1 Introduction

This thesis is a 15 HEC Master's thesis in Software Engineering. It explores the domain of mobile application development, wearable health technology, and data normalization. As healthcare becomes increasingly more digital, wearable devices like smartwatches and fitness trackers are gaining traction as tools for collecting physiological and behavioral data.

The widespread adoption and acceptance of such devices offers insightful possibilities for health monitoring, especially for conditions like migraine that could benefit from health monitoring. However, as of today that landscape is flawed due to data formats and APIs beeing differently interpreted depending on the different providers, making it difficult to integrate health data across platforms or use it effectively in research and machine learning. [1]

This thesis aims to investigate this challenge and contributes by exploring how a mobile software framework solution might help unify and normalize health data from various sources. The aim is to simplify data handling for researchers and developers while also hopefully enabling more effective use of health data in fields such as machine learning.

1.1 Background

As digital health technologies advance, the integration of wearable devices into everyday life has become increasingly common. The market for devices such as smartwatches, fitness trackers, and other biometric sensors is something that has seen rapid growth in recent years, with big stakeholders such as Alphabet Inc/Google, Apple inc, Garmin Ltd and Samsung Electronics Co. Ltd being some of the most prominent players. [2]. This has in turn led to a surge in the amount of possible health data being collected which has unlocked new potential within health research, disease identification and also potential treatment. [3]

One medical area where this type of data collection posibillities can have a high impact in is migraine detection and prevention. Migraines are complex neurological conditions often triggered by combinations of both physiological and environmental factors [4]. The ability to monitor physiological signals and important metrics that can potentially indiciate an impending migraine attack could be a game changer. Wearable devices can provide possibilities for monitoring these signals, allowing for early detection and prevention. This would in turn lead to a road of benefits for both patients and healthcare providers, including improved quality of life, reduced healthcare costs, and more effective treatment plans.

However, while the capabilities of individual wearable devices are improving, the broader ecosystem remains limited. Each device manufacturer typically provides its own proprietary API and data format. This inconsistency makes it difficult for software developers and researchers to combine data from multiple sources, interpret it uniformly, and make something meaningful out of it.

One major challenge in this area is the lack of a unified framework for integrating wearable health data in a standardized way, especially within mobile environments. There are some existing solutions closely related to these issues but none of those provides a full solution. For example solutions such as the health package and React Native Health [5], [6] are both libraries enabling access to health data, but none of the data is normalized or standardized. This means that the data collected is not in a format that can be easily used for machine learning or other data analysis tasks, where it is perfered to have structured data [7]. This lack of standardization poses a significant barrier to the effective use of wearable health data in both research and development.

1.2 Related work

In recent years, as the market for wearable devices has grown, so has the amount of research and development in this area. The current approaches to wearable data intergration similar to what is

proposed in this thesis can be divided into three categories: web-based solutions, data processing frameworks and data extracting solutions. One noteable example is WearMerge [8], which was presented at IEEE International Conference on Pervasive Computing and Communications Workshops in 2022. WearMerge is a web-based approach which converts wearable data into Open Mhealth schemas. This solution offers similar data aggregation capabilities to our work but lacks the ability and support for mobile usage, which is highly relevant for potential wearable health monitoring.

Along similar research and works there is also Tasrif [9]. Tasrif is a Python based preprocessing framework for wearable data. It is designed to effectively handle and process data from widely used platforms such as Apple Health. More than just targeting bigger platforms, it also directly supports integration with machine learning libraries and focuses on offline data processing. Tasrif is a promising solution for data preprocessing purposes but just like WearMerge, it is not designed for mobile integration.

In addition to web and data processing frameworks, there are also several libraries for extracting data from wearable devices which is closely related to the work in this thesis. The solutions that exist are platform specific and are designed to work for their respective platforms. Some examples for these solutions for this are the health package, [5], React Native Health [6] and React Native Health Connect [10]. More than these libraries being designed to work with specific platforms, they also lack the ability to normalize and standardize the data.

1.3 Problem formulation

While wearable devices have become increasingly capable of capturing health related metrics, there are still room for growth and gaps that can be filled. This is primarily due to the fragmented landscape of health data platforms, where most provider offers its own proprietary data format and API. As discussed in the previous sections, although some server based frameworks like Shimmer support multi-provider integration, they are not designed for mobile environments or real time use.

Existing solutions like WearMerge and Tasrif focus on data aggregation and preprocessing but either lack mobile compatibility or do not address data normalization. Furthermore, platform specific libraries such as React Native Health and Health Connect provide access to raw data but without standardization. This leaves a significant technical gap, which can be summarized as the lack of a unified, mobile based framework capable of collecting and normalizing wearable health data from multiple providers. This gap between single platform solutions and server based frameworks has been identified as a significant barrier in the context of big data healthcare solutions. [11].

