self-explanatory category *"Walking"*, and the fastest category *"Running"*. In case the movement status is *"Unknown"* at the time when the backend wants to send a message to the user, all messages can be selected by the backend, regardless of their required movement status. If the movement status is known, every prompt has a minimal and a maximal movement status, and only if the user's current status is in between the minimum and the maximum, the message is considered for the pool of possible messages to send. The status *"In Vehicle"* is not considered for minimum and maximum boundaries for prompts, as users will not receive messages while their status shows that they are driving a vehicle.
To decide if a message should be picked from the stress management pool or the physical activity improvement pool should be picked at a certain moment, the system will compare the stress and physical activity reports of the last two days. Garmin does provide an in-house evaluation to see if stress is *"Rest"*, *"Low"*, *"Medium"* or *"High"* for a given day, which can also be displayed on a percentage scale. For physical activity, Garmin

provides a dynamic step goal for users, which can slightly vary between days. If the user constantly improves their daily step count, the step goal might increase, and lighten up if the daily step goal is not reached several times. Based on this, the system picks a *"Low"* (if the user's steps are way below the step goal), *"Medium"* (if the steps for a day are close to the step goal) and *"High"* (if the steps for a day are above the daily step goal.). If a prompt has to be sent out, only one of the two pools of messages will be selected, based on whether or not stress reduction or physical activity improvement is more urgent.

Table 4.1: Urgency values for stress level and physical activity

Physical Activity	
Level	Urgency
High	33
Medium	66
Low	99

Stress Level	
Level	Urgency
Rest	25
Low	50
Medium	75
High	100

As seen in Table 4.1, an *"Urgency"* value is assigned based on the context of physical activity and stress level values. A prompt will be selected for the category with a higher urgency (for example, if both states are medium, a stress reduction task will be sent out, as 75 is a higher value than 66).

Explanation for shorthands: Labels:
(Len. = Length, Loc. = Location, Min. Mov. = Minimal Movement Status, Max. Mov. = Maximal Movement Status, Min. Strs. = Minimal Stress Level, Max. Strs. = Maximum Stress Level, Min. Phys. = Minimal Physical Activity Level, Max. Phys. = Maximum Physical Activity Level)

Values:
Length: (SuSh = Super Short, Sh = Short, M = Medium, L = Long)

Location: (Un = Unimportant, Ho = Home, Wo = Work)

Movement Status: (St = Stationary, Wa = Walking, Ru = Running)

Physical Activity/Stress Levels: (R = Rest, Lo = Low, M = Medium, H = High)
For the first prototype iteration, the following tasks were used for Physical Activity Tasks:

Id	Task	Len.	Loc.	Min. Mov.	Max. Mov.	Min. Strs.	Max. Strs.	Min. Phys.	Max. Phys.
1	Task: remember to sit neatly and upright, do not curl the spine.	SuSh	Un	St	St	R	H	Lo	H
2	Task: stand up and stretch	SuSh	Un	St	St	R	M	Lo	H
3	Task: Stand up and walk a short lap (5 minutes or less)	Sh	Un	St	St	R	M	Lo	M

4	Task: Stand up and do 12 squats/12 push-ups or similar!	Sh	Un	St	St	R	Lo	Lo	Lo
5	Task: Take a short walk (15 minutes or less)	M	Ho	St	Wa	R	H	Lo	H
6	Task: Time for a short training session! Do 3 sets of 12 squats each	M	Ho	St	Wa	R	H	Lo	M
7	Task: Time for a short training session! Do 3 sets of 12 pushups each	M	Ho	St	Wa	R	H	Lo	M
8	Task: Take a long, extensive walk (longer than half an hour)	L	Ho	St	Wa	R	M	Lo	M
9	Task: Time for a round of running!	L	Ho	St	Wa	R	Lo	Lo	Lo
10	Task: Time for a round of cycling!	L	Ho	St	Wa	R	Lo	Lo	Lo
11	Tip: Experts believe that exercise releases chemicals in your brain that make you feel good. Try to make physical activity that you enjoy a part of your day!	SuSh	Un	St	Wa	R	M	Lo	Lo
12	Tip: Regular exercise can boost your self-esteem and help you feel better. Experts say that most people should exercise for about 30 minutes at least 5 days a week.	SuSh	Un	St	Wa	R	M	Lo	M
13	Tip: Exercise also keeps your brain and your other vital organs healthy. Exercise doesn't just mean playing sports or going to the gym. Walks in the park, gardening or housework can also keep you active.	SuSh	Un	St	Wa	R	M	Lo	Lo
14	Tip: Physical activity is helpful for a healthy heart and improving your joints and bones. But did you know that physical activity is also good for your mental health and well-being?	SuSh	Un	St	Wa	R	M	Lo	H
15	Tip: Exercise can boost our self-esteem. Self-esteem is how we perceive ourselves and our self-worth. It is an important indicator of our mental well-being and our ability to deal with stressors in life	SuSh	Un	St	Wa	R	M	M	H
16	Tip: Physical activity has great potential to increase our well-being. Just 10 minutes of brisk walking increases our mental alertness, energy and positive mood.	SuSh	Un	St	Wa	R	M	Lo	Lo
17	Tip: Regular physical activity can increase our self-esteem and reduce stress and anxiety. It also prevents mental health problems and increases quality of life for people with mental health problems.	SuSh	Un	St	Wa	R	M	Lo	H
18	Tip: Exercise can be very effective in reducing stress. Research on working adults has shown that highly active individuals tend to have lower stress levels compared to less active individuals.	SuSh	Un	St	Wa	R	M	Lo	M
19	Tip: Studies show that the risk of depression and dementia is about 20% to 30% lower in adults who engage in daily physical activity.	SuSh	Un	St	Wa	R	M	Lo	M
20	Tip: Avoid exercising too close to bedtime. Exercise gives you energy. But slow, relaxing activities like yoga can help you wind down before bed.	SuSh	Un	St	Wa	R	M	H	H
21	Tip: Stick to quiet activities just before bed, like reading.	SuSh	Ho	St	Wa	R	H	H	H
35	Tip: It's great that you're active! Physical activity has great potential to increase our well-being. Keep it up!	SuSh	Un	Ru	Ru	R	H	Lo	H

The following Tasks for Stress Reduction:

Id	Task	Len.	Loc.	Min. Mov.	Max. Mov.	Min. Strs.	Max. Strs.	Min. Phys.	Max. Phys.
22	Task: Short relaxation exercise. Briefly close your eyes and take a deep breath!	SuSh	Un	St	Wa	M	H	Lo	H
23	Task: Close your eyes. In the next minute, try to breathe in and out slowly 4-6 times!	SuSh	Un	St	St	M	H	Lo	H
24	Task: Take your time and pay attention to your breathing for the next 5 minutes. Relax. 4-6 breaths per minute are ideal!	Sh	Un	St	St	M	H	Lo	H
25	Task: Take your time and find slow, relaxing music. Relax and pay attention to your breathing for the next few minutes. 4-6 breaths per minute are ideal.	M	Ho	St	Wa	M	H	Lo	H
26	Tip: We are all only human. Sometimes we get tired or overwhelmed when we don't feel well or when things go wrong. When things get too much for you and you feel like you can't cope, ask for help	SuSh	Un	St	Wa	M	H	Lo	H
27	Tip: Variety is good for your mental health. Variety could be a five-minute break from cleaning your kitchen, a half-hour lunch break at work, or a weekend spent discovering something new!	SuSh	Un	St	Wa	M	H	Lo	H
28	Tip: A few minutes can be enough to de-stress you. A break, can mean doing nothing, but also exercise. Relax, try yoga or meditation, or just put your feet up!	M	Un	St	Wa	M	H	Lo	H
29	Tip: Exercise not only increases physical health, but also our self-esteem. Self-esteem is an important indicator of our psychological well-being and our ability to cope with stressors in life	SuSh	Un	St	Wa	R	M	M	H
30	Tip: Mindfulness helps with several conditions, including stress, depression, addictive behaviors such as alcohol or drug abuse and gambling, and physical problems such as high blood pressure, heart disease and chronic pain.	SuSh	Un	St	Wa	R	H	Lo	H
31	Tip: Follow a routine. Try to go to bed at the same time every night and wake up at the same time every morning, even on weekends!	SuSh	Un	St	Wa	R	H	Lo	H
32	Tip: If we don't give ourselves time breaks, stress can build up until we feel too overwhelmed to do anything.	SuSh	Un	St	Wa	H	H	Lo	H
33	Tip: Exercise can be very effective in reducing stress. Research on working adults has shown that highly active individuals tend to have lower stress levels compared to less active individuals.	SuSh	Un	St	Wa	R	M	Lo	M
34	Tip: Stick to quiet activities just before bed, like reading.	SuSh	Ho	St	Wa	H	H	Lo	H
36	Tip: Great that you are active! Exercise can be very effective in reducing stress, keep it up!	SuSh	Un	Ru	Ru	R	H	Lo	H

Please note that these prompts are translated into English. The actual messages used in the user trials were written in German, as the test runs were conducted in Austria with German-speaking test persons

As seen in the table above, prompts were separated into tips and tasks.

Tips These were short pieces of information regarding trivia about stress reduction or physical activity. They fit into a single iOS notification. Users were instructed to

just read them, and accept the notification when they were done reading.

Tasks Tasks give users a longer exercise to perform., like breathing- or sports exercises. These generally take longer than just reading the prompt, and users were asked to accept them when they were finished performing the exercise. Accepting a message was done by tapping on it once, if users did not feel like doing an exercise or were not able to, they were instructed to dismiss the message by ignoring it or swiping it away.

In the second test run, tasks would be removed and replaced with more tips. While it was good to motivate users to perform longer tasks, the instruction to only accept them when the task was performed, split the prompt outcomes into "read" and "performed". This made results rather incomparable between tips and tasks but also put a greater burden on the user to use the application as intended to provide usable data for the test run. In the first test run, however, there will be both tips and tasks, as seen in the tables above.

CHAPTER 5

First Iteration

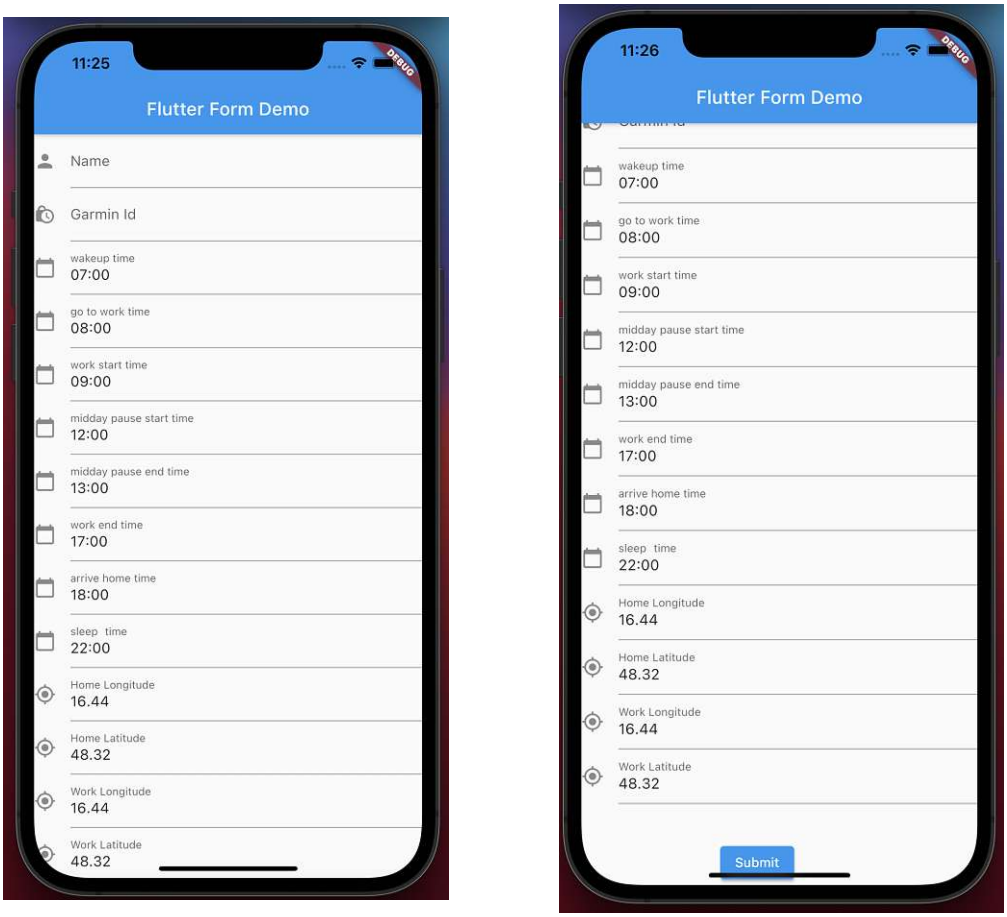
5.1 Development of First Version

It was decided that the development and test phase of the program would be split into two phases. The first version of the system would already have a working backend and frontend, but the goal was to only have a working iOS version for the first test run, and adapt the Flutter app to Android later. After the first development phase, a test run with five iOS users would be conducted, and the feedback from that test run would be used to update the mobile app and further tweak the functionality of the Frontend to achieve better results.

5.1.1 Setting up new users

The first thing users would be asked when starting their two weeks as a test person is to set up their Garmin™device by registering a Garmin™Account, downloading the Garmin Connect application on their phone, and connecting it to their smartwatch via Bluetooth. After that, users were asked to go to a website hosted by the Austrian Institute of Technology, to register their Garmin account, so their fitness and stress data would be synchronized with the backend. Users would then receive a UUID which they could later use to register their Garmin account with their profile of the intervention application.
After users downloaded the application with the Apple test flight program, they would see the screen depicted in 5.1

Here users can set up their personal information such as their name and work times. In the field "Garmin Id", users enter their Garmin id which they got from registering their device on the AIT Backend. Then, the initial registration form offers many different customization options for defining a user's time of day. These include the time a user (usually) wakes up, goes to work, arrives at work, starts their midday pause, ends their midday pause, ends work and drives home again, arrives at home, and goes to sleep.



(a) Multiple Notifications

(b) Multiple Notifications, stacked

Figure 5.1: Two screenshots depicting the user account creation. This will be the first screen a user will see upon opening the app for the first time

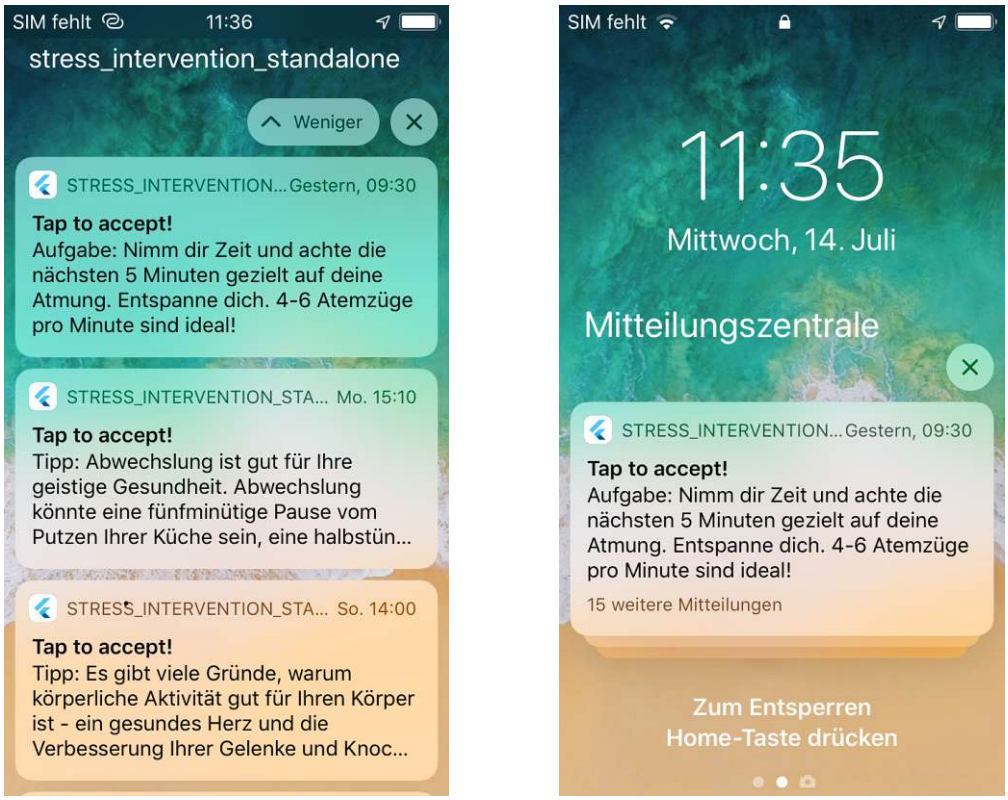
Depending on these times, the system would define what the current "day segment" is currently in, as the same time of the day might have different meanings for individual users and these segments should not be generalized to fixed timestamps for the entire user base. Another important factor is that a user can provide coordinates for their home and workplace, so that the backend can distinguish if a message was sent when the user was at home, at work, or another location. The coordinates will not be sent to the backend but saved locally for privacy reasons. The mobile phone will report at which location a user is currently in, every time the user is within 200 meters of one of the two locations (or when a location is left).

