



HARVARD
MEDICAL SCHOOL



Massachusetts General Hospital
Founding Member, Mass General Brigham



**Politecnico
di Torino**

Politecnico di Torino

Master's Degree Course in Biomedical Engineering

Academic Year 2025/2026

Development of a Mobile Application for a Sensorized Knee Sleeve System in Osteoarthritis Monitoring

Supervisors:

Prof. Danilo Demarchi
Prof. Paolo Bonato
Dr. Benito Lorenzo Pugliese

Candidate:

Andrea Orlando

Abstract

Knee osteoarthritis can affect everyday life through pain, stiffness, and reduced mobility. These symptoms are not always constant, but can change over time and during daily activities. For this reason, a periodic clinical visit may give only a partial view of the patient’s condition. Wearable sensors connected to a mobile application can therefore support day-to-day monitoring of knee-related signals and, after proper validation, may also allow this monitoring to be extended outside the clinical setting.

This thesis presents the design and development of a research-oriented mobile application for a sensorized knee sleeve system intended for knee osteoarthritis monitoring. The application was developed as a modular React Native prototype and was designed to communicate with wearable sensing devices through Bluetooth Low Energy. The system supports device discovery, connection management, command transmission, real-time data streaming, packet decoding, signal visualization, local storage, data export, and basic post-processing.

In the first development phase, the application was organized as a modular hub including separate workflows for temperature and bioimpedance monitoring. The temperature module allows real-time visualization of twelve thermal channels, offline inspection, data export, and spatial heatmap visualization. The bioimpedance module includes calibration, continuous acquisition, sweep measurements, offline visualization, and export of resistance, reactance, and knee flexion–extension angle. In the second phase, the software architecture was extended through a Developer Mode aimed at preparing the future integration of a fully multisensorized knee sleeve. Since the final integrated hardware was not available, this part was implemented using simulated integrated packets and reconstructed multimodal datasets. The pipeline included packet design, binary parsing, CRC32 integrity verification, synchronization, and processing of temperature, bioimpedance, inertial, and EMG-related signals. Exploratory indicators such as gait-event timing, EMG activation percentage, Heat Distribution Index, and a frequency-dependent bioimpedance ratio were computed and visualized.

The developed system showed the technical feasibility of a mobile platform for multimodal wearable knee monitoring. However, the extracted indicators should be interpreted as technical and exploratory descriptors, not as clinically validated biomarkers. Future work should focus on integration with the final hardware device and validation in real acquisition conditions.

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Prof. Danilo Demarchi and Prof. Paolo Bonato, for their guidance and support throughout this thesis work. I am also deeply grateful to Benito Lorenzo Pugliese for his supervision and constant support during all the activities I carried out in the laboratory. I would also like to thank the Motion Analysis Lab team for welcoming me and for giving me the opportunity to work in a stimulating research environment. Finally, I would like to thank my family and friends for their constant support.

Table of Contents

Acknowledgements	II
List of Figures	VII
Onward	1
1 Introduction	3
1.1 Background	3
1.1.1 Pathophysiology	3
1.1.2 Clinical Manifestations	5
1.1.3 Treatment and Rehabilitation	6
1.2 Digital Health and Telerehabilitation	7
1.3 Rationale for the Proposed System	8
1.4 Aim of the Thesis	9
2 Materials & Methods	11
2.1 System Requirements	11
2.1.1 Functional Requirements	11
2.1.2 Non-Functional Requirements	12
2.1.3 Design Constraints	13
2.2 Hardware Architecture	14
2.2.1 Target Mobile Platform	14
2.2.2 Thermal Measurement Module	15
2.2.3 Bioimpedance Measurement Module	17
2.2.4 Inertial Measurement Module	17
2.2.5 Open-Source EMG-IMU Dataset	20
2.3 Software Architecture	21
2.3.1 Overall System Design	21
2.3.2 Framework and Libraries	23
2.4 Bluetooth Low Energy Communication	24
2.4.1 BLE Communication Architecture	24