To explore this gap, the thesis is structured by the following research questions:

1. How can wearable health data from different platforms be effectively normalized into a unified format suitable for migraine-focused machine learning applications?
2. What are the key requirements and challenges in implementing real-time data normalization for migraine-relevant wearable data in a mobile environment?
3. How effective is the Open mHealth schema as a standardized format for representing migraine-relevant physiological data from diverse wearable devices?

1.4 Motivation

From a scientific perspective, by unifying heterogeneous data from wearable devices into a common format, the thesis contributes to ongoing research in health data integration and machine learning. The findings could potentially support and enable effective data analysis and machine learning applications in the different fields of health research. By proposing a mobile based approach that supports real time data collection and normalization, the thesis aims to address the limitations of

existing solutions and provide a more practical framework that can be used for more efficent research.

Additionally from a societal perspective, by enabling and effectively using wearable health data, wearables can be used to improve health monitoring and also be a tool for early detection and prevention of possible health issues. One of the areas that the thesis aims to focus on is migraine, which is a condition that can possibly be predicted and prevented due to the fact that migraine can be triggered by physiological and environmental factors [4]. By enabling these insights through wearable devices, the project can potentially support better health outcomes and quality of life for individuals suffering from migraines or any other predictable health conditions.

Furthermore, from an industry standpoint, the proposed framework provides a robust and reusable solution for developers and potential healthtech companies by building a framework which has built in support for multiplatform integration along with data normalization. This greatly contributes to effective development in the field of health data management and machine learning. The framework can be used as a foundation for future applications, enabling developers to focus on building their solutions rather than dealing with the complexities of data integration and normalization.

1.5 Results
None obtained

1.6 Scope/Limitation
Not finished

1.7 Target group
The primary target group are software developers and researchers working within the field of digital health, particularly those focusing on mobile application developments and wearable data integration. A challenge these stakeholders potentially face is working around the fragmented health data formats when developing solutions or preparing datasets for machine learning solutions.

Furthermore the framework can also be of interest for healthcare focused research teams aiming to collect and analyze physiological datapoints for conditions like migraine. By offering easier ways to access and normalize data across different platforms, the framework can support both development solutions but also research studies.

1.8 Outline
Not finished

2. Method
The applied method for this thesis is Design Science Research (DSR). DSR was chosen due to its natural fit for projects involved in creation of a software artifact. It is described by vom Brocke et al [12] as a paradigm that seeks to enhance knowledge through the creation of an innovative artifacts. They further explain that the aim of DSR is to generate knowledge about how things can and should be constructed, knowledge that is referred to as design knowledge within the DSR paradigm. Within the scope of this thesis, several research methods are used in combination. These include literature review to understand the theoretical background, artifact construction through iterative development and experimentation to evaluate the artifacts ability to meet the defined objectives. Together these methods form a multimethod approach aligned with the DSR process model.

2.1 Research project

The company Neurawave [13] presented a need for collecting health metrics that could be used for migraine prediction with machine learning. In this work, we follow the DSR methodology outlined by Peffers et al [14].

Phase 1 - Problem identification and insight gathering

We began our research project by conducting open-ended interviews with the founders of neurawave [13]. This approach helps us build a deeper understanding of both stakeholder needs and the problem domain. One of the key requirements was that the data collection mechanism had to be integrated into their existing cross-platform mobile application. The main reason for this requirement is the ability for the existing application to act upon realtime health data. Additionally, we conducted literature review to gather insights regarding the state of the art available solutions, and which health data is commonly used in migraine prediction.

Phase 2 - Objective formulation

Based on the analysis of identified requirements gather in open-ended interviews and resarch gap found in the literature review, we have defined the projects objectives.

Phase 3 - Design and implementation

Based on the formulated objectives, the software artifact which is the data collection framework for mobile platforms was developed and refined iteratively.

Phase 4 - Demonstrating efficacy

Once the artifact has been developed, we will demonstrate its capabilities in addressing the defined problem through system testing. This will be followed by rigarous testing of how well the artifact meets the objectives, likely by comparing the implemented functionality with the stakeholder-defined goals.

Phase 5 - Critical evaluation and feedback integration

The quantitative assessments using controlled experiment is applied to measure the accuracy and reliability of the data collection framework in different scenarios or use-cases.

Phase 6 - Dissemination and community engagement

Our findings and its implications are articulated in this thesis work for dissemination among AI researchers and developers.