After a user hits the Submit button, the user data will be sent to the backend via a GET HTTP request. After several integrity checks for the data (for example, if the Garmin

ID exists), the user account is created, and the backend returns a JSON Web Token for identification of further requests. If a user's phone already has a token, the registration screen will not show up again if the application is re-opened, but rather the blank screen depicted in 5.3. Also, the moment a user is registered, they can receive push notifications from the backend.

5.1.2 Notification Design

After the content that should be sent out was decided on in the planning phase, it was needed to choose how to deliver and display the content of prompts to a user. The idea was to keep the interaction with the notifications as simple as possible, as the primary goal was to track if the time a user received a message was appropriate or not. Therefore the design was kept rather simple. As seen in the screenshots of 5.2, notifications would appear in their standard iOS design, without any special action buttons.



(a) Multiple notifications

(b) Multiple notifications, stacked

Figure 5.2: Two screenshots depicting the notification design. Taken on an iPhone SE with German language settings

Users were instructed to "Tap the notification" when they either completely read the message (for tips), or when they were right now willing to perform the exercise described

in the notification (for tasks). If users were not able to perform said task right now, due to any reason, they were instructed to just swipe the message away or ignore it. By clicking on the message, the application would be opened, just showing a blank white page as seen in 5.3, and sending a notification to the backend, informing it about which notification was clicked, and when. As notifications were sometimes too long to fit into the "preview" display of notifications, it was sometimes needed to "expand" the notification. A message containing more content than its preview can fit will be displayed like the second or third message of the first screenshot of 5.2 (a), interrupting its text and adding ellipsis points (e.g. "..."). On iOS, this is done by sliding it slightly to the side (or pressing it slightly harder on devices with Apple's 3D Touch enabled) to display the full message. So sometimes users would have to perform this action before deciding if they want to tap on the notification or not.

5.1.3 Notification Scheduling

To provide users with notifications, a system to schedule the notifications was put into place. A test run lasted 2 weeks for a user. The scheduling process was divided into two phases, the first week, which will be referred to as **"Random Phase"** and a second phase spanning over the second week, is named the **"Deliberate Phase"**.

During the **Random Phase**, a user would receive up to 3 messages a day, randomly spread across the day. To achieve this, the backend generates 3 points in time every day which users will receive messages in. As there is no reasonable benefit to creating randomized times for each user, random times will only be created on a per-day basis to avoid unnecessary processing and provide a more easily scaleable time generation method for the backend. One issue however is that users might have different times submitted for standing up or going to sleep, and to avoid sending users messages at times when they are asleep, as they are guaranteed to not respond to notifications, and if they do anyway, it means they woke up, preventing them from getting proper sleep, which is even worse. Therefore, the backend does not create points in

Figure 5.3: Screenshot of the application when opened. Translation of the message: "App is running in the background, new messages will arrive soon! "

time, but three percentages from 0 to 100. These percentages are then mapped to the time a user is awake.
An example of this mapping can be found at 5.1. In the table, two example users are shown. The start and end points of this mapping are set by the wake-up- and go-to-sleep-times of users. **User 1** wakes up at 8 a.m. (which corresponds to 0%) and goes to sleep at 10 p.m. (corresponding to 100%). As for **User 2**, the generated percentages of 0 and 100 are mapped to 6 a.m. and 9:30 p.m. respectively. If a number between 0% and 100% is generated, the values will be mapped to times-of-day for **User 1** and **User 2** as seen in the table 5.1. For example, if the backend randomly generates 30% as a point in time where a message will be sent out that day, then it will be scheduled for **User 1** in their "'Morning - Work " Section at 12:20, while **User 2** receives the message their message at their corresponding time for 30%, therefore at about 10:39 a.m, during their break.

Table 5.1: Example of how generated values will be mapped for two different users

User 1				
Section	Start - Section	End - Section	Start - Percentage	End - Percentage
Morning - Home	08:00	09:00	0%	7,14%
Morning - Commute	09:00	09:30	7.14%	10.71%
Morning - Work	09:30	13:00	10.71%	35.71%
Break	13:00	13:30	35.71%	39.28%
Afternoon - Work	13:30	17:00	39.28%	64.28%
Afternoon - Commute	17:00	17:30	64.28%	67.85%
Afternoon - Home	17:30	22:00	67.85%	100%

User 2				
Section	Start - Section	End - Section	Start - Percentage	End - Percentage
Morning - Home	06:00	06:30	0%	3.22%
Morning - Commute	06:30	07:00	3.22%	6.45%
Morning - Work	07:00	10:30	6.45%	29.03%
Break	10:30	11:00	29.03%	32.25%
Afternoon - Work	11:00	18:15	32.25%	79.03%
Afternoon - Commute	18:15	19:30	79.03%	87.09%
Afternoon - Home	19:30	21:30	87.09%	100%

The *Notification Manager Service* of the backend is responsible for running a CRON-job every 10 minutes, which checks if any of the three generated points-in-time is reached, for each user. If this is the case, then the backend gets the context of the current user. There are some hard-coded limits where the delivery of the message will be canceled. If a user's context suggests that they are currently in a car, the message will not be sent, even if it is currently the "randomly generated time". Furthermore, users will not receive

messages if they are asleep according to their schedule, even though this limit never matters during the random phase, as only times when the user is awake are generated. After a user receives a message, they can answer it by tapping on it. This will trigger a REST response to the backend, registering that the message was answered. To ensure that the notification was answered "just-in-time", responses must be sent within 45 minutes. After 45 minutes, notification responses can still be sent to the backend, but the backend will register them as "dismissed"

The *Random Phase* lasts exactly for one week. After the initial week which serves the purpose of providing data when a user is likely to respond to messages, and under which circumstances they are most likely to just be an annoyance or unnecessary. After one week of randomly timed messages, the **Deliberate Phase** is started for a user. During this phase, the *Notification Manager Service* keeps checking the current state of the user every 10 minutes, by requesting the current status of all registered users from the Variable Aggregator Service. Furthermore, the notification manager service then forwards the current state of the user to the *History Manager Service*, which then checks if a user is likely to respond under the current circumstances, by checking the history of the user's likeliness to respond to messages (the inner workings of this decision algorithm are described in 5.1.4). If the History Manager deems the current situation to be beneficial and reports this information back to the Notification Manager Service, the Notification Manager Service will then ask the Rule Engine Service for an appropriate message for the current circumstances. The Rule Engine Service gets a JSON object referencing the context of a user and filters out all possible notification texts that do not make sense for the current user context. Out of all available messages that were not filtered out, a random message is picked and sent back to the Notification Manager Service. The Notification Manager service then persists the information which messages were sent out to which user at which time in a database. Also, the notifications get sent out to the users via Firebase notifications. Users then get 45 minutes to answer a notification, and every new notification answer will influence the History Manager's future judgment about whether or not a message will lead to a successful response or not.
The backend sends up to three messages out to a user, two of which are deliberate, and one *explorative-random* message. The explorative message is one random message every day - just like all messages in the random phase. The purpose of this message is to avoid plateaus and give the system a chance to move out of a state where a local maxima or minima is reached in the learning process. While the CRON-job is executed every 10 minutes to check the state of all users, there are three "killer criteria"that can be met and the system does not send a message to a user. This happens when a user sleeps, is in a vehicle, or there was already a message sent out in the last two hours. This is done in both the deliberate and the random phase, so a user might not receive three messages a day, even though three were scheduled because some were skipped.

5.1.4 Evaluating response likelihood with Naive Bayes

As previously mentioned, the History Manager gets to decide if a current context will lead to a successful response from a user or not. During the first test run, the solution of choice for this task was to create a Naive Bayes Classifier Model for each user, based on their previous answers. This algorithmic solution was picked due to it being used in a similar application for predicting the response rate of a user to notifications.

The Naive Bayes Classifier is based on the basic Bayes theorem (Equation (5.1)). Using this theorem, the probability of A can be found, if it is already known another event B has happened.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{5.1}$$

The Naive Bayes Classifier makes use of the theorem using several predictors/features (in this case, the predictors are made up of the different variables of the context of a user), in order to predict the likelihood of these several factors leading to the user giving a response to a notification, based on the historical evidence for that user.
An important thing to note here is that the Naive Bayes Classifier assumes that the individual components of the context are completely independent of each other. The variables taken into account for the user context might not be a hundred percent independent from each other. For example, if a user is walking or not, might be influenced by the fact that a user is currently in the office or outside. Therefore, Naive Bayes's ability to predict a user response might not be as accurate as it could be if dependencies between user variables were correctly taken into account. Another assumption that the classifier makes about the outcome is that all variables should be weighted equally. This might not be the case in general and might differ even further between users.
For this project, a **Multinomial Naive Bayes** was used. It is often used to classify documents into one of several different categories, based on the frequency of different words appearing. For the intents and purposes of notification response prediction, the classifier tries to classify a context of a user into either the *"Yes, the user will respond."* category, or the *"No, the user will not respond."* category. The formula for the classifier itself can be seen in 5.4. [69]

The formula calculates the probability of the current context of a user leading to a response. If the probability is at or above 50%, a message will be sent. The probability is calculated as seen in 5.4. Therefore, for each of the six variables, the system checks how many notifications were already sent with the variable having the value of the current context, and how many times this resulted in a positive answer from a user. These two values then get divided to determine the probability of that specific value resulting in a Yes-Instance. To avoid division by zero, the system adds additional **"black box values"**. Two to the total amount of occurrences of that value, and one to the occurrences which led to a positive response. An example of this can be seen in 5.5

By adding the black box values, neither side of the division will ever be zero. If there is

$$P(R|C) = \frac{P(C.Loc|R)P(C.Str|R)P(C.MS|R)P(C.PA|R)P(C.TS|R)P(C.DS|R)P(R)}{P(C.Loc)P(C.Str)P(C.MS)P(C.PA)P(C.TS)P(C.DS)}$$

$P(x/y)$	Probability of x, happening, assuming y
R	User responds within 45 minutes
C	Current context of a user (According to context)
$C.Loc$	Current location of a user (According to context)
$C.Str$	Current stress level of a user (According to context)
$C.MS$	Current movement status of a user (According to context)
$C.PA$	Current physical activity level of a user (According to context)
$C.DS$	Current day segment of a user (According to context)
$C.W$	If it is a working day or not (According to context)

Figure 5.4: Naive Bayes Classifier Formula

$$P(R|Home) = \frac{HomeYes + 1}{HomeOverall + 2}$$

$P(R\|Home)$	Probability that a user will respond, based on the assumption that the user is at home
$HomeYes$	Amount of previous notifications where the user was at home and responded in time
$HomeOverall$	Total amount of previous notifications where the user was at home

Figure 5.5: Example for calculating if a certain location value will lead to a response from the user

no data at all in the data set, this equation will amount to a 50% probability of the user responding under these circumstances. Furthermore, adding the black box values makes sure that the equation never becomes 0%. This is important, as there could only be a small data set for a certain value. Even after a week of testing, it could be that there was only one message sent where the user was walking. If no black box values were used, and the user did not respond, the system would think that there is a 0% probability of the user responding, and the user would never again receive any notifications while walking, even though the user did not respond just once under this condition. By using black box values, the notification might still be sent, if the other parameters are decent. Furthermore, the black box values become less relevant with every new addition to the notification database. Their purpose is to make it easier to cope with data sets that have very little or no information about a certain value.

5.2 Evaluation of First Version

To evaluate this first version of the application, a set of five users was picked to evaluate the program. The goal for the first version was to have a working backend, and an iOS version of the frontend application, and the goal of the first test run was mostly - besides getting some first testing data - to identify any issues with the application or the general

workflow, so that the software can be altered and improved to allow for a proper, second test run if needed.
For this test, a set of 5 people (2 male, 3 female) was picked. As the application was designed for office workers with a full-time working schedule in mind, the most important criterion for picking participants was that they

- Own an iOS device compatible with the notification application
- Own an iOS device compatible with the Garmin application (iOS 14.0+)
- Have a schedule that is as close to a 5-days-a-week job as possible
- Commute to the office and back on a workday

Unfortunately, the last two criteria were not as easy to fully fulfill. Due to the Covid-19 pandemic, many office workers were able to switch to a home-office setting. Therefore not all test subjects traveled to and from their workplace every day of the week. Furthermore, as the test run was held in August, some people took some days off, so a workday on their schedule might not always correspond to an actual workday. Nevertheless, the test persons offered some great insight for a first evaluation of the application and provided great feedback about their experience with the app, which helped improve the system for the second test run.

5.2.1 Setup for test users

All of the test users first got a Garmin Device and were asked to create a Garmin account and download the iOS application to their private smartphone. After that, users were asked to register their Garmin account on the Website of AIT so that their data would be synchronized with the AIT database. Finally, users received an invite to download the notification application via Apple's test flight app. After that, the user was asked to enter their personal information about their schedules and coordinates of their workplace. After that, their device and account would be registered with the backend, and from that point on, their 7 days randomly timed message period would start, followed by 7 days of messages scheduled with the Naive Bayes Classifier. Users were instructed about the two different types of messages, **tips** and **tasks**. Users were asked to tap on a notification with a tip after they read it and to tap on a task only if they were planning to do a day task, otherwise, they should dismiss them (which results in a dismissed message, even if the user read it). All 5 users started their test run within a week, but not all on the same day. But all ended after 14 days of using the app.

5.2.2 Results of the First Test Run

The application aims to send out notifications when users will most likely respond, but avoid sending notifications when a user will most likely not see the notification or won't

respond. So ideally, all users would have higher response rates from specifically tailored messages, instead of just randomly timed ones.
After 14 days, the test run was stopped, and the results of the 5 test users were collected, to evaluate the effectiveness of the Bayes model, and how the application should be improved for a later release.

5.2.3 Benefits and Issues of the Garmin Ecosystem

The three major reasons for picking Garmin devices over other fitness trackers are that the Austrian Institute of Technology already had a good insight into the usage of the Garmin Ecosystem, and reported that they had positive experiences with previous research projects. Furthermore, due to the cooperation with the AIT, Garmin Fitness trackers were readily available for this project. Finally, the Austrian Institute of Technology already had hardware and licenses set up for using the Garmin API to allow for easy access to user data.
Users reported their own stress and movement data tracked by the Garmin Device as being plausible when looking at their daily report in the Garmin App. As seen in 5.6, users can see their information in the Garmin application, nicely displayed in differently colored graphs and various data widgets.

This data gets also synchronized with the backend. Unfortunately, not all of this data can be accessed immediately. Garmin sends quick reports every 15 minutes to be used with the notification app. Taking data that is at worst up to 15 Minutes old is already quite hindering for using them to evaluate just-in-time interventions. These "Quick Reports" do not include all necessary information to evaluate a user's stress and physical activity level, but mostly consist of how many steps were taken in the last 15 minutes, combined with additional information about the user's current activity, but this is not always reliably included.
The more reliable method of gathering information is the daily reports, which are sent at midnight, and include all of the information a user can see in the application. The issue here is, of course, that notifications would ideally have up-to-date data about a user's status the moment a notification is sent out. With daily reports, this is not possible. So to still use stress and physical activity as parameters, the Bayes Model for this test used the *average over the last two days* for this value.
This value however seemed to be rather unimportant if a user would respond to a notification or not. First of all, as the average was taken over a quite large period, it was mostly the same for a user at all times. And on those rare occasions, the average amounted to something out of the ordinary, there was no conclusive evidence if this actually affects the response rate of a user in any signification way, as the occasions where this happened were very rare.