2.4.2	Device Discovery and Connection Establishment	25
2.4.3	Device Configuration and Operating Modes	26
2.4.4	Real-Time Data Streaming	26
2.4.5	Data Transmission Protocol	27
2.4.6	Connection Monitoring and Robustness	28
2.5	Data Management and Storage Architecture	29
2.5.1	Session Model & Organization	29
2.5.2	Local Persistence Strategy	29
2.5.3	Session Export and File Handling	30
2.6	Signal Processing and Computation Methods	30
2.6.1	Temperature Signal Processing	31
2.6.2	Bioimpedance Signal Processing	32
2.6.3	EMG Signal Processing	33
2.6.4	Gyroscope Signal Processing	35
3	Results	37
3.1	Phase 1 – Multi modalities App Hub	38
3.1.1	Multi-Modality Application Hub	38
3.1.2	Temperature Module	39
3.1.2.1	Real-Time Mode	40
3.1.2.2	Offline Visualization and Data Export	42
3.1.2.3	Grid Mode	45
3.1.3	Bioimpedance Module	49
3.1.3.1	Calibration Mode	50
3.1.3.2	Real-Time Continuous Mode	52
3.1.3.3	Sweep Mode	53
3.1.3.4	Offline Visualization and Data Export	55
3.1.4	Settings Module	57
3.2	Phase 2 – Multimodal Developer Mode	60
3.2.1	Dummy Multimodal Dataset Construction	62
3.2.2	Implemented Packet Architecture and Parsing	63
3.2.3	Multimodal Signal Processing	67
3.2.4	Extracted Indicators and Reporting	68
3.2.5	Developer Mode Implementation	70
4	Discussion	74
4.1	Interpretation of Results	74
4.1.1	Feasibility of a Multimodal Monitoring System	74
4.1.2	Robustness of the Signal Processing Pipeline	75
4.1.3	Physiological Meaning of the Extracted Indicators	78
4.2	Limitations and Critical Considerations	78

4.2.1	Use of Simulated Multimodal Data	78
4.2.2	Limited Experimental Validation	79
4.2.3	Simplified Physiological Interpretation	79
4.2.4	Computational and Mobile Constraints	80
4.3	Future Work	80
5	Conclusion	82
5.1	Summary of the Work	82
5.2	Main Contributions	83
6	Appendix: Digital Platform for Nutrition Support and Remote Follow-Up in Patients with Brain Injuries	84
6.1	Clinical Context and Project Rationale	84
6.2	Objectives of the Platform	85
6.3	Overall System Architecture	85
6.4	Mobile Application for Patients	86
6.5	Web Dashboard for Healthcare Providers	89
6.6	Backend, Authentication, and Data Synchronization	91
6.7	Potential Impact and Future Developments	92
	Bibliography	93

List of Figures

1.1	Comparison between a normal knee joint and an osteoarthritic knee, showing cartilage degradation, subchondral bone remodeling, synovial hypertrophy, osteophyte formation, ligament dysfunction, muscle atrophy, and meniscal damage[2].	4
2.1	Google Pixel 5 used as the target mobile platform for application development and testing.	15
2.2	Temperature Knee Sleeve used for spatial temperature monitoring around the knee.	16
2.3	BioZ Knee Sleeve used to estimate knee flexion–extension movement and bioimpedance.	19
2.4	Sensor locations selected from the open-source EMG-IMU dataset. The black boxes indicate the IMU positions used in this work, while the red box indicates the EMG signal selected from the vastus medialis.	21
2.5	Block diagram of the software architecture, showing the main application modules and their organization into functional workflows.	23
2.6	Magnitude response of the low-pass filtering applied to the temperature signal. The 2nd-order Butterworth filter and the effective forward–backward response are shown, with the cutoff frequency set at 0.15 Hz.	32
2.7	Frequency response comparison of the filters considered for bioimpedance signal preprocessing. The 2nd-order Butterworth filter and the exponential moving average filter are shown, with the -3 dB cutoff frequency set at 6 Hz.	33
2.8	The top figure shows the bandpass filter, which is the first filter used to isolate the desired signal, while the bottom figure shows the low-pass filter used to isolate the envelope.	35
3.1	General structure of multi-modality application hub developed during the first phase of the project.	38