2.2 Research methods

We began our research project by conducting open-ended interviews with the founders of Neurawave [13]. Open-ended interviewing allows for a flexible, free-flowing conversation and is considered a qualitative method of data collection [15]. This method was chosen because the stakeholders possessed significantly more domain knowledge than us, making them essential in framing the problem accurately. As described by Alshenqeeti [15], the purpose of open-ended interviews is to “broaden the scope of understanding of investigated phenomena”. We believed this approach would help us build a deeper understanding of both stakeholder needs and the problem domain.

Open-ended interviews fall within the broader category of qualitative methods, which are generally holistic and aimed at answering “what” questions. In contrast, structured interviews are more quantitative, relying on closed-ended questions, such as those with yes/no responses [15]. As Lakshman et al. [16] explain, quantitative methods focus on examining the effect of an independent variable on a dependent variable in a measurable, numerical way. However, at this early stage in the

project, a qualitative approach was better suited to developing our understanding and framing our objectives.

Following the interviews, we conducted litterature research to expand our contextual understanding of the problem area. Literature research involves exploring a broad body of work related to a topic and is typically less rigid than a systematic literature review [17]. A systematic review, by contrast, targets a specific research question and is intended to gather empirical evidence to support conclusions. We chose a narrative-style literature review to ground our work theoretically and better understand the current landscape of data collection for health-related machine learning applications.

Alternative approaches, such as expert interviews could also have been employed to gather real-world insights. As described by Döringer [18], expert interviews are valuable for problem-centered exploration and knowledge gathering. However, due to the time and resource constraints, this method was not feasible within the scope of this thesis. Nonetheless, it represents a promising avenue for future research and validation.

During the design and development phase, we followed the artifact creation process outlined in the Design Science Research methodology. While this is not an empirical method in the classical sense, it is a core research activity in DSR, where existing knowledge and stakeholder input are synthesized into a functional solution. We began by defining application requirements that map directly to stakeholder requirements and then iteratively developed the plugin aimed at fulfilling those needs. No specific structured development framework as followed, as our team had prior experience in collaborative software development.

To evaluate the developed solution, we will use experimentation as the primary method of validating functionality. According to Basili et al. [19], experimentation is an iterative process of hypothesizing and testing. In our case, this involves defining (or redefining) software requirements, implementing them (the hypothesis), and verifying wether those requirements are fulfilled (the test).

Additionally, we will perform validation to ensure tha the plugin's software requirements align with the original stakeholder expectation [20]. As an alternative to experimentation, we considered survey-based research to gather stakeholder opinions on desired functionality. However, we ultimately decided against this due to our limited timeframe. We also believed that the iterative loop supported by experimentation would be hindered by the time required to create, distribute and analyze a survey. In our context, experimentation offers more immediate feedback and supports rapid iteration, which we viewed as essential for effective development.

2.3 Data for migraine prediction using machine learning

During the literature review of previous studies investigating migraine prediction using machine learning, we found only two studies [21], [22]. The health metrics used in both studies are outlined below. It is worth noting that several health metrics was included in either study, such as hours of working [22] out and step count [21]. We also found some studies [23], [24], [25] that examined different triggers (internal and external factors) such as weather, diet, hormonal changes that could be valuable in predicting migraine episodes, but none of those were included in the studies that combined migraine prediction with machine learning.

2.4 Wearables and datastores

We decided that the framework should target the native health data stores due to the fact that they provide a unified API for extracting health data that is not affected by the third party wearable. Since Android has 71.9% of the global market share and IoS having 27.68% [26], thus making Google Health Connect and Apple Health Kit the native data stores we mainly target. During the validation of the framework we will perform experiments using Apple Watch, Google Fitbit and Garmin smart

watches as providers of health data to the native datastores. The selection of wearables are based on available resources and a reasonable coverage of the most common wearables.

Health metric	Description	Refs
Heart rate	The rate of heart beats per minute.	[21], [22]
Heart rate variability	Variation in time between heartbeats	[21]
Skin temperature	The temperature of the skin, preferably measured at finger or wrist	[21], [22]
Skin conductance	Measurement of how good the skin conducts electricity. Has been shown to increase due to sweating, as a response to stress	[21]
Respiratory rate	Measurement of the rate of breathing. Measured as breaths per minute.	[21]
Sleep time	Measurement the amount of hours slept	[21], [22]

We have decided to include heart rate and skin temperature in our data processing. The selection is based on a combination on what data is available from regular wearables and what health metrics have been shown in previous studies to be of high value.