Another major issue with the Garmin application is that the application did not consis-

(a) Overview in Garmin App (b) View of Daily Stress Report in Garmin App

Figure 5.6: Two screenshots of the Garmin App, depicting some of the Information a user sees about themselves (Steps, Heart rate, Physical Activity, Stress Report...)

tently sync data with the backend server. The Garmin app is supposed to always sync its data automatically with the API, but sometimes it randomly stops synchronizing data for a user. In that case, the users were asked to manually sync their data. This can be easily done by hitting the “Synchronize“ Button, which can be seen in the top right corner of picture (a) in figure 5.6, but the user must be made aware of this, and even if they do this, it might stop synchronizing at any time again. Also, there is nothing which makes the user aware of the fact that their app stopped synchronizing, as they will still see all of their data in their own, local app. Fortunately, hitting the synchronize button retroactively adds all reports to the backend, even if they are several days old at that point. Due to this, it was decided for later versions that the physical activity and stress parameters could not directly be used to time the notifications at the time they were being sent out but to use the data later in the analysis as evidence to consider whether or not people were more likely to respond to messages when they were in stressful situations

or situations of high physical activity.

On the positive side, users had no issues at all setting up the Garmin devices by themselves and reported using the fitness trackers as an interesting experience. There were no technical issues or usability issues reported by users about the Garmin devices themselves. One user reported that she felt motivated to do more sporting activities in her life, just because she was wearing a fitness tracker all day, and liked the gamified experience of seeing her sporting progress in the application.

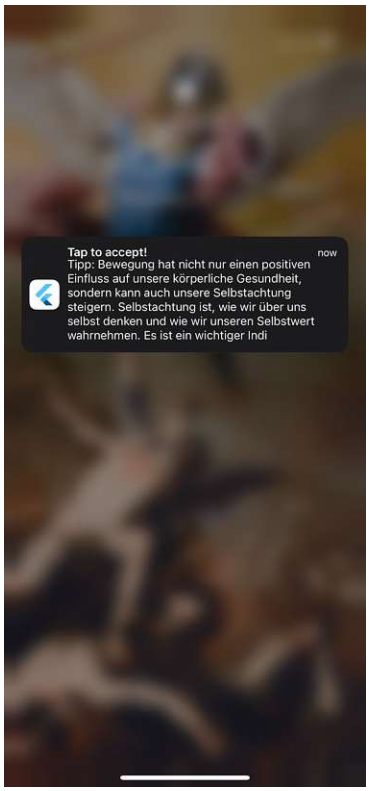
5.2.4 User Feedback and Technical Issues

Users were guided through the setup of the application, and after everything was ready to use, they were left alone with the application for the duration of their test run. Users were also instructed to follow guidelines for tips and tasks, in which case they should manually dismiss tasks if they would not do them.
Several users did report that often they had a few minutes to read a message but would dismiss them as the task they would have to perform was not in the scope of their currently available time. Therefore, if the measured success of the system was only dependent on the fact that users responded or not, then tips would have a significantly better response rate than tasks, and the potential response rate can be improved by only sending tips. Furthermore, this made notifications sent across the day harder to compare to each other. While tips are always just a quick read and roughly equal in time commitment terms, tasks can not only take up different amounts of time but also tasks of roughly equal length can have different acceptance rates under the same circumstances. For example, a task can be seen as "socially deviant", as suddenly standing up during an office day to perform sit-ups in front of one's superior might not be a very attractive task, while having a quick internet research about something healthy to eat might be acceptable in the same scenario. While the application did attempt to filter out tasks that seemed inappropriate for the current user context, it could still send out notifications which would take varying amounts of time commitment, which would sometimes lead to users dismissing messages due to time constraints, even if they would have had time to read a short tip notification. A further complaint by users about the notifications was that many of the tips were either similar or that they had the feeling of seeing the same messages many times. This happened due to the user's context not changing as much as originally anticipated in the design phase. Often users would remain in very similar user-context-variable-combinations throughout their week, which resulted in them getting notifications from the same pool every time. Because of this and the limited supply of distinct notification messages, the pool of notifications was a bit too small under some circumstances for some users. While users were still clicking in duplicate messages, they did not really find a huge enjoyment in reading them a second time or stopped reading them precisely and accepted them straightaway without fully re-reading them again once they realized they were familiar with the text.

Another reported issue people had with the notifications, was that some of the notifications were too long and were cut off on their iPhones, even when expanded. Surprisingly enough, this did not happen when tested on an older iPhone 5S model with iOS 12.5.4. The full message does arrive on the iPhone via the API according to logs, but the notification will be cut off if it is too long on newer models, even when fully expanded. The threshold for messages seems to be 256 characters for newer iOS devices, according to internal tests, as seen in 5.7

(a) Screenshot #1 from iPhone SE (2nd generation, released in 2020)

(b) Screenshot #2 from iPhone SE (2nd generation, released in 2020). Notice that the notification text ends with a white space



(c) Screenshot #1 from iPhone XS Max

Figure 5.7: Three Screenshots of notifications being cut off after 256 characters on newer iOS devices.

While this did happen on newer devices, it did not happen on the older iPhone 5S which was mainly used for development. In the picture 5.8, a comparison of the same notification being sent to an iPhone 5S (released in 2012) and an iPhone SE (1st generation, released in 2016) can be seen. The message has well beyond 256 characters, yet it is only displayed correctly beyond the 256-character limit on the iPhone 5. The iPhone SE message is instead cut off at 256 characters again. While the two iPhones in the Picture do use different system-wide settings for the text size, this does not seem to be the reason why the text is cut off on the right iPhone, as it can clearly be seen in the screenshot compilation 5.7 that other iPhone models with smaller text size also cut off messages at exactly 256 characters. The prompts were also sent from the same backend infrastructure, with the same Firebase and Apple push notification setup. So this is not a limitation on the backend side, but a case of the operation system hiding information from the user.

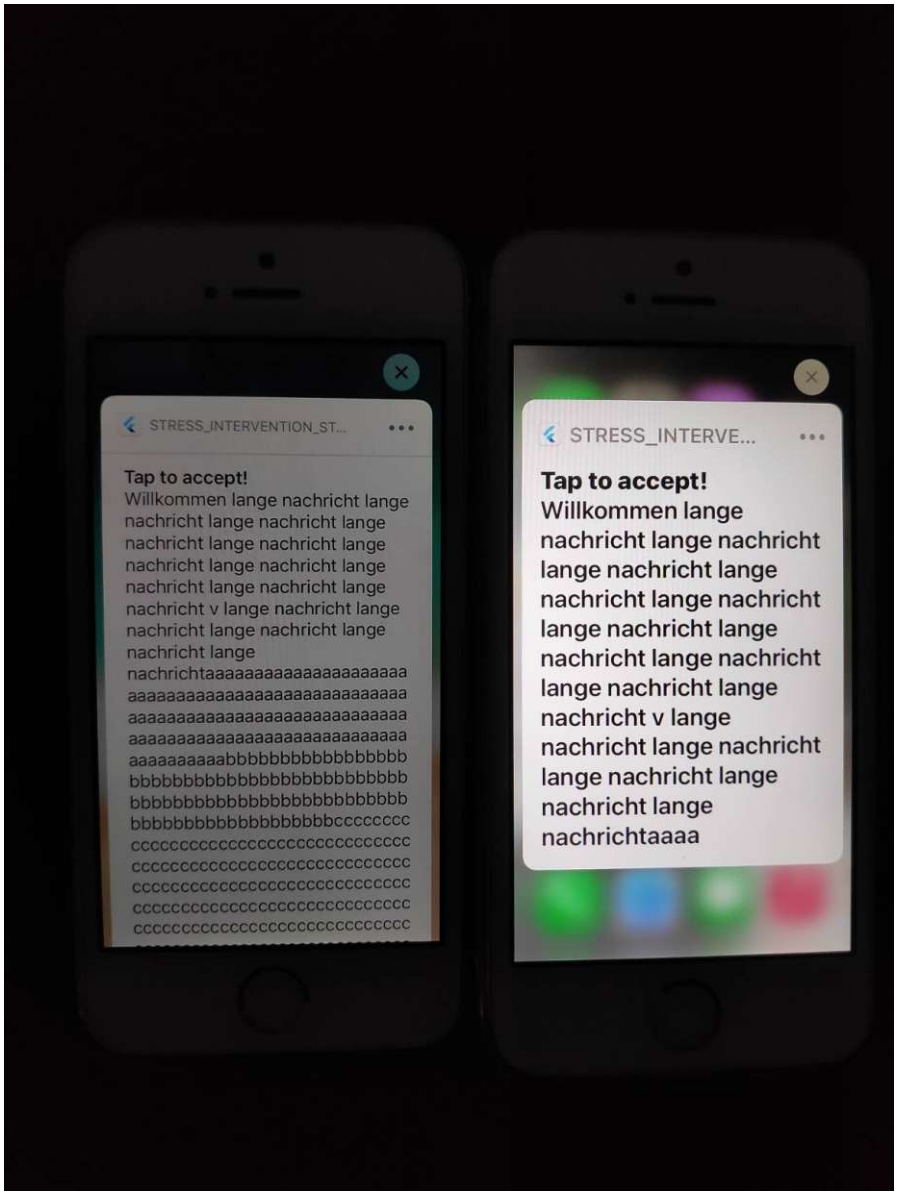


Figure 5.8: An older generation iPhone 5S (Left) and a newer generation iPhone SE (Right) displaying the exact same, long notification.

Interestingly, this behavior does not seem to be documented anywhere officially by Apple. The official guidebook for sending remote notifications mentions that there is a limit between 4 KB (4096 bytes) and 5 KB (5120 bytes) for the notification payload [70]. However, this is much more data than the roughly 300 bytes of payload used for the messages in 5.7. There are however a few blog- and forum posts of different users experiencing the same issue, however, it is hard to find a credible and reliable source

for when exactly this issue happens, and what the exact limits for notifications are. For example, a forum post from the Apple developer forums mentions that they experience this issue on “iOS 14 or later“, and that the cutoff is at “around 256 characters“, while this does not happen “for iOS 13 or earlier“ [71]. Another user [72] claims to experience a similar issue with the “shortcuts“ app for iOS, where they claim that notifications are cut off after “256 characters (including spaces and returns) or 20 lines. Whichever comes first.“ According to this post, this happens “as of iOS 15.6.1“ with an iPhone 13 mini. While notifications of the app of this thesis are indeed cut off after 256 characters, no tests were made if the 20-line limit holds. But during tests made for this thesis, it was possible to display a notification with 22 visible lines inside of the notification body (see 5.8) on an iPhone 5.

As previously mentioned, the main device used for testing during development was an iPhone 5. When the app is first opened and all user information for registering the device is entered, the application will ask the user for the permissions it needs to function properly. This includes the *GPS/location data permission* of a user, and permission to access system information about whether or not a user is walking, sitting, or moving in any other way, to determine a user’s movement status. This information can be accessed by an app by asking the user for the *Health & Motion activity* permission.
Asking for the location permission worked without unexpected issues for all users. Newer iPhones will give users the option to allow the location permission only while using the app (which would not work with the app’s intended usage), while older iPhones with older iOS versions will only give users the option to either completely allow or disallow the location usage. But as long as users selected “Allow always“ as instructed, the app worked as intended.
An unfortunate issue however came with the *Health & Motion activity* permission. This permission is linked to the privacy settings regarding motion and fitness on an iOS device. The operating system on Apple phones allows users to completely disable the “Motion and Fitness“ setting for all apps (as seen in 5.9). Unfortunately, this led to the application not being able to ask for the motion permissions at all, and it also did not display an error message to the user that this is a mandatory permission for using the app.

On the bright side, the distribution of the app via test flight went well. Furthermore, there weren’t any performance issues with the backend, as the usage by 5 users simultaneously over more than two weeks worked flawlessly. Furthermore, there was no downtime of the backend at all. While no quantitative measurements of battery usage were taken, users reported that the battery usage of their phones did not seem to take a noticeable hit compared to when they normally used their phones. Also, the delivery of notifications to end-user phones worked reliably. The transmission of tracking data from users’ phones to the backend - aside from the previously described issues - worked as intended.

5.2.5 Quantitative Results of User Interaction

The first test run was mostly intended as a technical and logistical experiment, to evaluate which parts of the software system should be improved for the final version, and to

Figure 5.9: Motion and Fitness Settings on iOS [73]

pinpoint technical and usability issues to improve on.
Nevertheless, the notification response rate of both the deliberate and the random phases were checked for all users, to evaluate the first version of the Bayes classifier.
The collected data was used to identify the tendencies of users and to check if the decisions that the Bayes classifier would make were reasonable and resulted in better response rates. Unfortunately, in this first test run, the Bayes classifier did not work the way it was intended.
One of the major issues of the system was that it was programmed to send a response to a user if the Bayes classifier determined a certain situation to have an above 50% response rate. The fatal flaw of this system is, that some users responded to basically all messages, and some to almost none. If a user had a near 100 percent response rate in the first week, this would result in the system determining every situation as a situation

in which a user was likely to respond to messages, as even the theoretical worst situation would be seen as *"good enough"*. As the backend checks every 10 minutes if the current moment is a good moment to send a message, this would lead the system to instantly deploy the first message at the first instance it could on a given day - right at the time of day that a user set at their wake up time. As the backend has a limiter to not send another message for a user for two hours after a notification was sent out, the following second deliberate message would be sent 2 hours after the set wake-up time.

Table 5.2: First week with only random messages

PersonId	Total Sent	In time	Not in time	Response rate
Person #1	20	8	12	40%
Person #2	18	16	2	88%
Person #3	20	17	3	85%
Person #4	21	9	12	43%
Person #5	18	9	9	50%

On the other hand, for people who had a very low response rate in general, the algorithm would deem all situations as "below 50%" probability that a response would be sent, which often lead to users not receiving any messages throughout the day, as every single situation would be deemed as *"not good"*.
During the deliberate week, 2 different kinds of text messages were sent out. First of all, two deliberate messages were sent out during the day when the system decided the current parameters of a user would more likely than not lead to a positive response. This unfortunately lead to users with a high response rate always receiving their two messages right after they woke up, as the system would deem every situation to be "'good", while users with a low response rate might receive no messages at all, or only during times and situations which had extremely high response likeliness in the first week.
Secondly, during every day of the second week, one "explorative random" message was sent out as well. It was basically a random message like in the first week, and its intended purpose was re-calibrating the trained model if it should run into a "plateau".

Table 5.3: Second week with deliberate and explorative random messages

PersonId	Total Sent	In time	Not in time	Response rate	+/-
Person #1	19	10	9	52%	+14%
Person #2	20	15	5	75%	-13%
Person #3	20	10	10	50%	-35%
Person #4	9	3	6	33%	-10%
Person #5	7	1	6	14%	-36%

As seen in 5.3, the response rate of users went **down** for most people, as much as 36 percent compared to the first week. Furthermore, the amount of messages sent in the second week dropped a lot for some users.
Person 5 only received the 7 explorative random messages, as the response rate of that user was roughly 50% in the first week. As the person did not respond to the first explorative-random message in the second week, the response rate dropped further, leading the system to believe that the person is below 50% likely to respond in most situations, never to send another deliberate message again. Therefore only the random messages remained.
For *Person 4*, a similar thing happened, as the response rate was rather low in the first week. The system would only send a message under very specific circumstances, as it detected the user was very likely to respond when they were walking and when it was in the afternoon, so only two deliberate messages were sent out during the whole week when these two circumstances were met at the same time.
Person 2 and 3 had a quite similar test run. Both answered almost 90% of the random messages that were sent to them in time in the first week. This meant the system assumed that every situation was a good situation to send a message. This meant the system attempted to send a message right after it was allowed to every day - right when a user woke up, or more accordingly, right after the time they set as wake-up time. According to a post-test interview with the users, their set wake-up time was an approximation and not too precise. Often they would wake up several minutes after their initially set wake-up time, which would lead to them being either still asleep when a message was sent, or still in their morning routine before they had a chance to check their phone.
The only increase in responses was for Person 1. The person had just a 40% response rate, but 3 random messages in the first week were sent during the “MORNING_WORK“ time slot of that person, and all of them were answered. The system capitalized on the 100% response time during “MORNING_WORK“ the second week and sent out seven messages during that time period, 5 of which were answered in time.