3.2	Overview of the temperature module, including the main operating modes implemented for acquisition and visualization.	39
3.3	Temperature module interfaces showing the main visualization options and acquisition-related screens.	41
3.4	Offline visualization of recorded temperature sessions, including channel selection and signal inspection tools.	44
3.5	Example of recorded temperature data visualization before and after interactive signal processing.	45
3.6	Color mapping pipeline used in the Grid Mode. The temperature range is selected either in fixed or dynamic mode, then the values are normalized, clamped, quantized, and finally mapped to the corresponding heatmap colors.	47
3.7	Grid Mode interface showing the spatial heatmap visualization and the additional temperature-related indicators.	49
3.8	Overview of the bioimpedance module, including the main operating modes implemented for calibration, continuous acquisition, sweep acquisition, and data visualization.	50
3.9	Calibration workflow of the bioimpedance module, including IMU offset calibration, flexion–extension angle calibration, and access to the mode selection page.	51
3.10	Real-time continuous mode of the bioimpedance module, showing the simultaneous visualization of knee flexion–extension angle, resistance, and reactance during acquisition.	53
3.11	Sweep Mode interface of the bioimpedance module, showing the Cole–Cole plot and the table of resistance and reactance values acquired at different frequencies.	55
3.12	Offline visualization mode of the bioimpedance module. The interface allows the user to select saved sessions and visualize knee flexion–extension angle together with resistance and reactance over the same time domain.	56
3.13	Settings Module interfaces showing the configurable parameters for BLE communication, acquisition, visualization, filtering, and storage.	59
3.14	Initial workflow of the Developer Mode. The interface includes Bluetooth device scanning, acquisition management, and initialization of the multimodal analysis pipeline.	61
3.15	General structure of the binary packet implemented for the multimodal Developer Mode. The packet includes a sync word, packet identifier, sequence number, timestamp, payload, and CRC32 integrity field.	63

3.16	Example of packet arrival sequence in the simulated multimodal stream. Each cell represents an incoming packet, ordered according to the transmission frequency of the corresponding sensing modality (e.g., IMU, EMG, TEMP, Bioimpedance 5kHz (BIOZ), and Bioimpedance 5kHz (BIOZ)).	66
3.17	Developer Mode interfaces showing the visualization of multimodal signals and extracted indicators, including gait-related parameters, EMG activation, and synchronized signal trends.	72
3.18	Additional Developer Mode interfaces related to multimodal analysis and data export, showing the integration of the different processing outputs within the mobile application.	73
6.1	Mobile application interfaces. The left screen shows the calendar page with a scheduled Zoom appointment, while the right screen shows the nutrition dashboard with calorie tracking, water intake monitoring, food search, and logged meals.	87
6.2	Additional mobile application interfaces. The left screen shows the modal for adding water intake, while the right screen shows the account registration page.	88
6.3	Web dashboard page for meeting management. The clinician can select a patient, define date, time, duration, title, and Zoom link, and save the follow-up appointment.	90
6.4	Web dashboard page for nutrition plan management, including calorie target, hydration goal, and macronutrient distribution.	90
6.5	Web dashboard page for patient monitoring, showing nutritional data, water intake, last update time, and export options.	91

Onward

Onward Experience at the Motion Analysis Lab

The Motion Analysis Lab

The thesis project was carried out at the Motion Analysis Lab (MAL) of Spaulding Rehabilitation Hospital, affiliated with Harvard Medical School and Massachusetts General Hospital, in Boston, USA. The MAL is a research environment focused on biomechanics, rehabilitation engineering, and the development of innovative technologies for the analysis and recovery of human movement.

The laboratory integrates different approaches, including wearable sensing systems, motion capture technologies, and data-driven methods, with the goal of improving the assessment and rehabilitation of patients affected by neurological and musculoskeletal disorders.