2.5 Reliability and validity

Validity

One limitation related to validity stems from the scope of the evaluation. The developed component will be tested by two individuals in controlled, but limited, settings. As such, the generalizability of the findings to a broader user base or different use cases is uncertain. Furthermore, the component is validated exclusively on migraine related health data. While it may be adaptable to other domains within health monitoring, such applicability remains untested and therefore unknown.

Another validity concern involves the integration of the component with the datastores and/or wearable devices. The component will initially be tested using two or three specific health data providers. Its performance and compatibility with other, non integrated wearables remain unverified, and thus it cannot be assumed that the framework will function the equally well across other health data providers.

Additionally the component rely on third party API's for data collection. Any inaccuracies or failure in those APIs could directly compromise the integrity of the data and therefore the validity of the results. This is particularly concerning given that the quality and frequency of data vary significantly between high cost devices (e.g, Apple Watch, Fitbit) and low cost alternatives (e.g, Bangle.js, EmotiBit). The component might perform well in high quality data environments but underperform in cases where the input data is sparse, noisy or unreliable.

Reliability

A significant threat to reliability is the dynamic nature of third party wearable health data provider APIs. These APIs are frequently updated, and changes in their structure or functionality may break the integration with out framework. This implies that future researchers attempting to replicate this

study might need to adapt the codebase to updated APIs. To mitigate this risk, we will document the integration process thoroughly and provide implementation guides to facilitate replication.

On the other hand, the controlled experiments conducted as a part of this project will be carefully designed and well documented. This will support the reproducibility of results and allow others to validate the findings under similar conditions.

Risks and mitigations strategies

A number of general risks may impact both the reliability and validity of the project outcomes:

- Lack of access to wearable data may hinder testing or validation of the component under real world conditions. To address this, we prioritize using devices with public or open APIs and ensure local caching of test data were possible.
- Complexity of data normalization and integration may result in inconsistent behavior across devices. This will be mitigated by adopting standardized data formats and implementing preprocessing checks.
- Data quality issues, especially from low-cost or experimental devices may reduce the effectiveness of the component. We will include a validation layers to detect and handle poor input.
- Time management and scope creep pose risks to project completion. A well defined timeline and iterative planning approach are used to ensure focus and manage scope.
- Limited expertise in wearable APIs and data pipelines may slow progress. To mitigate this, we will rely on documentation and existing libraries where possible.
- Difficulty demonstrating machine learning readiness of the collected data may arise due to insufficient volume or inconsistency. We will evaluate and report data quality with descriptive statistics and document its limitations for future machine learning integration.
- API rate limits and access restrictions may impact data collection throughput. Mitigation strategies include caching responses, adhering to API usage guidelines and contacting vendors if necessary.

By identifying these threats and planning accordingly, this project strives to maintain both reliability and validity, despite the challenges inherent in working with third-party hardware and health data.

2.6 Ethical considerations

While this project doest not involve direct interaction with human participants outside of the stakeholders in the initial interviews, it does involve collecting and processing of health related data via wearable devices. This introduces several ethical considerations, particular around privacy, data confidentiality and data ownership.

Confidentiality

The component is designed to collect health metrics that are sensitive by nature, such as heart rate, heart rate variability and skin temperature. Even though no personally identifiable information is intended to be handled, care must be taken to ensure that the processing of data is in accordance with privacy regulations such as GDPR.

To address this, all test data used during development and experimentation will be anonymized. No user names, contact information, or device IDs linked to individuals will be stored or processed in any way that could allow for re-identification. No data will be stored within the component, only processing of the data. No data will be shared externally or stored in the cloud.

Sampling bias

The testing of the component will be done with using a limited number of wearable devices and datasets related to migraine prediction. As a result, the dataset is narrow in scope which may

introduce selection bias. This poses a limitation upon the generalizability of the results to other conditions such as other devices or populations. Since the primary objective is to develop and evaluate a working prototype in a focused context, this tradeoff is considered acceptable for the scope of the project.

Participation and consent

If any real user data is collected (for example, if Neurawave employees or external users voluntarily contribute data for evaluation), explicit informed consent will be obtained in written form. Participants will be made aware of what data is being collected for processing and how it will be used. Participation will be strictly voluntary, and participants will have the right to withdraw at any time without any consequence.

Risk of harm

This project poses minimal risk of harm. as it does not involve any physical or psychological intervention. However, improper handling of sensitive data could lead to privacy breaches. To minimize this risk, the component itself will not store any data.