CHAPTER 6

Second Iteration

6.1 Development of second iteration

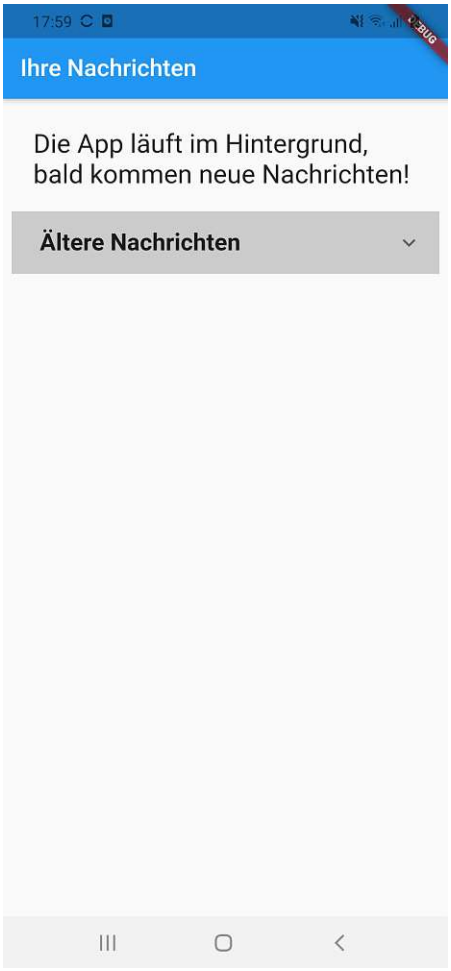
After the first test run, its data was analyzed and decisions were made about which part of the system needs improvement. This was done by looking at the response data which came back from users, but also by listening to user feedback. Users were encouraged to report any issues they had with the application. Furthermore, there were short interview sessions conducted with the users to get a grasp on what worked and what did not work all too well with the user experience.
With this accumulated data, aspects of the application that needed improvement were identified. In the following sections, the individual findings together with a proposed solution and their development process will be described in the sections below.

6.1.1 Frontend Design

Regarding Frontend design, a complete overhaul for the UI was created for the second test run. Not only did this include a new way of engaging with the notifications themselves - by actually forwarding the user to a detailed view of that notification - but enabled a historical timeline view of all previous notifications, giving users a more in-depth look at the previous messages they received, and whether or not they reacted in time. This way users had a better overview of the messages they received throughout the week.
Starting with the visual changes, the first change in appearance happens on the default home screen.

> **Note**
>
> As a major part of the second iteration was to make the application available on Android devices, the following screenshots will be on Android devices.

(a) The base look of the application **before** the rework, Screenshot taken on Android



(b) The base look of the application **after** the rework, Screenshot taken on Android

Figure 6.1: The evolution of the home screen

Account creation

While other views had more drastic changes for the second iteration of the application, the account creation stayed the same as in 5.1. While the design of the account creation was rather simple, it did not pose a problem to anyone who would get into the application. Further supporting the choice to not focus the efforts of the UI overhaul on the user

creation screen is the fact that users will only see it once, and improvement efforts are better suited for screens the user sees more often, as it benefits the user more often than improvements to the initial login screen.
Secondly, users will be guided through the creation of their account anyway. Even if there could be parts that could be reworked in a way that would make them more self-explanatory, if the user has any questions during the account creation process, they can always ask their overseer if what they do is correct, while the overseer can interfere if a user would enter something incorrect during the login process.
Therefore, the account creation view was kept the same, and efforts were put into re-working the views which did not achieve the intended functionality.

Home screen

The evolution of the home screen can be seen in 6.1. Before the rework of the user interface, the home screen had no interactive functionality. The original reasoning for this was that it did not objectively *need* any functionality, as the application focus was only on the notifications and answering them. But since notifications needed to be tied to an application that would also be open-able when either a notification was pressed, or the user just opened the application by clicking on its application icon.
So while there was no intended functionality missing, people were indeed confused when they clicked on a notification, that they were greeted by an empty screen without any information. While there was a message sent to the backend just as intended, and the application worked technically as expected, users were a bit underwhelmed and confused, that upon opening a message, they were greeted by a blank screen.
To give users clearer feedback that the software registered that they saw their notification and clicked it, two solutions were implemented in the frontend application.

- Firstly, upon tapping on a message - which would result in the application being opened - users would now see a detailed view of the message they just tapped. This feature is further described in 6.1.1
- Secondly, users can now also see their past messages. This is to avoid presenting users with an empty screen when inspecting the application on its own via its application icon, without opening the application through a notification. This feature is described in 6.1.1

Notification detail

While previously there was no indication that the application correctly registered the user's tap on the notification message, the re-design has the detailed notification message displayed. A screenshot of this re-design can be seen in 6.2. Additionally to the notifications contents, the user also gets information about when the message was sent out, and how much time elapsed between the sending of the message and the user opening it.

Additionally to the new message display, messages could now also receive a ”Like”, just like in many social media applications. As is convention, the like button is displayed by an outlined thumbs-up symbol, which becomes filled with a black color once it is tapped. The corresponding text for the like button asks if the message was helpful or not. Upon liking a message, the information that the user liked the notification was stored in the backend. While the information about liked messages was stored, they were not taken into account for scheduling new messages. The reason why a like button was included is to give users an extra layer of interactivity and to give users another way to interact with the application once it was opened upon clicking a notification message. This is to prevent users from having the previous experience, where the app would open, and users were confused since there was nothing else to do but stare at a blank application screen.

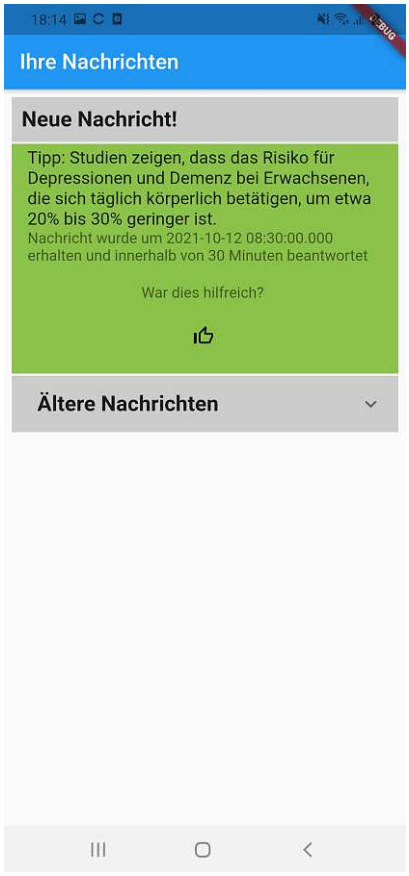


Figure 6.2: How a new message is displayed

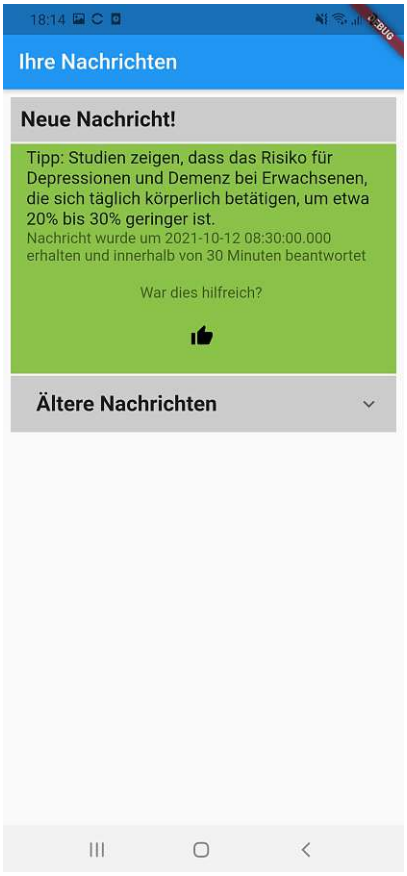


Figure 6.3: A message after receiving a “Like”

Notification history

To accompany the new changes to the notification details as discussed in 6.1.1, there was also an additional ”History View” implemented to allow users to access their old

notifications which they received throughout the test program. As can be seen in 6.1, the home screen was expanded by an additional drop-down for "Historical messages" (or "Ältere Nachrichten", in German, as seen in the screenshot). Upon clicking on this expandable menu, the user sees a list of previous messages, sorted in order of their appearance. The view can be scrolled down to reveal even older messages which do not fully fit in the initial view. A screenshot of the expanded historical messages can be seen in 6.4. Old messages are displayed in three colors: green, yellow, and red. These colors indicate how a user responded to a message.

- Green tells the user that a message was checked within the time frame the backend considers "in-time", which is that a response was sent by the user within 45 minutes of sending the response.
- Yellow means that the message was accepted by the user and the response was registered by the backend, but the response was not sent in time, therefore more than 45 minutes elapsed before the message was answered.
- Red means that the backend did not receive any answer to that particular message at all. This can happen if the user never taps the notification, swipes it away in the notification tray, or clears all notifications before ever tapping the notification. Another reason might be if there was a technical error when the user tried to submit an answer, for example, if their phone has no internet connection when attempting to send a response.

Other than the *detail view*, the colored messages of the *history view* are also accompanied by a smiley face, which corresponds to the colors (Sad face for red, happy for green, neutral for yellow). Just like in the detail view, the user can see detailed information about the response and send-out time, accompanied by a thumbs-up symbol if the user liked a message (and no symbol if the user did not like the messages). By displaying old messages, not only do users have a good overview of messages they received throughout the experiment, but the application also provides some functionality other than a befuddling with a blank screen if it is opened through the application icon.

6.1.2 Android Version

A major change from the previous test run was that this time, three of the test persons were Android users, while it was just iOS before. Development for iOS was a much easier task, as iPhones are quite similar to each other, and if the application works on a test device, it will mostly work on other iPhones as well. While Flutter offers native export to both iOS and Android devices, promising the same functionality for both operating systems, the dependency packages might not always work the same way on Android and iOS.

The biggest downside of Android devices for this application, in particular, was that there are no *strict* or *consistent* rules on when Android Devices kill off applications that try to operate in the background.
If you want to constantly track a user's position or geo fences on iOS devices - while there is no way to force an "always on" solution - the rules of how the tracking will work on a device are clear and transparently communicated by Apple's documentation. While the application is running, it tracks the user. If it is open the screen is turned off as well. If it is in the background, it might track the user more inaccurately, and if the user manually closes it, it will reactivate if a user moves out of a preconfigured-length radius (in this case the smallest possible radius of 200 meters was selected).
While there is no way of "forcing" the application to stay always open, at least you have a consistent system of knowing when the application will activate and deactivate. For Android systems, this is not as simple. Android devices are produced by many different vendors, and each manufacturer sets their phones up in a different way. As mentioned by the *"Don't kill my app!"* website, many manufacturers try to artificially improve the battery time of their smartphones by aggressively killing off applications in the background or limiting their functionality. While this may yield better results regarding battery lifetime, this will break the proper functionality of applications that rely on background activity.

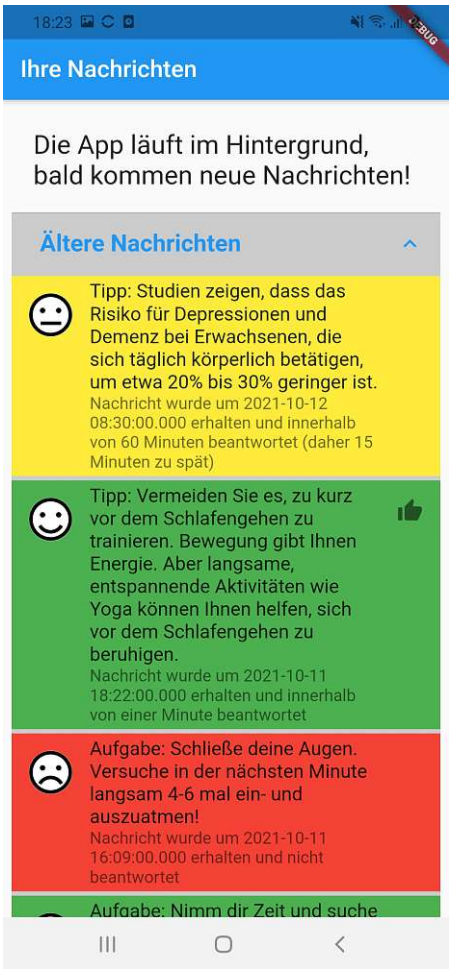


Figure 6.4: Old messages displayed in the reworked app

Not only do phone models and manufacturers make a great difference, but since Android smartphones are much more customizable than iPhones, individual user settings and battery-saving settings can matter a lot. To give each phone of every test person the best chances of the application running in the background, users were tasked to follow the instructions on the *"Don't kill my app!"* website for their smartphone to keep the application up and running [74].

6.1.3 Backend Changes

Notification Content

The notifications of the first iteration caused several issues. First of all, tasks and tips had different lengths, and users were instructed to only accept tasks if they would do

them right away. This meant that “positive responses“ were rather incomparable between tips and long tasks. While tips only required a user to read a short message which only took up to 30 seconds, a task that prompted a person to go for a long run which could take an hour might not be a task that a user would always perform, yet they were treated in the same way by the system. To make messages with different content comparable, tasks were completely removed and only tips were kept.
Not only did this create a system where notification responses were more comparable, but it also simplified the user experience. Users could just tap on a notification when they were interested and would not be forced to think about whether or not this task was doable at the moment. Moreover, this cut down on potential usability errors, as users now only had two options: Opening the notification when they wanted to read it, or not. The users did not have to think anymore about whether or not the task was currently doable. Furthermore, as all tips take roughly the same, small amount of time to read, the “Length“ property of notifications was removed.

Furthermore, as previously mentioned in 5.2.4, messages would be cut off after 256 characters. All notification texts in the new iteration were simplified to conform to this threshold (The notification texts in this paper were translated from the original German messages from the test run. Therefore, notification texts in this paper might or might not conform to the 256-character limit)
A table including all notifications can be seen in the table below.

