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 [2]. 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 [[3]]. Leading industry actors include Alphabet Inc./Google, Apple Inc., Garmin Ltd., and Samsung Electronics Co., Ltd. [[3]].

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 [4], representing quantities like step conts. In contrast, Google Health structures similar metrics as domain-specific records such as StepRecord [5], 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) - 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	• Is not natively supported on mobile.

Framework	Features	Missing features
	providers into Open mHealth data format.	
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.	[6]
Open mHealth	Data standard for mobile health data. Provides schemas for creating a uniform data structure for health data recorded by wearable devices.	[7]
IEEE P1752 - Standard for mobile health data working group	Provides data standard for representing physical activity, sleep and metadata.	[8]

Efforts toward standardizing mobile health data are ongoing. Babu et al. [2] 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 [9] on iOS and Google Health Connect [10] (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 [11] or HeartRate [12], each containing metadata like startTime, endTime and a value field. In contrast, Apple HealthKit uses types such as HKQuantityType [13] or HKWorkoutType [14]. When requesting step data, for example, a HKQuantitySample [15] 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 [16], Flutter [17], LynxJS [18] and Kotlin Multiplatofmr [19].

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.

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] 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.

[3] 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.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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