3. Theoretical background

The current landscape of health data has seen a significant transformation due to the prliferation of wearable devices. Devices such as smartwatches, smart rings and fitness trackers have advanced rapidly in recent years, now offering high-quality, clinically certified data collection capabilities [1]. This longitudinal and increasingly accurate health monitoring opens up new possibilities in health research, disease detection and personalized treatment strategies [3]. The global wearable device market continues to grow at a substantial rate, with an estimated compound annual growth rate of 14.6% between 2023 and 2030 [[2]]. Leading industry actors include Alphabet Inc./Google, Apple Inc., Garmin Ltd., and Samsung Electronics Co., Ltd. [[2]].

While the growth of wearable technologies presents valuable opportunities, it also introduces significant technical and interoperability challenges. Each provider typically offers its own platforms for accessing and modifying health data, with unique APIs, data models and permission systems. This fragmentation complicates efforts to aggregate data from multiple providers. For instance, Apple's HealthKit provides a generalized abstraction for measurements via types such as HKQuantityType [27], representing quantities like step conts. In contrast, Google Health structures similar metrics as domain-specific records such as StepRecord [28], including associated metadata like start and end times. These inconsistensies in data modeling extend beyond simple metrics such as step counts to more complex physiological measurements such as heart rate variability, sleep stages and stress levels. The absence of standardized approaches for normalizing and integrating such heterogenous health data represents a significant obstacle for developers and researchers building cross platform applications, particulary for those leveraging machine learning techniques.

Existing frameworks for collecting health data

The existing frameworks for collecting health data are outlined below:

Framework	Features	Missing features
Health 12.0.1, Flutter package	Enables reading and writing health data to and from Apple health and Google Health Connect.	• Lacks support for several providers (Fitbit, garmin etc)

Framework	Features	Missing features
		• No support for Open mHealth data format.
React Native Health, React native package	Package for interacting with Apple HealthKit for iOS.	• Lacks support for several providers (e.g., Google Health connect). • No support for Open mHealth.
React native health connect, React native package	Package for interacting with Health Connect for Android.	• Lacks support for several providers (e.g Apple HealthKit for iOS) • No support for Open mHealth.
Shimmer, web platform	Application for extracting health data from multiple providers into Open mHealth data format.	• Is not natively supported on mobile.
Tasrif, python application	Application for extracting health data from multiple providers. Integrates with existing python ML libraries.	• Is not natively supported on mobile. • Does not support Open mHealth.

While each of these frameworks fulfills part of the requirements for multi-provider health data integration, none currently provide a complete, mobile-native, cross-platform solution that supports standardized output such as Open mHealth format.

Health data standards

The current state of e-health data standards are outlined below:

Standard	Description	Reference
Fast Health Interoperability Resources - FHIR	Data standard for exchaning health care information digitally. Modular specification with focus on health care, with modules such as medications, diagnostics, etc.	[29]
Open mHealth	Data standard for mobile health data. Provides schemas for creating a uniform data structure for health data recorded by wearable devices.	[30]

Standard	Description	Reference
IEEE P1752 - Standard for mobile health data working group	Provides data standard for representing physical activity, sleep and metadata.	[31]

Efforts toward standardizing mobile health data are ongoing. Babu et al. [3] highlight the importance of cross-organizational collaboration to enhance data quality, consistency and interoperability.

Health data stores and API

The plugin developed in this thesis targets two major health data stores: Apple HealthKit [32] on iOS and Google Health Connect [33] (formerly Google Fit) on Android. While both APIs offer similar capabilities for reading and writing data, they differ significantly in internal structure and terminology.

Access to these data stores requires explicit user permission, granted per-app and per-data-type, ensuring user privacy and conrol. Once permissions are granted, both platforms expose APIs for querying health data.

Google Health Connect uses specific record classes, such as Steps [34] or HeartRate [35], each containing metadata like startTime, endTime and a value field. In contrast, Apple HealthKit uses types such as HKQuantityType [36] or HKWorkoutType [37]. When requesting step data, for example, a HKQuantitySample [38] is returned containing the step count as an HKQuantity along with metadata like the measurement time window and data source.

Selected software

One requirement for this framework is that it must be cross-platform and mobile-native. Several development frameworks support this, including React Native [39], Flutter [40], LynxJS [41] and Kotlin Multiplatofmr [42].

Flutter was selected for this project due to existing infrastructure and developer experience within the organization (Neurawave). Flutter uses the Dart programming language, with native platform functionality implemented in Swift (iOS) and Kotlin (Android). Platform-specific functionality is accessed via MethodChannels, which allow Flutter code to call native APIs directly.

No additional third-party plugins will be used beyond what is required to interface with HealthKit and Health Connect.