For the second prototype iteration, the following tasks were used for Physical Activity Tasks:

Id	Task	Loc.	Min. Mov.	Max. Mov.	Min. Strs.	Max. Strs.	Min. Phys.	Max. Phys.
1	Tip: Try to sit neatly and upright, and not to curl the spine	Un	St	St	R	H	Lo	H
2	Tip: Try not to sit for too long. Also, take breaks and go for a walk if you have already been sitting for a longer time	Un	St	St	R	M	Lo	H
4	Tip: Try to avoid prolonged sitting and interrupt it with a short exercise such as squats or push-ups	Un	St	St	R	Lo	Lo	Lo
6	Tip: If they want to get more exercise into your daily routine, try 12 squats today!	Ho	St	Wa	R	H	Lo	M
7	Tip: If you want to get more exercise into your routine, try doing 12 push-ups today!	Ho	St	Wa	R	H	Lo	M
8	Tip: A long walk is a practical way to get more exercise into your daily routine	Ho	St	Wa	R	M	Lo	M
11	Tip: Experts believe that exercise releases chemicals in your brain that make you feel good. Try to make a physical activity that you enjoy a part of your day!	Un	St	Wa	R	M	Lo	Lo
12	Tip: Regular exercise can boost your self-esteem and help you feel better. Experts say that most people should exercise for about 30 minutes at least 5 days a week.	Un	St	Wa	R	M	Lo	M
13	Tip: Exercise also keeps your brain and your other vital organs healthy. Exercise doesn’t just mean playing sports or going to the gym. Walks in the park, gardening, or housework can also keep you active.	Un	St	Wa	R	M	Lo	Lo

14	Tip: Physical activity is helpful for a healthy heart and improving your joints and bones. But did you know that physical activity is also good for your mental health and well-being?	Un	St	Wa	R	M	Lo	H
15	Tip: Exercise can increase our self-esteem. Self-esteem is how we perceive ourselves and our self-worth. It is an important indicator of our psychological well-being and our ability to deal with stressors in life	Un	St	Wa	R	M	M	H
16	Tip: Physical activity has great potential to increase our well-being. Just 10 minutes of brisk walking increases our mental alertness, energy, and positive mood.	Un	St	Wa	R	M	Lo	Lo
17	Tip: Regular physical activity can increase our self-esteem and reduce stress and anxiety. It also prevents mental health problems and increases the quality of life for people with mental health problems.	Un	St	Wa	R	M	Lo	H
18	Tip: Exercise can be very effective in reducing stress. Research on working adults has shown that highly active individuals tend to have lower stress levels compared to less active individuals.	Un	St	Wa	R	M	Lo	M
19	Tip: Studies show that the risk for depression and dementia is about 20% to 30% lower in adults who engage in daily physical activity.	Un	St	Wa	R	M	Lo	M
20	Tip: Avoid exercising too close to bedtime. Exercise gives you energy. However slow, relaxing activities like yoga can help you calm down before bed.	Un	St	Wa	R	M	H	H
21	Tip: Stick to quiet activities just before bed, like reading.	Ho	St	Wa	R	H	H	H
35	Tip: It's great that you're active! Physical activity has great potential to increase our well-being. Keep it up!	Un	Ru	Ru	R	H	Lo	H

The following Tasks for Stress Reduction:

Id	Task	Loc.	Min. Mov.	Max. Mov.	Min. Strs.	Max. Strs.	Min. Phys.	Max. Phys.
22	Tip: If the stress goes to your head, try a short relaxation exercise! Close your eyes for a moment and take a deep breath!	Un	St	Wa	M	H	Lo	H
23	Tip: If your day gets too stressful, take a minute and pay specific attention to your breathing, 4-6 breaths per minute is ideal!	Un	St	St	M	H	Lo	H
26	Tip: We are all only human. Sometimes we get tired or overwhelmed when we don't feel well or when things go wrong. When things get too much for you and you feel like you can't cope, ask for help	Un	St	Wa	M	H	Lo	H
27	Tip: Variety is good for your mental health. Variety could be a five-minute break from cleaning your kitchen, a half-hour lunch break at work, or a weekend spent discovering something new!	Un	St	Wa	M	H	Lo	H
28	Tip: A few minutes can be enough to de-stress you. A break, can mean doing nothing, but also exercise. Relax, try yoga or meditation, or just put your feet up!	Un	St	Wa	M	H	Lo	H

29	Tip: Exercise not only increases physical health, but also our self-esteem. Self-esteem is an important indicator of our psychological well-being and our ability to cope with stressors in life	Un	St	Wa	R	M	M	H
30	Tip: Mindfulness helps with several conditions, including stress, depression, addictive behaviors such as alcohol or drug abuse and gambling, and physical problems such as high blood pressure, heart disease and chronic pain.	Un	St	Wa	R	H	Lo	H
31	Tip: Follow a routine. Try to go to bed at the same time every night and wake up at the same time every morning, even on weekends!	Un	St	Wa	R	H	Lo	H
32	Tip: If we don't give ourselves time breaks, stress can build up until we feel too overwhelmed to do anything.	Un	St	Wa	H	H	Lo	H
33	Tip: Exercise can be very effective in reducing stress. Research on working adults has shown that highly active individuals tend to have lower stress levels compared to less active individuals.	Un	St	Wa	R	M	Lo	M
34	Tip: Stick to quiet activities just before bed, like reading.	Ho	St	Wa	H	H	Lo	H
36	Tip: Great that you are active! Exercise can be very effective in reducing stress, keep it up!	Un	Ru	Ru	R	H	Lo	H

Please note that these prompts are translated into English. The actual messages used in the user trials were written in German, as the test runs were conducted in Austria with German-speaking test persons

Notification Scheduling

As previously discussed in 5.2.5, while the scheduling of the notifications during the "Random Phase" was working as expected, the "Deliberate Phase" did not bring any improvements to user response rates
The reason behind this was that the system would only send out messages when a point in time was considered to be "above 50%" probability to invoke a response from a user, using Naive Bayes. The issue with this approach was, that different users would have different base response rates depending on their phone usage habits. So for users who answered most of the notifications, **every** situation was deemed as good, therefore all notifications would be sent out right away in the morning in the deliberate phase, while users who answered under 50% of the messages sent to them might not have received any messages at all, as most situations would be seen as under 50% probability to get a response, therefore a "bad" situation to send a message.
Overall, this would result in worse response rates than when sending the messages out at random. To combat this, a new system to schedule messages was developed. The main idea behind the new system was that messages would be sent out if the context will not allow for an "above 50%" expected response rate, but if the current context allows for a conversion rate which is above average *for the specific user*. This allows for a much more precise and intuitive concept of evaluating a user's context. For example, if a user has a 25 % response rate to notifications overall, but 40% if the user is walking on their daily commute home, this *should* be a good moment to send this user a message. While

a message during this period would be dismissed in the previous system, as it would yield a sub 50% response rate, it will send a message in the new system. The goal is to *improve* the response rate in the second week, not necessarily to push it over 50 percent.
Compared to the previous system with Naive Bayes the new system works in the following way in the Deliberate Phase: Every 10 minutes, the current user-context is fetched from the Variable Aggregator Service by the Notification Manager Service. The Notification Manager Service then requests the History Manager Service, to make an educated decision on whether or not a message should be sent by comparing the current context to historical response data.
This is done by extracting the average response rate for each state of the current context, and then checking how many times (in percent) a message sent when this parameter was active resulted in a response. Just like with Naive Bayes, there are *"black box values"* added (as explained in 5.1.4), to avoid dividing by 0.
But unlike with the Naive Bayes approach, the values are combined in a different way this time. To assess if a message should be sent, first, the probability values (including black box values) are calculated. An example for this could look like this (6.4):

Table 6.3: Example probabilities for responses for parameters in a certain context. (Black box values are in red)

Parameter	Value	MSG Answered	MSG Sent	Response rate
Location	Home	7 + 1	18 + 2	40%
Movement Status	Stationary	3 + 1	11 + 2	30.77%
Weekend/Workday	Weekend	6 + 1	12 + 2	50%
Time of Day	Morning Work	5 + 1	9 + 2	54.55%
Total	-	9 + 1	19 + 2	47.62%

As can be seen in (6.4), a message is currently evaluated where the user context currently consists of being at home and being stationary during the morning of a weekend day. If the average of the response probabilities is built, the result is:

$$(40\% + 30.77\% + 50\% + 54.55\%)/4 = 43.83\%$$

So the system thinks it will get a response from the user with a probability of 43.83%. As can also be seen in (6.4), the system knows the user has an average response rate of 47.62%, as this is higher than 43.83%, no message will be sent (as it is seen as "below

average"). But let's assume the user had a much higher response rate while being non-stationary.

Table 6.4: Example probabilities for responses for parameters (With high response rate while moving). (Black box values are in red)

Parameter	Value	MSG Answered	MSG Sent	Response rate
Location	Home	7 + 1	18 + 2	40%
Movement Status	Moving	9 + 1	11 + 2	76.92%
Weekend/Workday	Weekend	6 + 1	12 + 2	50%
Time of Day	Morning Work	5 + 1	9 + 2	54.55%
Total	-	9 + 1	19 + 2	47.62%

The probability for a user response is:

$$(40\% + 76.92\% + 50\% + 54.55\%)/4 = 55.37\%$$

As the probability 55.37% is larger than 47.62%, a message will be sent out to the user under these conditions, as they are deemed above average for a response.

6.2 Evaluation of second iteration

To evaluate this version of the program, 5 test persons were picked again for the experiment. Two of them were iOS users. Both of them were also enrolled in the first test program. Other than that, three new users were enlisted to test the Android version in parallel. All were connected to the same backend.
Just like in the first iteration, users were tested over a 2-week period, but this time with the new and improved software system. Users were informed what data was tracked for this experiment, but not about the fact that their data from the first week will be used to schedule messages for the second, to avoid users behaviour to adapt to the experiment. The users were first instructed to set up their devices with the software and were then sent into a testing period which - again- lasted two weeks. During this period, data about the users' answering behavior was collected, and during the second week, the collected data was used as a means to individualize the sending out of data to specific users.

6.2.1 Setup for test users

Just like in the first test run, users were asked to set up their Garmin Account first (or re-use their existing one if they already had one) and then register their Garmin Account with the AIT backend to gain their Garmin API ID. A major difference between

setting up the first and the second iteration of the test run was the new Android application

While iOS users could still install the application easily via test flight and have it running properly with ease, Android phones suffer the major downside of users having different battery-saving settings, And different Android Phones work differently in that regard. While the two iOS users were just sent links and instructions on how to set up the application via test flight themselves, setting up the application with the three Android users was done in person with the test persons. After installing the APK File, the user's battery saving and background application settings were set to especially accommodate the experiment. To prevent Android phones from shutting down the application unexpectedly while it was running in the background, different measures had to be taken depending on the phone type of the test person. While some phones would allow the app to run the application in the background with ease, others would not allow this without modifying some settings first. As a guide for setting up the user's phones, the handy guide provided by the *"Don't kill my app!"* website (as previously mentioned in 6.1.2) was used. A summary of the used phones and taken measurements to enable background activity can be seen in the table 6.5. The "DKMA Rating" refers to the rating that the website (*"Don't-kill-my-app-rating"*) gives to phones of that particular phone manufacturer in terms of how aggressively your application will be killed by the phone. This is, of course, good for prolonging battery life, but it is a fake, artificial way of doing so, since it *does* break the functionality of applications, in favor of longer device running time. [74]

As for the two iOS users, an iPhone X and an iPhone SE were used. The app was installed through Apple's TestFlight, But as mentioned before, iPhone battery management behavior stays consistent across phones, so no additional measures had to be taken to enable background tracking.
While Garmin has an application for both Android and iOS, the applications do look the same for both operating systems, so for the Android and iOS setup and the smartwatch of the application the setup process was the same.
After users connected their Garmin account and filled out their initial forms, the test phase started. One thing to note here was that participants did not all start on the same day of the week, but their starting times were spread out over the week. Nevertheless, all participants had their own 2-week period to work with.

6.2.2 Results of the Second Test Run

After conducting the two-week test run, two different ways of evaluating the application were taken. First of all, as the goal was to improve the response rate of users, the response rates of users in the training and deliberate phase were compared, therefore if and how much or little the response rates of users improved.
Secondly, as the participant number of n = 5 was rather low, the decision was made to not do a greater analysis of the participant's data to come to conclusions about the

Brand	Phone Model	DKMA Rating	Measures taken
OnePlus	6	Terrible, second worst rated.	• Lock the application, • Disable Battery Optimization, • Disable Auto-Launch, • Disabling Deep optimization/Sleep standby optimization, • Disabling Deep Clearing of closed apps.
Huawei	P20 Lite	Terrible, third worst rated.	• Enable auto-start of application. • Disable Battery Optimization, • Enable performance mode,
Xiaomi	Pocophone F3	Bad, fourth worst rated.	• Pin application. • Set power plan to performance • Set battery save for the app to 'No restriction'

Table 6.5: Measures for allowing background activity for the application, as published by [74].

benefits and downsides of the system, but rather to have interview-styled discussions with the individual users to gain a better understanding of the test run.

6.2.3 Quantitative Results of User Interaction

First, the difference in how many users responded in time during the first and second weeks will be analyzed. The second test run aimed to build upon the insights gained from the initial experiment and address the technical and usability issues encountered. Its primary focus was to refine and enhance the software system for the final version, based on the lessons learned. As the first test run did not yield any properly useable results for comparing quantitative results as the system had quite unintuitive and ineffective scheduling for messages in the second week, user responses mostly dropped for the second week. As it is unpreferable to have randomly sent out messages beat the scheduling system, the anticipated outcome of the second test run was that users would have higher response rates when the system sent out messages deliberately.
We can see the results of people in their first week during the second test run in the table below 6.6.

Table 6.6: First week with only random messages

PersonId	Total Sent	In time	Not in time	Response rate
Person #1	20	9	11	45%
Person #2	18	11	7	61%
Person #3	20	14	6	70%
Person #4	21	3	14	14%
Person #5	20	11	9	55%

As can be seen in the table above, all participants received between 18 and 21 messages during the first week. Between the participants, there was a wide range of response patterns, with some individuals responding to a high number of messages, others responding moderately, and some responding very minimally. As a result, at least one representative user was selected for every style of answering.
For the second week, **only** deliberate messages were sent out this time. The results for the second week can be found in the table below (6.7).

Table 6.7: Second week with deliberate messages

PersonId	Total Sent	In time	Not in time	Response rate	+/-
Person #1	11	4	7	36%	-9%
Person #2	16	12	4	75%	+18%
Person #3	17	15	2	88%	+18%
Person #4	17	6	11	35%	+21%
Person #5	10	6	4	60%	+5%

As can be seen in 6.7, the amount of messages sent out to users together with their respective likeliness to answer messages is shown. As expected, this time users were sent way fewer messages during the deliberate phase compared to the random phase, as the deliberate phase has 3 notifications as maximum, but might hold back on messages if there are no proper opportunities which will probably not yield any response.
As can be seen above, the last column displays how much more (or less) percent the response rate was during the second phase compared to random messages. For persons who responded to more messages in the first phase than in the deliberate week, this number is red, for all people who had an increase in responses, this number is green.
As is apparent from the table above, **four out of five** people had an increase in their message response rate. For three people, their response rate was about 20% greater than in the first week, and the single person who had a *decrease* in responses, only had a decrease of a few percent.

So from what can be gathered from the small sample size, the system increased the response rate of 80% of the participants, for the one participant who did not increase their response rate, the decrease amount is in the single-digit range percentage-wise. So the new backend fulfilled the intended goal and increased response rates. But to better understand why this is the case, one must look at the data of individual test persons. As response patterns tend to be quite personal, the best way to understand why something worked or why the system did not, is to see how the system responded to the situations of individual users and get a grasp of how it coped with the answers a user presented. It is important to note that these findings are just the result of a pilot study with a really small sample size, and bigger studies need to be performed to have conclusive results.

6.2.4 Individual Results of Test run

In this section, the results of users of the second test run will be looked at individually to gather a greater understanding of certain tendencies in user behavior and contextualize them. After the test run, the user was also individually asked about what they thought about their experience with the application, and they were presented with the tendencies the backend discovered and how they would personally explain those tendencies themselves.

Disclaimer: As the analysis of values will be similar for each person, the same types of tables will be used, with the same labeling of the header lines. The meanings of the data and columns will be explained **only in the section for Person 1**, and will be omitted afterward, as tables will have the same structure for all 5 test persons.