Overview of Activities

During the nine-month visiting period, my work focused mainly on the development and testing of mobile software modules for wearable data acquisition, BLE communication, real-time visualization, and offline inspection of recorded signals. In parallel, I also had the opportunity to contribute to experimental activities and data collection protocols, gaining direct experience with patients and clinical monitoring systems.

Knee Sleeve Project

The primary activity of my work was developed within a broader project focused on the creation of a sensorized knee sleeve system for monitoring knee-related physiological and biomechanical parameters in patients with knee osteoarthritis (KOA). The project was based on wearable knee sleeve prototypes intended to acquire complementary signals around the knee, including temperature, bioimpedance, and flexion-extension angle. In this thesis, these signals were used mainly to guide the design of acquisition, visualization, storage, and preliminary processing modules. Within this framework, my main contribution focused on the software side of the project, in particular on the development of a mobile application capable of interfacing with the wearable devices, managing data acquisition and communication, and

supporting real-time visualization and processing. This work therefore represented the software component of a broader multimodal monitoring system, intended as a research prototype for data acquisition and visualization.

Additional Activities and Experimental Experience

In addition to the main thesis project, I was involved in several data collection sessions related to intra-articular knee injuries and post-traumatic osteoarthritis. This experience allowed me to work directly with patients and to carry out complete data acquisition protocols.

The experimental setup included kinematic analysis using optoelectronic motion capture systems, thermal assessment of the knee using infrared cameras, bioimpedance measurements using a dedicated analog front-end based on the MAX30009, and electromyography signal acquisition during different conditions, including baseline, walking, and running.

This hands-on experience was particularly valuable, as it provided a deeper understanding of the clinical context and the challenges associated with real-world data acquisition.

Additional Software Development Activities

Alongside the main project, I contributed to the development of a second application dedicated to patients with brain injuries. In this case, the application was designed not only for monitoring, but also to remote follow-up interactions between patients and clinicians.

The system aimed to assist patients in daily activities, in particular in dietary management, by providing reminders for meal consumption and supporting behavioral routines. In addition, the application included features for remote communication between patients and clinicians through integrated video calls.

A complementary web application was also developed to enable data synchronization with a central server and database, supporting remote access and data management. A brief description of this additional software development activity is reported in the Appendix.

1 Introduction

Knee osteoarthritis (OA) is one of the most common chronic musculoskeletal disorders and a leading cause of pain, disability, and reduced mobility in adults and older individuals [1, 2]. In 2019, approximately 528 million people worldwide were living with osteoarthritis, with the knee being the most frequently affected joint, accounting for around 365 million cases. This burden is expected to increase further due to population ageing, obesity, and the long-term effects of joint injury and mechanical overload [1, 3]. At the same time, healthcare systems are increasingly required to manage chronic conditions through approaches that go beyond occasional hospital visits, supporting continuous follow-up, rehabilitation, and patient self-management in daily life. Therefore, knee OA represents a significant biomedical engineering challenge. On one hand, it is a clinical condition that requires long-term management rather than a single intervention. On the other hand, its symptoms, functional limitations, and physiological changes can vary over time and are not always fully captured through episodic assessments [2]. Based on this evidence, there is growing interest in digital health strategies that combine rehabilitation, remote monitoring, and wearable sensing [4]. This chapter will introduce the clinical background of knee OA, the role of digital health and telerehabilitation in its management, and the rationale of the present work.

1.1 Background

1.1.1 Pathophysiology

Knee osteoarthritis (KOA) has traditionally been considered a passive disease related to degeneration and mechanical wear. However, current scientific evidence describes it as an active process caused by an imbalance between tissue damage and repair mechanisms [5]. KOA is not defined only by cartilage deterioration, but it is a chronic disease that involves the entire joint, including subchondral bone, synovial membrane, joint capsule, ligaments, and periarticular muscles [2, 6], as shown in Figure 1.1.