Research gap and problem formulation

Despite the growing availability of health data APIs and frameworks, there is no existing mobile-native, cross-platform plugin capable of aggregating health data from multiple providers and exporting it in a standardized format such as Open mHealth. While it is theoretically possible to combine mutiple existing tools to achieve similar results, this approach is highly impractical and prone to compability issues, platform-specific limitations, and increased development complexity.

This fragmentation presents a significant barrier for developers and researchers who wish to build cross-platform health solutions or apply machine learning techniques to unified health datasets. The framework developed in this thesis aims to address this gap by offering a native, extensible solution for standardized mobile health data integration.

4 Research project - Implementation

4.1 Design and technique

We designed a Flutter plugin using the Dart programming language, with platform-specific implementations written in Swift for iOS and Kotlin for Android. The plugin exposes a public API that enables communication with the native health data stores. When a user invokes a method from this API, a MethodChannel is used internally to bridge the Dart code and the native platform code. The Dart layer is responsible for defining the public API, making method calls to the platform specific implementations, and performing data transformation. The native code handles permission requests and performs the actual data extraction.

The high level data flow is illustrated below:

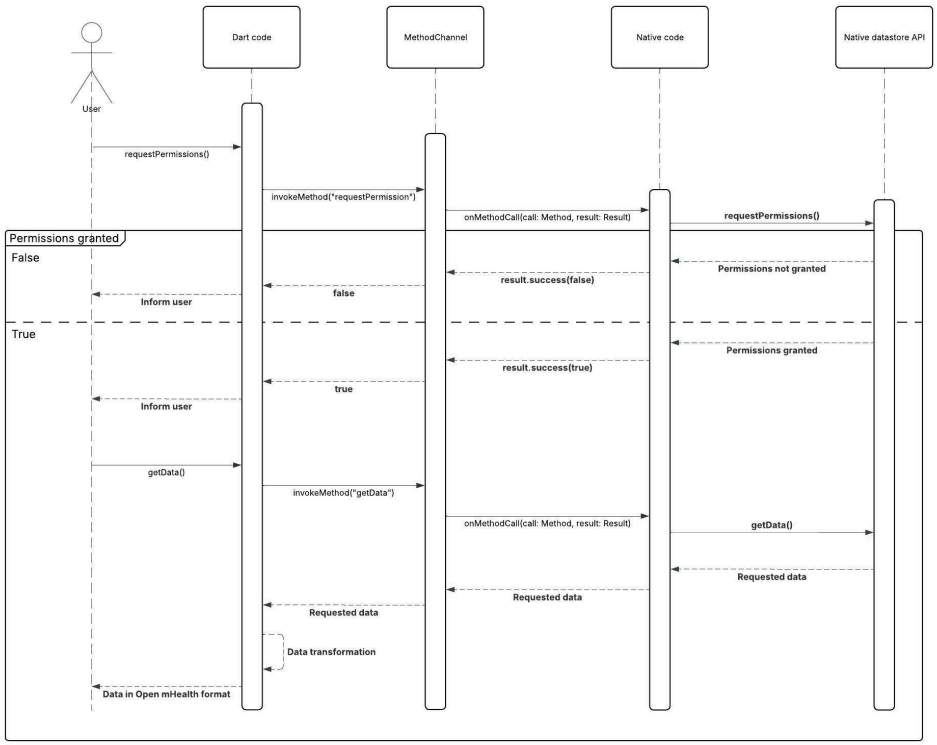

Figure 1: Data flow of requesting permissions and extracting health data

Several standardization models exist for structuring health data, including Open mHealth, FHIR and IEEE P1752. We chose Open mHealth due to its lightweight nature, JSON-compatibility and its specific focus on mobile health data. While FHIR is a powerful and widely adopted clinical standard, it introduces significant overhead for mobile use cases, requiring modeling of patients, medications and other elements irrelevant to this project. Additionally, Open mHealth has begun incorporating elements from IEEE P1752, further reinforcing its suitability for mobile-focused applications.

This framework targets heart rate and skin temperature data. On Android, it extracts HeartRate [35] and SkinTemperature [43] records from Google Health Connect. On iOS, it extracts HKQuantitySample objects with type heartRate [44] and bodyTemperature [45], filtering for locations associated with skin contant measurements (e.g armpit, body, finger, toe, forehead). The extracted data is transformed into the Open mHealth Heart Rate schema [46] and a custom Skin Temperature schema modeled after the Open mHealth Body Temperature schema [47].

4.2 Implementation concerns

To ensure interoperability, the plugin normalizes all extraced data into Open mHealth format regardless of the source platform or underlying API. Metadata not defined within the Open mHealth schema, such as device name or specific hardware information, is discarded to maintain consistency and reduce variability across platforms.