Person 1

In this section, the results from *Person 1* will be presented, who was the only person to have a decrease in their response rate. As their Response Rate decreased by 9%, the person was asked why they thought that their response rate was lower in the second half of the test run. This was explained by the test person having an *unexpected amount of work and stress* during the latter half of the second week, *therefore not checking their phone that often.* During the last few days of the experiment, the response rate even dropped to 0.
Nevertheless, a look into the response likeliness under certain conditions can reveal some tendencies. The table below provides a quick breakdown of the response likelihood of the Person during **different times of their day**:

Info: *Time* refers to the time slot in which that particular person was sent the notification. The *Week 1* refers the the odds that the user would respond to a message sent during that particular "Time" Variable during Week 1. Please note that this also includes the *"black box values"* added (as explained in 5.1.4) values, therefore variables with 0 messages sent still have a value of 0.5.
The first *Sent* column refers to the messages sent where the variable of the current context was the value of that row. The *Week 2* and the second *Sent* columns are the same as their week 1 counterpart. For "Sent" the format is as follows: *Messages sent overall* (+ *Messages only from Week 2*)

Time	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Morning Freetime	0.5	2	0.5	2 (+0)
Morning Commute	0.5	0	0.33	1 (+1)
Morning Work	0.44	7	0.4	8 (+1)
Break	0.5	2	0.67	7 (+5)
Afternoon Work	0.5	2	0.4	3 (+1)
Afternoon Commute	0.66	1	0.4	3 (+2)
Afternoon Freetime	0.375	6	0.33	7 (+1)
Overall	0.45	20	0.42	31

Table 6.8: Person 1: Time data

As can be seen in table **Afternoon Commute** was seen as an above-average moment to send out notifications. As the overall response rate was only 45%, this led to all time-slots that were never used in the first place (in this case only **Morning Commute**) being seen as "above average". As can be seen, the backend therefore attempted to send one message during the **Morning Commute** time-slot in the second week, which made it instantly drop down to 33% response rate as the user did not answer. The scheduler also tried to make use of the **Afternoon Commute** time-slot, as it yielded the - theoretically - best results for the first week. But - as it is the trend with most variables in this table, this did not work out, as there were a very small amount of responses in week 2.
One tendency that was picked up during the initial phase which the backend later sent a lot of messages for was the **Break** time-slot. While it only had a 50% response rate based on two messages, it was still ranked above average. Therefore the backend tried to use it for messages every day - and with success! 5 more messages were sent during the break time slot, resulting in an overall response rate of 67%. This also correlates with many papers that said that during the break on workdays, most people are in their most perceptive state for push notifications.
Secondly, the workday data. This is less complex than the other variable sets since there are only two options: A message can be sent either on a workday or a weekend/holiday. Please note that there was no national holiday in Austria at the time of testing, so the latter only includes Saturdays and Sundays.

Day	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Workday	0.44	14	0.45	20(+6)
Weekend	0.5	6	0.38	11 (+5)
Overall	0.45	20	0.42	31

Table 6.9: Person 1: Weekday data

As can be seen in this table (6.9), the responses during workdays were mostly the same during week 2 with no significant deviation. For weekends, the response probability dropped noticeably, but something important to note here is the small sample size of weekend messages overall, so if the participant stopped responding to messages altogether in the latter half of their test run, and their test run ended with a weekend, these data points will create outliers that are difficult to interpret meaningfully. Nevertheless, there was a fifty-fifty response rate during the first week, which was above average and way better than workday messages.

Unfortunately, location data was invalid for this person as the wrong coordinates were set.

When it comes to Stance data, the following data can be observed:

Stance	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Stationary	0.4375	14	0.38	19 (+5)
Walking	0.5	6	0.5	12 (+6)
Running	0.5	0	0.5	0 (+0)
Overall	0.45	20	0.42	31

Table 6.10: Person 1: Stance data

For stance, one general trend for all test persons can be observed: There was very little data ever captured while people were **running**. The state was recorded many times for many people, but the odds of people randomly getting a message during their weekly run are rather slim. Even if a person goes on a 30-minute run every second day, there is not a high chance of the person randomly receiving a message while running. Therefore, for most test persons the amount of recorded messages during running is 0, and the probability is therefore 50%.
However, what can be observed is a clear tendency to answer messages more easily while being in the **walking** state, rather than the **stationary** stance. The model predicted a 50% chance of response during the first week, which was above average. This streak was continued during the second week, where six more messages were sent out, half of which were answered. While stationary, the backend tried to send fewer messages for the stationary state than for the walking state, as it was seen as less preferable.
While interviewing this person, they answered that they think that walking is likely

better to get a response from them quickly, as walking usually means "not at work", therefore not in a meeting and more able to have a look at the phone.
Another personal feedback by this test person was that they felt the notification content had *very little personal relation* to them. Furthermore, they felt like the *messages were not relevant to their situation* content-wise.

For evaluating the **stress data** of the user, data could unfortunately not be taken into account for scheduling messages, as the Garmin API does not offer live access to stress data, but only gives them out as a collected report at the end of a day. Nevertheless, stress data can be retroactively mapped to the yes/no answers of a user, to check whether or not they responded more under high, medium, or low stress.
For test person 1, one issue with the Garmin stress data became apparent. Garmin is a lot of times quite inconfident in the stress value at one specific point in time for some of the test persons. It is unknown what exactly causes this, but as a result, 13 out of 31 notifications did not have valid stress data attached to them, so only **58%** of the data could be used for evaluation.

Info: *"Total messages"* include messages with both valid and invalid stress data, *"Answered messages"* and *"Unanswered messages"* and their respective *"Average stress for..."* values only include entries that have valid stress data.

	Week 1	Week 2 (with Week 1)
Total Messages	20	31
Invalid Stress Data	8	13
Answered messages	5	7
Average stress for answered	24.2	26.14
Unanswered messages	7	11
Average stress for unanswered	33.43	34.5

Table 6.11: Person 1: Stress data

As can be seen in the table 6.11, only 7 answered messages that could be used for evaluation were recorded in the first week, and 11 were unanswered together with the second.
The first takeaway from this data is that - at least for this person - the Garmin API fails to provide reliable data for stress, even in retrospect. *More than a third* of the data can not be taken into account, due to missing data. This makes the rather small set of samples even smaller.
For the first week, there seemed to be a tendency that the average stress level was about *9%* lower on average when messages were answered, compared to the average value of unanswered messages. This trend also continued in the second week. While the sample size is very small, this could suggest that a Garmin watch registering a lower stress value

could correlate with a higher response rate by the user. Or at least - this particular user.

Person 2

In this section, the results from *Person 2* will be presented, who did increase their response rate by **18%**. During the interview session, the person was asked, why they think their response rate was higher during the deliberate week. They responded by saying that they noticed that messages did have better timings, for example, that they were not really bothered by messages at the workplace anymore, and more during their free time, when they had more time to check their phone. These tendencies are also confirmed when looking at the result data.

Time	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Morning Freetime	0.5	0	0.5	2 (+2)
Morning Commute	0.33	1	0.33	1 (+0)
Morning Work	0.3	8	0.33	8 (+0)
Break	0.66	1	0.83	4 (+3)
Afternoon Work	0.75	2	0.75	6 (+4)
Afternoon Commute	0.5	0	0.5	2 (+2)
Afternoon Freetime	0.875	6	0.846	11 (+5)
Overall	0.6	18	0.66	34

Table 6.12: Person 2: Time data

As can be seen in the data seen in 6.12, There were some clear tendencies regarding time slots the backend identified. First of all, **Morning Commute** and **Morning Work** seemed to yield a very poor response rate of about 0.3, which was half of the average response rate. So no further messages were sent during the second week. However, during break periods and in the afternoon, response rates were much higher for that user. So the backend tries to deploy most new messages during the **Afternoon Freetime** segment, which had a high response rate in the first week, and a comparably high response rate in the second, both resulting in a higher-than-*0.8* response rate. Also, the break period during work was a very effective time slot to get a response from the user.

Day	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Workday	0.64	12	0.71	26(+14)
Weekend	0.5	6	0.5	8 (+2)
Overall	0.6	18	0.66	34

Table 6.13: Person 2: Weekday data

Regarding weekday and workday comparisons, while the user was more likely to not respond during work-related activities, the user was still more likely to respond during workdays as a whole than during weekend days. Furthermore, the weekend response rate did neither increase nor decrease during the second week, even though there were more messages sent during weekends, while the message response rate during workdays increased. This increase exclusive to workdays could be explained due to the user mostly not responding during work-related tasks. This is something the backend detected and emphasized during the second week. However since work-related tasks only happen during workdays and not during the weekend, this is an optimization that could only be used for improving response rates during the workday, not during the weekend.

When it comes to Stance data, the following data can be observed:

Stance	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Stationary	0.5789	17	0.625	30 (+13)
Walking	0.66	1	0.833	4 (+3)
Running	0.5	0	0.5	0 (+0)
Overall	0.6	18	0.66	34

It can be seen that stationary response rates are increased, which is mostly due to the messages being shifted to non-work time slots, therefore increasing the overall response rate. What can also be observed is that there was a tendency to get more responses during walking. This seems to be a general tendency among all participants, since **"Walking"** does not only mean the person is walking, but since the walking state is triggered by the motion sensors of the phone, it means that the phone is not lying around somewhere charging or stored on some backpack which lays still on the ground, but it has to be on the user themselves to trigger a motion state.

Location	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Unknown	0.66	1	0.833	4 (+3)
Work	0.42	5	0.42	5 (+0)
Home	0.64	12	0.66	25 (+13)
Overall	0.6	18	0.66	34

A very strong tendency the system noticed is that the location mattered a lot for this user. The user had a very poor response rate while being at **Work**, but had an above average response rate for being at **Home** or anywhere else than work. The system stopped sending out messages while the user was at their workplace completely as a consequence of this. This was a noticeable change in the second week according to the user.

	Week 1	Week 2 (with Week 1)
Total Messages	18	34
Invalid Stress Data	3	6
Answered messages	10	20
Average stress for answered	38.55	43.05
Unanswered messages	5	8
Average stress for unanswered	17.3	28.93

Table 6.14: Person 2: Stress data

In the table 6.14 the stress data of person 2 can be seen. From the data, it can be seen that this person - just like Person 1 - had some points of data that resulted in invalid stress data. But unlike person 1, there was much more useable data, as only 6 points of stress data were invalid. It is unclear why there is such a huge gap in invalid stress data between users.
Interestingly enough, the existing data set suggests that this person does have a higher response rate while being under stress, while unanswered messages are typically tied to little stress. This is the complete opposite of Person 1. This supports the statement that the response behavior of users is very unique for each person and notification scheduling should be tailored for each user, as it is hard to make a general statement under which condition users will respond.

Person 3

Person 3 had an already quite high response rate of **70%** in the first week, but an even higher response rate of **88%** during the second week, with only **two unanswered** messages in the second week, whereas **six** were unanswered in the first week.
This increase can be credited to the backend noticing poor response rates when the test person was at work, therefore avoiding sending messages to the test person while the location of the user appeared as **Work**. Also, the person themselves stated that they think that their work time is a very bad time of day to respond to messages, but the time immediately after is very good. They also stated that they found the messages quite motivating and felt the timings of the messages were noticeably better during the second week.

Time	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Morning Freetime	0.5	0	0.66	1 (+1)
Morning Commute	0.66	1	0.85	5 (+4)
Morning Work	0.33	4	0.33	4 (+0)
Break	0.5	0	0.66	1 (+1)
Afternoon Work	0.63	9	0.63	9 (+0)
Afternoon Commute	0.66	1	0.89	7 (+6)
Afternoon Freetime	0.85	5	0.75	10 (+5)
Overall	0.68	20	0.77	37

Table 6.15: Person 3: Time data

When it comes to time slots, there were similar tendencies as for Person 2, where the system identified that it is not a good idea to send messages during work times. Afternoon work turned out to have a 63% response rate, but it was still deemed to be ”below average”, therefore the system did not send any more messages in the second week in this time slot. Other than that, every other time slot did receive more messages in the second week and the respective response rates were increased.

Day	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Workday	625	14	0.76	28 (+14)
Weekend	0.75	6	0.72	9 (+3)
Overall	0.68	20	0.77	37

Table 6.16: Person 3: Weekday data

For their weekday data, there was no significant difference between response rates of work- or weekdays.

Stance	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Stationary	0.63	17	0.78	31 (+14)
Walking	0.8	3	0.63	6 (+3)
Running	0.5	0	0.5	0 (+0)
Overall	0.68181818181	20	0.76923076923	37

During the first week, it seemed like there was a tendency to get better response rates when the test person was walking. Unfortunately, this could not be replicated during week 2. Walking was most likely seen as more promising than it actually was due to the small sample size. Another thing that could have happened is that because the response rate for the walking state was so high in week 1, in week 2 unfavorable states would have been classified as ”okay” if the walking status was active, which leads to a decrease in response rates.

Location	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Unknown	0.83	4	0.81	9 (+5)
Work	0.45	9	0.45	9 (+0)
Home	0.77	7	0.86	19 (+12)
Overall	0.68181818181	20	0.76923076923	37

For location, there was a clear tendency for this test person. A thing that was already apparent from the time slot probabilities for this person is that they usually did not respond during work times. This is also reflected in the location data of this user, as being at work was associated with a below-average response rate compared to being at any other location. This led to no new messages being sent out while the user was physically at their workplace.

	Week 1	Week 2 (with Week 1)
Total Messages	20	37
Invalid Stress Data	11	19
Answered messages	7	16
Average stress for answered	26.64	31.43
Unanswered messages	2	2
Average stress for unanswered	25.5	25.5

Table 6.17: Person 3: Stress data

Due to the high amount of invalid stress data, only two of the unanswered messages had usable data for the stress evaluation. From the small sample size that has evaluable stress data, there does not seem to be a huge difference between the stress data for answered and unanswered messages, as they only have a discrepancy of about 6 Garmin stress points. There seems to be a slight tendency that the stress for answered messages went up during the second week, but this is most likely due to the small sample size.

Person 4

Person 4 is quite interesting, as their behavioral patterns for responding to messages are quite different than other users, as they tended to respond to messages more easily during their working time than their leisure time. This is a good example of why message scheduling can benefit from a personalized rule set for each user rather than having general rules for when users should receive messages.
Furthermore, the person had generally a really low response rate. During the first week, only 14% of the messages were answered by that person. During the second week, 35% were answered in time. It is important to note here this person in particular did answer many messages after the 45-minute threshold, but as for this test the time limit for reacting to messages ”in-time” was set to 45 minutes, Nevertheless, the response rate was increased by 19%.

Time	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Morning Freetime	0.33	1	0.4	3 (+2)
Morning Commute	0.2	3	0.28	5 (+2)
Morning Work	0.4	3	0.36	9 (+6)
Break	0.4	3	0.42	5 (+2)
Afternoon Work	0.25	6	0.3	8 (+2)
Afternoon Commute	0.5	0	0.2	3 (+3)
Afternoon Freetime	0.14	5	0.14	5 (+0)
Overall	0.17391304347	21	0.25	38

Table 6.18: Person 4: Time data

For this person in particular, sending messages during the **MORNING WORK** or during the **BREAK** of the work time was by far the best option. While the response rates for these two work times stayed mostly the same during the second week, sending messages during other time slots, for example, **AFTERNOON FREETIME** was avoided. This improved the overall response rate.

Day	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Workday	0.235	15	0.3	28 (+13)
Weekend	0.125	6	0.17	10 (+4)
Overall	0.17391304347	21	0.25	38

Table 6.19: Person 4: Weekday data

Furthermore, this person showed a significant response rate difference between weeks and workdays. The person responded more easily when being at work during the week. And also for a whole week, it was easier for the person to respond to messages during a workday than on a day off.

Stance	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Stationary	0.17391304347	21	0.25	38
Walking	0.5	0	0.5	0
Running	0.5	0	0.5	0
Overall	0.17391304347	21	0.25	38

For this person, not a single message was sent when the person was in a moving position. During the interview with this test person, they stated that they usually do not have their phone on them, but positioned on their table or somewhere else. This might also explain why they seemed to be generally more unavailable for notifications on their phone than other test persons which had a much higher response rate overall.

	Week 1	Week 2 (with Week 1)
Total Messages	20	30
Invalid Stress Data	11	19
Answered messages	2	5
Average stress for answered	15.75	17.9
Unanswered messages	8	12
Average stress for unanswered	30.5	33.83

Table 6.20: Person 4: Stress data

For this person in particular it seemed that the stress level during successful responses was way lower than for unanswered messages. Unfortunately due to the small sample size, and many messages sent when there was invalid stress data, this observation might not say a lot about the general facts.

Person 5

Person 5 only showed a slight improvement in their response rate. During the first week, 55% of the messages were answered. During the second week, 60% of the notifications resulted in a response in time. The person stated during their retrospect interview session that they felt that the content of the messages did not feel useful. However, during the first week, they felt like they were bombarded by messages and felt relieved that fewer messages were sent during the second week. As the main reason for not responding in time, they stated that having other stuff to do at the time a message reached them was the main reason for not responding in time.