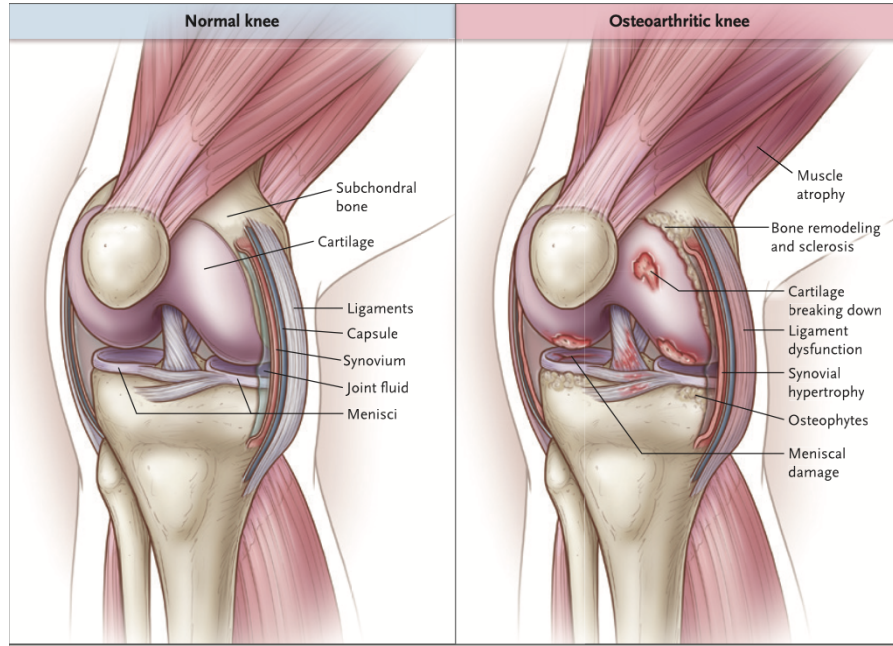


Figure 1.1: Comparison between a normal knee joint and an osteoarthritic knee, showing cartilage degradation, subchondral bone remodeling, synovial hypertrophy, osteophyte formation, ligament dysfunction, muscle atrophy, and meniscal damage[2].

The main pathophysiological mechanisms that characterize this disease are:

- **Imbalance between synthesis and degradation:** The progression of knee osteoarthritis is driven by an imbalance between anabolic and catabolic processes, with pro-inflammatory cytokines promoting the activation of cartilage-degrading enzymes and contributing to extracellular matrix breakdown. Moreover, reactive oxygen species contribute to oxidative stress, inhibit cartilage matrix synthesis, and promote the degradation of collagen and proteoglycans [7, 8].
- **Extracellular matrix (ECM) degeneration:** The damage manifests as a metabolic imbalance of the extracellular matrix. Matrix metalloproteinases (MMPs) play a key role in this process by degrading collagen and other structural components[9].
- **Subchondral bone and synovial alterations:** Bone remodeling, including subchondral sclerosis and osteophyte formation, is a primary feature that may precede cartilage degeneration and accelerate its progression. At the intra-articular level, hyperemia and synovial membrane hyperplasia are also observed [2, 6].

- **Pathological apoptosis of chondrocytes:** A key mechanism involved in the progression of KOA is the excessive apoptosis of chondrocytes. In osteoarthritic cartilage, reported levels of apoptotic chondrocytes are higher than in normal cartilage, with values ranging from approximately 2–5% in normal tissue to around 18–22% in OA cartilage [10, 11]. In regions with a high presence of apoptotic cells, there is a marked reduction in proteoglycan content, which negatively affects cartilage structure and function. Moreover, the percentage of cell death is directly related to the severity of the disease. This abnormal process is mainly regulated by mitochondrial pathways, death receptor pathways such as the Fas pathway, and endoplasmic reticulum stress. Several regulatory genes are involved, including *BCL-2*, members of the ICE family, *p53*, and *c-myc* [11].