The plugin ensures that permission handling is performed correctly and that permission states are regularly verified, as users may revoke permissions at any time. Both HealthKit and Health Connect offer similar permission models, which posed minimal challenges during implementation.

Data granularity and sampling resolution is determined by the user via an input time window. The plugin adheres to the data limits and resolution supported by the respective health data store, returning data only within the requested time frame.

To account for situations where a health data store is not available (e.g the data store is not installed), the plugin includes logic to direct users to the respective app store to download the required app. Additionally, if a user attempts to use the plugin on an unsupported OS version, they are informed of the compatibility issue through user-facing feedback.

The plugin is designed with extensibility in mind. Its architecture allows for straightforward integration of additional data types, data providers (e.g Fitbit, Garmin) and health metrics in fututre iterations.

4.3 Demonstration

The plugin will be demonstrated through a dedicated test application built in Flutter. This application implements the full data flow, including permission requests and data extraction. In addition, the application includes automated tests that validate the correctness of the data transformation and adherence to the Open mHealth schema.

The test application will be evaluated using real devices including Apple Watch, Fitbit and Garmin Venu to simulate real-world usage and ensure compatibility across a range of wearables. The demonstration will showcase both real-time and historical data extraction, within the constraints of each data stores retention policy and access limitations.

The plugin will be considered succesful if it consistently provides accurate and correctly formatted Open mHealth data on both iOS and Android platforms, across a diverse set of wearable devices.

5 Results

No results have been obtained since the thesis project is not completed yet.

6 Analysis

No analysis has been made since the thesis project is not completed yet.

7 Discussion

No discussion has been done since the thesis project is not done yet.

8 Conclusion

No conclusion has been drawn since the thesis project is not completed yet.

Bibliography

[1] Sophie Huhn *et al.*, "The Impact of Wearable Technologies in Health Research: Scoping Review," *JMIR Mhealth Uhealth*, vol. 10, no. 1, Jan. 2022.

[2] Grand View Research, "Wearable Technology Market Size, Share & Trends Analysis Report By Product (Head & Eyewear, Wristwear), By Application (Consumer Electronics, Healthcare), By Region (Asia Pacific, Europe), And Segment Forecasts, 2023 - 2030," 978–1–68038–165–8.

[3] Mohan Babu, Ziv Lautman, Xiangping Lin, Milan H B Sobota, and Michael P Snyder, "Wearable Devices: Implications for Precision Medicine and the Future of Health Care," *Annual Review of Medicine*, vol. 1, no. 1, Nov. 2023.

[4] Hassan Kesserwani, "Migraine Triggers: An Overview of the Pharmacology, Biochemistry, Atmospherics, and Their Effects on Neural Networks," *Cureus*, vol. 13, no. 4, Apr. 2021.

[5] "Health Package." [Online]. Available: https://pub.dev/packages/health

[6] "React Native Health." [Online]. Available: https://www.npmjs.com/package/react-native-health

[7] Maryam Tayefi *et al.*, "Challenges and opportunities beyond structured data in analysis of electronic health records," *WIREs - Wiley Interdisciplinary Reviews*, vol. 13, no. 6, Feb. 2021, doi: 10.1002/wics.1549.

[8] Dimitrios Panteleimon Giakatos, Sofia Yfantidou, Stefanos Efstathiou, and Athena Vakali, "WearMerge: An Interoperable Framework for Self-tracking Data Integration and Standardization," May 06, 2022, *IEEE*. doi: 10.1109/PerComWorkshops53856.2022.9767462.

[9] Abdulaziz Al Homaid, Syed Hashim, Fadhil Abubaker, Ummar Abbas, Faisal Farooq, and Joao Palotti, "Tasrif: processing wearable data in Python," May 06, 2022, *IEEE*. doi: 10.1109/PerComWorkshops53856.2022.9767286.

[10] "React Native Health Connect." [Online]. Available: https://www.npmjs.com/package/react-native-health-connect

[11] Sumarsono, Muhammad Anshari, and Mohammad Nabil Almunawar, "Big Data in Healthcare for Personalization & Customization of Healthcare Services," Sep. 19, 2019, *IEEE*. doi: 10.1109/ICIMTech.2019.8843822.

[12] in *Design Science Research Cases*, 2020, pp. 1–13.

[13] Neurawave, "Neurawave's website." [Online]. Available: https://www.neurawave.se/

[14] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee, "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems*, vol. 24, no. 3, 2007, doi: 10.2753/MIS0742-1222240302.

[15] Hamza Alshenqeeti, *English Linguistics Research*, vol. 3, no. 1, Mar. 2014, doi: 10.5430/elr.v3n1p39.