Time	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Morning Freetime	0.5	0	0.66	1 (+1)
Morning Commute	0.5	0	0.33	1 (+1)
Morning Work	0.38	11	0.38	11 (+0)
Break	0.5	0	0.5	0 (+0)
Afternoon Work	0.6	3	0.5	4 (+1)
Afternoon Commute	0.5	0	0.66	4 (+4)
Afternoon Freetime	0.75	6	0.72	9 (+3)
Overall	0.54545454545	20	0.5625	30

Table 6.21: Person 5: Time data

A quite noticeable detail for the time slot data of this person is that they received zero messages during **Morning Freetime**, **Morning Commute**, **Break** and **Afternoon Freetime**. This is mostly due to the person setting their time slots to a really short amount of time, making it really unlikely for a random message to be scheduled for these time slots. Other than that, the person was much more likely to respond during the

afternoon. The backend only sent two more messages which were not in the afternoon in the second week, while sending 8 more in the afternoon, which resulted in a slightly better response rate. An additional benefit was that only half as many messages were sent overall, so the person was bothered less often.

Day	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Workday	0.56	14	0.59	20 (+6)
Weekend	0.5	6	0.5	10 (+4)
Overall	0.54545454545	20	0.5625	30

Table 6.22: Person 5: Weekday data

Just like for most other test persons, there was no significant difference between workday and weekday messages. Workdays seemed to be better, but only by a very small amount.

Stance	Week 1	Sent	Week 2 (with Week 1)	Sent (with Week 1)
Stationary	0.52	19	0.54	29 (+10)
Walking	0.5	0	0.5	0
Running	0.66	1	0.66	1
Overall	0.54545454545	20	0.5625	30

Just like for work and weekdays, there does not seem to be a significant difference between response rates during different stances. Running seems to have a higher response rate, but there was only a single message captured with the person running, so there is no way to make a meaningful conclusion for this data point.

	Week 1	Week 2 (with Week 1)
Total Messages	20	30
Invalid Stress Data	7	8
Answered messages	6	12
Average stress for answered	20.5	24.29
Unanswered messages	7	10
Average stress for unanswered	38.43	37.0

Table 6.23: Person 5: Stress data

Just like for *Person 4*, there seems to be a tendency, that the person was under more stress when not responding to messages. Again, this observation is not backed by the data of every test person, and again, the sample size captured over two weeks is quite low. And as it is with the nature of Garmin stress data, a lot of it is not evaluable. While there seems to be a tendency for certain users to correlate with their stress level and their response rates, further testing must be done to make a definitive conclusion about this.

6.2.5 Results

The second test run was more successful than the first. While the first test run failed to improve response rates for most users, the second iteration managed to improve the response rate of 4 out of 5 users. The fifth test person for whom they could not find any improvements only had a decrease of 9 percent, with the test person admitting that they had a lot more stress during their second week compared to their first. This caused the test person not to really look at their phone anymore for the second week, which probably could not have been changed by smarter scheduling of messages.
Nevertheless, many parameters suffered from the small sample size, for example, the weekend parameter was only recorded for a few messages, which leads to really inconclusive data. Also, the Garmin API only sometimes recorded useful data, and even when it was not faulty, the data would only show a user to be in a calm state. To have a wide range of data for users being very stressed, calm, under a small amount of stress, and so on. The way things turned out for this particular test run, it is quite hard to make conclusions about outlier values that only occasionally appeared.

6.3 Conclusion

While the testruns were both only conducted over a small amount of time, they both gave some useful insight into people's behavior about answering notifications and the potential benefits of scheduling messages for individual users. Some ideas and information that already appeared in other papers were supported by this study (for example that the midday work break is usually a good point in time to send notifications to users). Other new experiential ideas - like using stress data for scheduling notifications - were not explored in-depth to make a full conclusion about it's usefulness, but some information was gained about what developers of a future research program that aims to accomplish a similar scheduling technique should be aware of and what to avoid. Furthermore, even though the test run had a rather small sample size, it seems like user behavior regarding notification answering is quite individual, and scheduling systems should definitely take individual preferences and tendencies into account if they want to send out messages to users during times when they can respond.
One thing to keep in mind when going through the analysis of these test runs is - again - to always keep the small sample size in mind.

CHAPTER 7

Discussion

7.1 Applicable scheduling information

7.1.1 Generally applicable scheduling information

One piece of information that constantly came up during research is that the work break time is a good time slot to respond to messages in general. While this is not true for every single person, since every person and every workplace situation is individual, this information was also supported by the research of this paper, as most people which were taken into account for this study had 12:00 (or the midday-break times set by a user) as a good time for sending messages to them. While this can vary from person to person, it shows that even if scheduling is not personalized, there are some generally applicable scheduling guidelines that can be used to get a better response rate from users over random scheduling. So even if there is no system in place to personalize notification scheduling, it can still be beneficial to follow some general rules for scheduling messages, rather than sending them at a random point in time.
Furthermore, this can most likely be taken further if some information about the user base exists. For example, people without a regular nine-to-five job (students, people working night shifts...) will most likely have a different response behavior. So if an application has a specific user base, notification timings can be tailored without personalizing the timings for each individual user.
It is important to note, however, that this is only applicable for predicting the average behavior of large groups of people. There is no way to make statements about the response behavior of an individual with such broad statements.

7.1.2 Individualized scheduling information

As seen in the research of this and also other papers, different parameters that do affect the response rate of a user might have very different implications for different human

beings. For example, there is an observable difference in response likeliness for people who are walking and people whose phones claim they are standing still. But there is no way to make a statement that applies to every human whether or not standing or walking gives better response results. Especially considering what a phone's sensor detects is not whether or not the person is moving, but whether or not the phone itself moves, which might have - again - different implications for different people. If a person has their phone on their body at all times, a detected "stationary pose" will actually mean a person is sitting, while it could also mean that the phone is just lying stationary on a table by itself otherwise.
So for some people, a "moving" phone will just mean they are carrying their phone on their body which leads to a high response rate, for others it means they are doing some sports activity and are absolutely unavailable. The same goes for being at a workplace. Some people turn their phones on silent mode during their work, so sending messages will just lead to clogging their notification center when they have a look at their phone afterward, and it will probably just be removed by the "clear all notifications" button without being read. While other people might behave in the exact opposite way and basically do not reach for their smartphone in their free time, but will have a look at their phone every once in a while being at the office desk.

7.2 Information which needs further inspection

While definitive conclusions are hard to draw from this pilot study, there are some aspects of the scheduling strategy that will need further inspection in the future with a bigger sample size. The main goal of using the Garmin watches to schedule information did unfortunately not bring definitive information on whether or not this parameter should be taken into account for scheduling notification messages. First of all, from the information that surfaced in this study, it seems that most persons seem to be at a stagnant stress level throughout their day, only having spikes every once in a while. While stress measurements could potentially give us a metric on which to decide on the potential response rate of a user, this would need a great number of samples drawn from multiple months if the messages are sent out at random, as it is quite unlikely to get many messages sent during the stressing situations, as they are quite sparse. So not only should the amount of participants be increased for this, but also the time period in which this is tested. Also, since Garmin provided a lot of data sets it deemed "erroneous" for stress data, this will either need another way of measurement which provides less unevaluable data, or just twice as many measurements to compensate for the unusable data.
Furthermore, while the Garmin version can give us information about the likelihood of response of users during different stress situations, as of today, the Garmin API does not allow for live polling of data, so this can only be evaluated in retrospect, not used for live scheduling. For a further study, it would be meaningful to investigate different ways of consumer-grade fitness trackers to find one that is best suited for this case. For further studies, stress-measuring devices would need to fulfill the following criteria:

Features	Description
Live Data Query	Data should be available to be accessed by the scheduler at any moment to make an educated decision on the user's current context.
Reliable Stress Data	The measuring of the stress data of participants should be reliably and consistently measured, without a significant amount of them being invalid.
Physical Activity Data	When it comes to physical activity data, it should be reliable just like the stress data. Furthermore, since fitness trackers usually prevent not just one metric from measuring physical activity, not only the tracker but also the variables taken from it to evaluate physical activity should be picked carefully.

7.3 System Design Improvements

While the test run was performed on a small number of people, for a larger sample size, it is essential to ensure the system will handle the amount of test persons as well. This should be ensured from a software design perspective, as well as the platform it is hosted on.
While the system used for this paper mainly used Naive Bayes and its derivatives for scheduling its messages, it would also be beneficial to try out other algorithms as well in the future. While Naive Bayes in this experiment seemed to have results that could give the desired effect, it is important to keep other options in mind as well.

CHAPTER 8

Conclusion and Future Work

This paper made a first effort to investigate the link between various context variables of an individual, and their correlation to a user's response rate to push notifications. While the focus of the study was mainly on the timing and response rate of any push notifications, the actual content of the messages was not the main focus of the experiments.
While some pre-existing research data can also be confirmed by this work (such as the lunch break being a good time to receive notifications for most individuals), the efforts for new knowledge - such as the effect of stress and physical activity on the response rate of a user - were mostly inconclusive. While the efforts of measuring stress and physical activity were limited to consumer-grade Garmin products, the API did unfortunately not offer the functionalities needed for scheduling live notifications, and furthermore, even with the data evaluated in retrospect, there does not seem to be conclusive evidence or enough data from the small sample size and time period the user information was recorded.
Nevertheless, a scaleable distributed system made up of several microservices was created as part of this pilot study, which paves the way for a potential follow-up study with more participants and a longer run time with an advanced server structure.
While only used on a small set of persons, the second iteration of the scheduling system looks promising and could improve the response rate of users over a small time period. It is yet to be seen if this process also works with a greater variety of persons and for a much bigger sample size.
What seems to be the conclusion - at least from the small sample size that this experiment has - it seems to be very apparent that different parameters have to be interpreted very differently for different human beings (for example whether or not a user will respond to messages while being at work will depend heavily on the person and their job). Therefore it is important to keep researching on how to individualize scheduling for each person, instead of making common assumptions about what the average response behavior of people might be.

8.1 Research Questions

Research Question 1: Which parameters of a user's state should be considered when evaluating proper timings of notifications?

Based on the current state of the art, most papers resort to assessing whether or not the user is currently at work and the position of the user. As an additional parameter, the stance (walking, sitting...) of a user is measured as well using the accelerometer of the phone. As one of the most important factors, the time of day influences the responsiveness of a user a lot. What is important here is that the meaning of different times of day might vary between people. For example, most people are quite fast to respond during their lunch break, but for some people, this might be at 12 o'clock, while others only start their break at 14:30.
From the testing done for this thesis, these parameters seem to be good indicators of the responsiveness of users. What is important however is to contextualize the times of day for each user in some way, as each person has a different schedule and wakes up and goes to sleep at different times of the day. Also, the tracked stance of a user's phone can make a great difference in whether or not users will respond. However, this might not always be due to the movement of a user indicating availability, but rather the absence of movement on a user's phone indicating that it is lying around somewhere and the user will not respond if a message is sent right now.
The differentiation between off days and workdays did not seem to be important for the user group consulted for this thesis.
As additional uncommon parameters, the stress and physical activity rates tracked by a Garmin fitness tracker were used. The next section goes into more detail regarding this.

Research Question 2: Can the notification response rate be improved with individualized message timings by tracking stress?

Unfortunately, this thesis could not achieve a conclusive answer to this research question. This is due to the small sample size of test persons, unreliable API data, and other technicalities of using the Garmin API in particular. When reviewing the stress data of participants in retrospect, it seemed like there could be tendencies for users to respond differently when certain stress levels were hit, but unfortunately, the amount of messages sent with usable stress data is rather low, meaning that more data needs to be collected in the future to get conclusive evidence on this.
Nevertheless, the software project lays the groundwork for testing the usefulness of stress data for a bigger pool of participants in the future. The software system could also be repurposed in the feature to compare and contrast stress data of different fitness tracker providers in the future.

Research Question 3: How should a prototype that implements adaptive notification scheduling look in order to increase the user response rate?

One of the most important observations during the testing and interviewing phase of this project is that there seems to be a link between at least some of the tracked user variables (time of day, stance), and the responsiveness of users. What is important to note, however, is that the interpretation of context variables of a user must be done individually. For some people receiving a message at 6 in the morning will often result in a response, for others, it will be a bad time. So while it is possible to judge the likelihood of a response by a user based on their current context, it is also important to design a system in a way that it can adapt to the individual likelihoods of each user, as the same context can mean very different things for different users.

8.2 Future Directions

In conclusion, this paper has opened up numerous possibilities on how user response rate might be improved using different variables for scheduling. While the pilot study does not provide definitive results (due to a small sample size and short testing period) by itself, it does offer a starting point and a base software for both the front- and back of the system to conduct a greater study and theorize on further possible conclusions regarding fitting context parameters and useful live extraction of health and fitness variable information of users. It is expected that more data will eventually be found in this field, to finally get a greater understanding about effective, user-tailored scheduling algorithms for notifications, and especially the impact of biometric data on the response rate of a user. Another possibility that opened up in recent years due to the rapid development of generative AI systems is to tailor notification content to users. While this thesis mostly focuses on the timing of notification messages, it also attempts to narrow down which notification to send under certain conditions. For future developments of notification scheduling systems, it might not only be relevant when a notification is sent but to also tailor the message content to the situation of a user. While this was already done to some degree by providing a variety of notification text for any given situation a user can be in, this could potentially be tailored further by using generative AI to adapt messages to the current context of a user. For example, messages could be converted to be written in a more calming way for users who are experiencing high stress. Another option would be to just generate messages from scratch based on the user's context.
Other than that, for a better understanding of the link between biometric signals and a user's response rate, it will be important for future research to perform similar studies to the one performed for this thesis, and test out different products to measure biometric signals. As the chosen fitness trackers for this thesis did not really bring the desired benefit, it will be important to also consider other ways of measuring stress and physical activity, and compare and contrast different consumer-grade fitness trackers to see if they can be used for notification scheduling.

List of Figures

List of Tables

Bibliography

[1] M. Pielot, K. Church, and R. de Oliveira, „An in-situ study of mobile phone notifications“, Sep. 2014, pp. 233–242. DOI: 10.1145/2628363.2628364.

[2] *Teens are exhausted by phone notifications but don't know how to quit, report finds*, Available at https://edition.cnn.com/2023/09/26/health/teen-hundreds-of-phone-notifications-report-wellness/index.html. Accessed: 16.6.2024.

[3] *Study: Average teen received more than 200 app notifications a day*, Available at https://www.michiganmedicine.org/health-lab/study-average-teen-received-more-200-app-notifications-day. Accessed: 16.6.2024.

[4] *Push Notifications Statistics (2024)*, Available at https://www.businessofapps.com/marketplace/push-notifications/research/push-notifications-statistics/. Accessed: 16.6.2024.

[5] H. Oh, L. Jalali, and R. Jain, „An intelligent notification system using context from real-time personal activity monitoring“, in *2015 IEEE International Conference on Multimedia and Expo (ICME)*, 2015, pp. 1–6. DOI: 10.1109/ICME.2015.7177508.

[6] T. Okoshi, K. Tsubouchi, M. Taji, T. Ichikawa, and H. Tokuda, „Attention and engagement-awareness in the wild: A large-scale study with adaptive notifications“, in *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2017, pp. 100–110. DOI: 10.1109/PERCOM.2017.7917856.

[7] F. Künzler, „Assessing and Predicting States of Receptivity for Physical Activity Interventions“, Ph.D. dissertation, ETH Zurich, 2020.

[8] V. Mishra, F. Künzler, J.-N. Kramer, E. Fleisch, T. Kowatsch, and D. Kotz, „Detecting receptivity for mHealth interventions in the natural environment“, *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies*, vol. 5, no. 2, pp. 1–24, 2021.

[9] K. Wunsch, T. Eckert, J. Fiedler, and A. Woll, „Just-in-time adaptive interventions in mobile physical activity interventions—A synthesis of frameworks and future directions“, *The European Health Psychologist*, vol. 22, no. 4, pp. 834–842, 2022.