1.1.2 Clinical Manifestations

Knee osteoarthritis (KOA) is characterized by a combination of symptoms and clinical signs that may vary in intensity depending on the progression of the disease [2]. The most common symptom is pain, which is often described as a dull and aching sensation and may involve either a specific region of the knee or a more diffuse pain pattern [12, 13]. This pain usually increases during joint use, such as prolonged walking, repetitive bending, or stair climbing, and may improve with rest in the early stages [14].

In the early stages, pain is often episodic and predictable, whereas in more advanced stages it may become less predictable and more persistent over time [12]. In severe cases, pain may also occur at rest or during the night, negatively affecting sleep quality and quality of life [12, 14].

Another common symptom is joint stiffness, particularly after periods of inactivity [2, 15]. It typically occurs in the morning and usually lasts less than 30 minutes, but it can also be experienced after prolonged sitting or rest during the day [15]. From a functional point of view, patients may experience reduced joint mobility and muscle weakness, which can progressively affect daily activities [2, 15]. Over time, these impairments may lead to reduced physical activity, fatigue, and disability, with a negative impact on activities of daily living [16]. In addition, chronic pain is often associated with psychological consequences such as depression, sleep disturbances, and an overall worsening of the patient's clinical condition [17].

From a physical examination perspective, typical findings include crepitus, bony enlargement, reduced range of motion, and joint tenderness [15]. Signs related to local inflammation, such as swelling and warmth, may also be present, especially during symptomatic flares [2, 15].

An important aspect of KOA is the discrepancy between symptoms and structural damage. Patients may report severe pain even in the presence of mild radiographic

changes, or conversely, show limited symptoms despite significant structural degeneration. This discrepancy can be explained by mechanisms such as pain sensitization, adaptation to chronic pain, and behavioral changes, including reduced activity to avoid discomfort [18].

Overall, the clinical manifestations of KOA reflect a complex interaction between structural joint alterations, functional limitations, and psychological factors, making the condition highly variable across individuals.

1.1.3 Treatment and Rehabilitation

The management of knee osteoarthritis (KOA) is mainly aimed at reducing symptoms, improving joint function, limiting disability, and maintaining a good quality of life, as no definitive cure currently exists [2, 19].

The basic treatment includes patient education, physical exercise, physiotherapy, and support in daily activities. These interventions are recommended for all patients, especially in the early stages of the disease [19].

When symptoms worsen or functional limitations increase, pharmacological therapies may be introduced. Surgical options are generally reserved for advanced cases or for patients who do not respond adequately to conservative treatment [2, 19].

Non-pharmacological treatment represents the cornerstone of KOA management. Patient education is essential to improve self-management and to promote healthy behaviors, including regular physical activity, weight control, and reduction of joint overload. These interventions are important not only for symptom management but also for maintaining long-term rehabilitation outcomes [20, 21].

Exercise therapy is one of the most important components of rehabilitation, as it can reduce pain, improve physical function, and enhance quality of life, with benefits that may persist even months after the end of the treatment. Aerobic exercise is commonly used, while muscle strengthening is particularly beneficial in the presence of weakness. Neuromuscular training aims to improve sensorimotor control and functional stability, whereas aquatic exercise allows low-impact movement, reducing joint stress[22].

Given the high variability of symptoms among patients, rehabilitation programs should be personalized. Personalization is important because adherence and outcomes may be influenced by factors such as pain intensity, fear of movement, self-efficacy, and patient expectations. Moreover, multimodal rehabilitation interventions can also be delivered remotely. Studies have shown that online education combined with prescribed exercises and pain management strategies leads to greater pain reduction and functional improvement compared to education alone [23].

Weight management is another key aspect, especially in overweight and obese patients. Recommendations include dietary control combined with physical activity and reduction of sedentary behavior. These interventions are important not only

for symptom control but also for slowing functional decline [19, 20, 21].

When non-pharmacological treatment is not sufficient, pharmacological therapy may be introduced. However, the use of medications requires caution. Non-steroidal anti-inflammatory drugs (NSAIDs) may cause gastrointestinal and renal side effects, while selective COX-2 inhibitors, although associated with lower gastrointestinal risk, may increase cardiovascular risk. Paracetamol is not recommended as a routine first-line treatment because evidence of benefit is limited; however, it may be considered for infrequent short-term pain relief when other options are contraindicated, not tolerated, or ineffective [19, 20].