[16] M. Lakshman, Leena Sinha, Moumita Biswas, Maryann Charles, and N.K. Arora, "Quantitative vs Qualitative research methods," *The Indian Journal of Pediatrics*, vol. 67, no. 1, May 2000, doi: https://doi.org/10.1007/BF02820690.

[17] Priscilla Robinson and John Lowe, "Literature reviews vs systematic reviews," *Australian and New Zealand Journal of Public Health*, vol. 39, no. 2, Apr. 2015, doi: 10.1111/1753-6405.12393.

[18] Stefanie Döringer, "The problem-centered expert interview. Combining qualitative interviewing approaches for investigating implicit expert knowledge," *International Journal of Social Research Methodology*, vol. 24, no. 3, May 2020, doi: 10.1080/13645579.2020.1766777.

[19] Victor R. Basili, Richard W. Selby, and David H. Hutchens, "Experimentation in software engineering," *IEEE Transactions on Software Engineering*, vol. 12, no. 7, Sep. 2012, doi: 10.1109/TSE.1986.6312975.

[20] P. Sujatha, G. V. Sankar, A. S. Rao, and T. Satyanarayana, "The Role of Software Verification and Validation in Software Development Process," *IETE Technical Review*, vol. 18, no. 1, Mar. 2015, doi: 10.1080/02564602.2001.11416938.

[21] Viroslava Kapustynska, Vytautas Abromavičius, Artūras Serackis, Šarūnas Paulikas, Kristina Ryliškienė, and Saulius Andruškevičius, "Machine Learning and Wearable Technology: Monitoring Changes in Biomedical Signal Patterns during Pre-Migraine Nights," *Healthcare (Basel)*, vol. 12, no. 17, Aug. 2024, doi: https://doi.org/10.3390/healthcare12171701.

[22] Anker Stubberud *et al.*, "Forecasting migraine with machine learning based on mobile phone diary and wearable data," *Cephalgia*, vol. 43, no. 5, May 2023, doi: https://doi.org/10.1177/03331024231169244.

[23] Antonio L Aguilar-Shea, Javier A Membrilla Md, and Javier Diaz-de-Teran, "Migraine review for general practice," *Aten Primaria*, vol. 54, no. 2, Nov. 2021, doi: 10.1016/j.aprim.2021.102208.

[24] Michael J Marmura, "Triggers, Protectors, and Predictors in Episodic Migraine," *Current pain and headache reports*, vol. 12, no. 81, Oct. 2018, doi: 10.1007/s11916-018-0734-0.

[25] Dana P Turner, Adriana D Lebowitz, Ivana Chtay, and Timothy T Houle, "Forecasting Migraine Attacks and the Utility of Identifying Triggers," *Current pain and headache reports*, vol. 22, no. 9, Jul. 2018, doi: 10.1007/s11916-018-0715-3.

[26] Statcounter, "Mobile os market share." [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/worldwide

[27] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkquantitytype

[28] [Online]. Available: https://developer.android.com/reference/kotlin/androidx/health/connect/client/records/StepsRecord

[29] [Online]. Available: https://hl7.org/fhir/

[30] [Online]. Available: https://www.openmhealth.org/

[31] [Online]. Available: https://standards.ieee.org/ieee/1752.1/6982/

[32] [Online]. Available: https://developer.apple.com/documentation/healthkit

[33] [Online]. Available: https://developer.android.com/health-and-fitness/guides/health-connect

[34] [Online]. Available: https://developer.android.com/reference/kotlin/androidx/health/connect/client/records/StepsRecord

[35] [Online]. Available: https://developer.android.com/reference/kotlin/androidx/health/connect/client/records/HeartRateRecord

[36] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkquantitytype

[37] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkworkouttype

[38] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkquantitysample

[39] [Online]. Available: https://reactnative.dev/

[40] [Online]. Available: https://flutter.dev/

[41] [Online]. Available: https://lynxjs.org/

[42] [Online]. Available: https://kotlinlang.org/docs/multiplatform.html

[43] [Online]. Available: https://developer.android.com/reference/kotlin/androidx/health/connect/client/records/SkinTemperatureRecord

[44] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkquantitytypeidentifier/heartrate

[45] [Online]. Available: https://developer.apple.com/documentation/healthkit/hkquantitytypeidentifier/bodytemperature

[46] [Online]. Available: https://www.openmhealth.org/documentation/#/schema-docs/schema-library/schemas/omh_heart-rate

[47] [Online]. Available: https://www.openmhealth.org/documentation/#/schema-docs/schema-library/schemas/omh_body-temperature