[10] V. Mishra, F. Künzler, J.-N. Kramer, E. Fleisch, T. Kowatsch, and D. Kotz, „Detecting Receptivity for mHealth Interventions“, *GetMobile: Mobile Computing and Communications*, vol. 27, no. 2, pp. 23–28, 2023.

[11] I. Nahum-Shani, D. W. Wetter, and S. A. Murphy, „Adapting just-in-time interventions to vulnerability and receptivity: Conceptual and methodological considerations“, in *Digital therapeutics for mental health and addiction*, Elsevier, 2023, pp. 77–87.

[12] I. Nahum-Shani, S. N. Smith, B. Spring, L. Collins, K. Witkiewitz, A. Tewari, and S. Murphy, „Just-in-Time Adaptive Interventions (JITAIs) in Mobile Health: Key Components and Design Principles for Ongoing Health Behavior Support“, *Annals of Behavioral Medicine: A Publication of the Society of Behavioral Medicine*, vol. 52, pp. 446–462, 2018.

[13] G. Alkhaldi, F. L. Hamilton, R. Lau, R. Webster, S. Michie, and E. Murray, „The effectiveness of prompts to promote engagement with digital interventions: a systematic review“, *Journal of medical Internet research*, vol. 18, no. 1, e6, 2016.

[14] *Lifesum App*, Available at `https://lifesum.com`. Accessed: 23.10.2021.

[15] M. Kim, *The effects of external cues on media habit and use: Push notification alerts and mobile application usage habits*. Michigan State University, 2014.

[16] L. Dennison, L. Morrison, G. Conway, and L. Yardley, „Opportunities and Challenges for Smartphone Applications in Supporting Health Behavior Change: Qualitative Study“, *J Med Internet Res*, vol. 15, no. 4, e86, Apr. 2013, ISSN: 14388871. DOI: `10.2196/jmir.2583`. [Online]. Available: `http://www.ncbi.nlm.nih.gov/pubmed/23598614`.

[17] W. Choi, S. Park, D. Kim, Y.-k. Lim, and U. Lee, „Multi-Stage Receptivity Model for Mobile Just-In-Time Health Intervention“, vol. 3, no. 2, Jun. 2019. DOI: `10.1145/3328910`. [Online]. Available: `https://doi.org/10.1145/3328910`.

[18] K. Anderson, O. Burford, and L. Emmerton, „Mobile health apps to facilitate self-care: a qualitative study of user experiences“, *PloS one*, vol. 11, no. 5, e0156164, 2016.

[19] T. Peters, „The role of motivation in consumer experience with mobile health (mHealth) applications“, 2022.

[20] E. Goodwin and T. Ramjaun, „Exploring consumer engagement in gamified health and fitness mobile apps“, *Journal of Promotional Communications*, vol. 5, no. 2, 2017.

[21] I. Vaghefi, B. Tulu, *et al.*, „The continued use of mobile health apps: insights from a longitudinal study“, *JMIR mHealth and uHealth*, vol. 7, no. 8, e12983, 2019.

[22] P. R. Palos-Sanchez, J. R. Saura, M. A. Rios Martin, and M. Aguayo-Camacho, „Toward a better understanding of the intention to use mHealth apps: exploratory study“, *JMIR mHealth and uHealth*, vol. 9, no. 9, e27021, 2021.

[23] A. S. Mustafa, N. Ali, J. S. Dhillon, G. Alkawsi, and Y. Baashar, „User engagement and abandonment of mHealth: a cross-sectional survey“, in *Healthcare*, MDPI, vol. 10, 2022, p. 221.

[24] H. O. Woldeyohannes and O. K. Ngwenyama, „Factors influencing acceptance and continued use of mHealth apps“, in *HCI in Business, Government and Organizations. Interacting with Information Systems: 4th International Conference, HCIBGO 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings, Part I 4*, Springer, 2017, pp. 239–256.

[25] W. Peng, S. Kanthawala, S. Yuan, and S. A. Hussain, „A qualitative study of user perceptions of mobile health apps“, *BMC public health*, vol. 16, pp. 1–11, 2016.

[26] R. Jakob, S. Harperink, A. M. Rudolf, E. Fleisch, S. Haug, J. L. Mair, A. Salamanca-Sanabria, and T. Kowatsch, „Factors influencing adherence to mHealth apps for prevention or management of noncommunicable diseases: systematic review“, *Journal of Medical Internet Research*, vol. 24, no. 5, e35371, 2022.

[27] J. L. Mair, L. D. Hayes, A. K. Campbell, D. S. Buchan, C. Easton, and N. Sculthorpe, „A personalized smartphone-delivered just-in-time adaptive intervention (JitaBug) to increase physical activity in older adults: mixed methods feasibility study“, *JMIR formative research*, vol. 6, no. 4, e34662, 2022.

[28] L. G. Morrison, C. Hargood, V. Pejovic, A. W. A. Geraghty, S. Lloyd, N. Goodman, D. T. Michaelides, A. Weston, M. Musolesi, M. J. Weal, and L. Yardley, „The Effect of Timing and Frequency of Push Notifications on Usage of a Smartphone-Based Stress Management Intervention: An Exploratory Trial“, *PLOS ONE*, vol. 12, pp. 1–15, Jan. 2017. DOI: 10.1371/journal.pone.0169162. [Online]. Available: https://doi.org/10.1371/journal.pone.0169162.

[29] S. Aminikhanghahi, R. Fallahzadeh, M. Sawyer, D. J. Cook, and L. B. Holder, „Thyme: Improving Smartphone Prompt Timing Through Activity Awareness“, in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017, pp. 315–322. DOI: 10.1109/ICMLA.2017.0-141.

[30] A. Visuri, N. van Berkel, T. Okoshi, J. Goncalves, and V. Kostakos, „Understanding smartphone notifications' user interactions and content importance“, *International Journal of Human-Computer Studies*, vol. 128, pp. 72–85, 2019, ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2019.03.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581919300205.

[31] X.-L. Pham, T.-H. Nguyen, W.-Y. Hwang, and G.-D. Chen, „Effects of Push Notifications on Learner Engagement in a Mobile Learning App“, in *2016 IEEE 16th International Conference on Advanced Learning Technologies (ICALT)*, 2016, pp. 90–94. DOI: 10.1109/ICALT.2016.50.

[32] S. Aminikhanghahi, R. Fallahzadeh, M. Sawyer, D. J. Cook, and L. B. Holder, „Thyme: Improving Smartphone Prompt Timing Through Activity Awareness“, in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017, pp. 315–322. DOI: 10.1109/ICMLA.2017.0-141.

[33] *About Garmin*, Available at https://www.garmin.com/en-US/company/about-garmin/. Accessed: 8.9.2021.

[34] *Fitness Watches*, Available at https://www.garmin.com/en-US/c/wearables-smartwatches/. Accessed: 8.9.2021.

[35] *Garmin Connect*, Available at https://connect.garmin.com/. Accessed: 8.9.2021.

[36] *What Is the Stress Level Feature on My Garmin Watch?*, Available at https://support.garmin.com/en-US/?faq=WT9BmhjacO4ZpxbCc0EKn9. Accessed: 8.9.2021.

[37] *Garmin Connect Developer Program*, Available at https://developer.garmin.com/gc-developer-program/overview/. Accessed: 8.9.2021.

[38] *Apple Core Motion*, Available at https://developer.apple.com/documentation/coremotion. Accessed: 7.9.2021.

[39] *Activity Recognition API*, Available at https://developers.google.com/location-context/activity-recognition. Accessed: 7.9.2021.

[40] N. Bidargaddi, D. Almirall, S. Murphy, I. Nahum-Shani, M. Kovalcik, T. Pituch, H. Maaieh, and V. Strecher, „To prompt or not to prompt? A micro-randomized trial of time-varying push notifications to increase proximal engagement with a mobile health application (Preprint)“, *JMIR mHealth and uHealth*, vol. 6, 2018. DOI: 10.2196/10123.

[41] T. Leech, D. Dorstyn, A. Taylor, and W. Li, „Mental health apps for adolescents and young adults: A systematic review of randomised controlled trials“, *Children and Youth Services Review*, vol. 127, p. 106 073, 2021, ISSN: 0190-7409. DOI: https://doi.org/10.1016/j.childyouth.2021.106073. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0190740921001523.

[42] S. Cohen, T. Kamarck, and R. Mermelstein, „A Global Measure of Perceived Stress“, *Journal of Health and Social Behavior*, vol. 24, no. 4, pp. 385–396, 1983, ISSN: 00221465. [Online]. Available: http://www.jstor.org/stable/2136404.

[43] A. Goyal, S. Singh, D. Vir, and D. Pershad, „Automation of Stress Recognition Using Subjective or Objective Measures“, *Psychological Studies*, vol. 61, no. 4, pp. 348–364, Dec. 2016, ISSN: 0974-9861. DOI: 10.1007/s12646-016-0379-1. [Online]. Available: https://doi.org/10.1007/s12646-016-0379-1.

[44] M. L. González Ramírez, J. P. García Vázquez, M. D. Rodríguez, L. A. Padilla-López, G. M. Galindo-Aldana, and D. Cuevas-González, „Wearables for Stress Management: A Scoping Review“, *Healthcare*, vol. 11, no. 17, 2023, ISSN: 2227-9032. DOI: 10.3390/healthcare11172369. [Online]. Available: https://www.mdpi.com/2227-9032/11/17/2369.

[45] A. O. Akmandor and N. K. Jha, „Keep the Stress Away with SoDA: Stress Detection and Alleviation System“, *IEEE Transactions on Multi-Scale Computing Systems*, vol. 3, no. 4, pp. 269–282, 2017. DOI: 10.1109/TMSCS.2017.2703613.

[46] V. K. Sharma, M. Trakroo, V. Subramaniam, M. Rajajeyakumar, A. B. Bhavanani, and A. Sahai, „Effect of fast and slow pranayama on perceived stress and cardiovascular parameters in young health-care students“, *International journal of yoga*, vol. 6, no. 2, pp. 104–110, 2013.

[47] M. A. Stults-Kolehmainen, „The interplay between stress and physical activity in the prevention and treatment of cardiovascular disease“, *Frontiers in physiology*, vol. 4, p. 74 131, 2013.

[48] S. Gedam and S. Paul, „A review on mental stress detection using wearable sensors and machine learning techniques“, *IEEE Access*, vol. 9, pp. 84 045–84 066, 2021.

[49] V. Pejanović and M. Radaković, „CLASSIFICATION OF COGNITIVE STRESS AS A PSYCHOLOGICAL INDICATOR THROUGH MACHINE LEARNING“, *SCIENCE International Journal*, vol. 3, no. 1, pp. 139–143, 2024.

[50] O. AlShorman, M. Masadeh, M. B. B. Heyat, F. Akhtar, H. Almahasneh, G. M. Ashraf, and A. Alexiou, „Frontal lobe real-time EEG analysis using machine learning techniques for mental stress detection“, *Journal of integrative neuroscience*, vol. 21, no. 1, p. 20, 2022.

[51] M. Naegelin, R. P. Weibel, J. I. Kerr, V. R. Schinazi, R. La Marca, F. von Wangenheim, C. Hoelscher, and A. Ferrario, „An interpretable machine learning approach to multimodal stress detection in a simulated office environment“, *Journal of Biomedical Informatics*, vol. 139, p. 104 299, 2023.

[52] R. K. Nath, H. Thapliyal, and A. Caban-Holt, „Machine learning based stress monitoring in older adults using wearable sensors and cortisol as stress biomarker“, *Journal of Signal Processing Systems*, pp. 1–13, 2022.

[53] B. Nachenahalli Bhuthegowda, A. Pande, and D. Mishra, „Next-Gen Stress Monitoring: Social Robot and AI Integration“, in *International Conference on Human-Computer Interaction*, Springer, 2024, pp. 87–98.

[54] T. Nijhawan, G. Attigeri, and T. Ananthakrishna, „Stress detection using natural language processing and machine learning over social interactions“, *Journal of Big Data*, vol. 9, no. 1, p. 33, 2022.

[55] Q. He and E. Agu, „On11: An activity recommendation application to mitigate sedentary lifestyle“, *WPA 2014 - Proceedings of the 2014 ACM Workshop on Physical Analytics, Co-located with MobiSys 2014*, Jun. 2014. DOI: 10.1145/2611264.2611268.

[56] I. Timm, M. Reichert, U. W. Ebner-Priemer, and M. Giurgiu, „Momentary within-subject associations of affective states and physical behavior are moderated by weather conditions in real life: an ambulatory assessment study“, *International Journal of Behavioral Nutrition and Physical Activity*, vol. 20, no. 1, p. 117, 2023.

[57] S. Zepf, N. El Haouij, J. Lee, A. Ghandeharioun, J. Hernandez, and R. W. Picard, „Studying Personalized Just-in-Time Auditory Breathing Guides and Potential Safety Implications during Simulated Driving“, in. New York, NY, USA: Association for Computing Machinery, 2020, pp. 275–283, ISBN: 9781450368612. [Online]. Available: https://doi.org/10.1145/3340631.3394854.

[58] *Flutter Background Geolocation*, Available at https://github.com/transistorsoft/flutter_background_geolocation:22.2.2022.

[59] *Flutter Background Geolocation - Philosophy of Operation*, Available at https://github.com/transistorsoft/flutter_background_geolocation:22.2.2022.

[60] M. Saponaro, A. Vemuri, G. Dominick, and K. Decker, „Contextualization and individualization for just-in-time adaptive interventions to reduce sedentary behavior“, in *Proceedings of the Conference on Health, Inference, and Learning*, ser. CHIL '21, Virtual Event, USA: Association for Computing Machinery, 2021, pp. 246–256, ISBN: 9781450383592. DOI: 10.1145/3450439.3451874. [Online]. Available: https://doi.org/10.1145/3450439.3451874.

[61] F. Cruciani, C. Nugent, I. Cleland, and P. McCullagh, „Rich context information for just-in-time adaptive intervention promoting physical activity“, in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2017, pp. 849–852. DOI: 10.1109/EMBC.2017.8036957.

[62] F. Künzler, „Assessing and Predicting States of Receptivity for Physical Activity Interventions“, Ph.D. dissertation, ETH Zurich, 2020.

[63] *About Garmin*, Available at https://firebase.google.com/pricing. Accessed: 7.5.2022.

[64] *Deploying to a Device without an Apple Developer Account*, Available at https://ionicframework.com/blog/deploying-to-a-device-without-an-apple-developer-account/. Accessed: 4.7.2022.

[65] *Choosing a Membership*, Available at https://developer.apple.com/support/compare-memberships/. Accessed: 4.7.2022.

[66] *Desktop Operating System Market Share Worldwide*, Available at https://developer.apple.com/support/compare-memberships/. Accessed: 4.7.2022.

[67] *Registering Your App with APNs*, Available at https://developer.apple.com/documentation/usernotifications/registering_your_app_with_apns. Accessed: 4.7.2022.

[68] A. Müller, A. Blandford, and L. Yardley, „The Conceptualization of a Just-in-Time Adaptive Intervention (JITAI) for the Reduction of Sedentary Behavior in Older Adults“, *SSRN Electronic Journal*, Jan. 2017. DOI: 10.2139/ssrn.3044368.

[69] *Naive Bayes Classifier*, Available at https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c. Accessed: 11.9.2022.

[70] *Generating a remote notification*, Available at https://developer.apple.com/documentation/usernotifications/setting_up_a_remote_notification_server/generating_a_remote_notification. Accessed: 25.9.2022.

[71] *Regarding the character limit of the body of push notifications*, Available at https://developer.apple.com/forums/thread/709671. Accessed: 25.9.2022.

[72] *The iOS Shortcuts Show Notification action is limited to displaying 256 characters (including spaces and returns) or 20 lines. Whichever comes first.* Available at https://www.reddit.com/r/shortcuts/comments/xclruz/the_ios_shortcuts_show_notification_action_is/. Accessed: 25.9.2022.

[73] *How to Enable (or Disable) Motion & Fitness Tracking with iPhone*, Available at https://osxdaily.com/2015/04/05/toggle-motion-fitness-tracking-iphone/. Accessed: 8.10.2022.

[74] *Don't kill my app!*, Available at https://dontkillmyapp.com/. Accessed: 24.3.2023.