From a rehabilitation perspective, KOA should also be considered in terms of its impact on daily function. Patients with lower limb OA commonly experience limitations in activities of daily living, especially in tasks that require mobility and joint loading, such as walking, rising from a chair, or stair climbing [16]. Moreover, pain-related avoidance of activity may occur in symptomatic KOA and can contribute to reduced muscle strength and, over time, to greater activity limitations [24, 25]. According to these findings, symptoms should be interpreted together with the patient's physical activity level, since pain alone may not fully reflect the functional impact or radiographic severity of the disease [26].

This highlights that rehabilitation should focus not only on pain reduction, but also on maintaining mobility and functional independence.

These considerations highlight the need for continuous monitoring and personalized rehabilitation approaches, which can be supported by wearable technologies and mobile health solutions.

1.2 Digital Health and Telerehabilitation

In recent years, Digital Health, and in particular m-Health, has gained increasing importance in the management of chronic musculoskeletal conditions, especially in diseases such as knee osteoarthritis (KOA)[27]. In these patients, traditional clinical evaluation based on periodic visits may not be sufficient to describe the evolution of symptoms and functional limitations in daily life.

Therefore, digital tools can provide an additional resource for monitoring and evaluating the progression of the clinical condition, supporting a more continuous and personalized management of the disease. The main technologies used include web platforms, mobile applications, messaging systems, phone calls, video consultations, and integrated systems connected to wearable devices[27, 28].

This aspect is central in KOA management, where physical exercise and self-management represent key components of conservative treatment. The possibility of performing part of the rehabilitation process at home can improve continuity of care and increase adherence to the rehabilitation program, especially for patients

who have difficulty maintaining frequent contact with clinicians[29, 30].

An important area of interest is the integration with wearable devices and remote monitoring systems. Wearable sensors can provide objective and repeated measurements over time related to movement, physical activity, and functional parameters, offering a complementary view of the patient’s condition compared to clinical assessments alone[31, 32]. In particular, inertial sensors have been proposed as promising tools for remote monitoring[4], movement analysis, and biofeedback in patients with hip and knee osteoarthritis[33].

At the same time, recent studies have shown good acceptance of portable technologies for frequent and remote monitoring of KOA, suggesting that the adoption of these systems is feasible, provided that usability, simplicity, and integration into clinical workflows are properly addressed[32]. However, several challenges remain, including the heterogeneity of technological solutions, the need for validation in real-world settings, and potential barriers related to digital literacy, privacy, and healthcare organization[27, 32].

Taken together, these considerations support the use of digital health and telerehabilitation in knee osteoarthritis management, as they can expand access to rehabilitation, support long-term monitoring, and enable more personalized and data-driven approaches [34]. Within this perspective, mobile systems integrated with wearable devices may provide a practical platform for combining data acquisition, visualization, and storage. The present work builds on this rationale and focuses on the development of a mobile application for a sensorized knee sleeve system aimed at multimodal knee monitoring.

1.3 Rationale for the Proposed System

Knee osteoarthritis (KOA) is a complex chronic condition characterized by symptoms that may vary over time and by functional limitations that are not always fully captured during traditional clinical assessments [2, 27].

Therefore, periodic clinical visits and standard assessment tools may provide only a partial representation of the patient’s condition, especially when symptoms fluctuate during daily life.

In particular, important changes may occur during movement, physical activity, or joint loading, but these aspects are difficult to observe continuously in a conventional clinical setting.

Since KOA involves multiple interconnected components, including pain, joint function, movement, muscle activity, and possible local tissue alterations, a single type of measurement may not be sufficient to describe the condition comprehensively. For this reason, the simultaneous observation of signals of different types could provide complementary information on movement, thermal distribution, and tissue

electrical response, which may be useful for future research on KOA monitoring after validation.

For these reasons, wearable systems represent a particularly promising solution [31, 32]. These devices can collect information outside the clinical environment and during functional activities, making them suitable for a more continuous and realistic monitoring approach.

The integration of wearable devices with a mobile application further increases their potential, because the same platform can support data acquisition, real-time visualization, storage, and processing.

1.4 Aim of the Thesis

The aim of this thesis was to design and develop a mobile application capable of connecting to wearable devices, in particular to different sensorized knee sleeves developed for the monitoring of knee osteoarthritis.

More specifically, the work focused on the implementation of a research-oriented software application able to interface with wearable sensing devices, manage Bluetooth Low Energy communication, and support handling of physiological data, including acquisition, decoding, visualization, storage, export, and post-processing. The proposed system was designed to explore how mobile and wearable technologies could complement episodic clinical assessments by enabling more frequent acquisition, visualization, and storage of knee-related physiological signals, including temperature, bioimpedance, inertial data, and EMG-derived information.

From a technical point of view, the thesis aimed to develop a mobile prototype capable of supporting heterogeneous sensing modalities related to knee joint monitoring. In its first development phase, the application was structured as a modular hub including separate workflows for thermal and bioimpedance monitoring. Within this framework, the system was designed to perform device discovery and connection, command transmission, real-time data streaming, packet parsing and decoding, signal visualization in both online and offline modes, local session storage, configurable data export, and parameter management.

A second objective of the thesis was to extend this software architecture toward a future multimodal knee sleeve scenario. Since the final integrated hardware was not yet available, this phase focused on the development of a multimodal prototype environment based on simulated integrated packets and reconstructed datasets. The goal was to define and test an entire pipeline including packet architecture design, binary parsing, CRC-based integrity verification, and multimodal signal processing. This pipeline handled different types of data, such as temperature, bioimpedance,

inertial signals, and EMG-related information, and enabled the computation of exploratory descriptors, including gait-event timing, EMG activation percentage, temperature dispersion, and frequency-dependent bioimpedance variation.

Since the final integrated multisensorized hardware was not available, the multimodal part of this work was evaluated using simulated integrated packets and reconstructed datasets. Therefore, the extracted indicators should be interpreted as technical and exploratory descriptors, not as clinically validated biomarkers or direct markers of disease progression.

The main contribution lies in the development of a modular and extensible application architecture that can support current single-modality wearable devices and, at the same time, serve as a foundation for the future integration of a fully multisensorized knee sleeve.

In this sense, the work provides a software basis for future integration with a fully multisensorized knee sleeve and for subsequent validation studies on wearable knee monitoring.

2 Materials & Methods

2.1 System Requirements

Before defining the architecture of the proposed mobile system, the main system requirements were identified in order to develop the prototype.

The requirements derived from the objectives of the project, specifically the development of a mobile application capable of connecting to a sensorized knee sleeve. Once the connection is established, the system must support the acquisition, visualization, storage, and post-processing of the physiological data received from the device.

The requirements are divided into functional requirements, non-functional requirements, and design constraints.

2.1.1 Functional Requirements

Functional requirements describe what the system must do in order to achieve the goals of the project.

In this work, they define the main functionalities required to acquire, visualize, store, and process data from connected wearable devices.

FR1. Device discovery and connection

The system must be able to scan for compatible Bluetooth Low Energy devices, establish a connection, and monitor the connection status during the session.

FR2. Command transmission

The system must be able to send commands to the wearable devices in order to start or stop the acquisition or select the operating mode.

FR3. Real-time data acquisition

The application shall receive BLE notification streams from the connected device and update the corresponding plot during acquisition, while preserving the samples required for session storage.

FR4. Support for multiple sensing modalities

The system must support different data types, including temperature, bioimpedance, inertial data, and, in the multimodal prototype, EMG signals.

FR5. Packet parsing and decoding

The system must be able to decode incoming packets, reconstruct frames, extract sensor values, and convert raw digital values into physical quantities.

FR6. Signal processing

The system must be able to apply signal processing procedures to smooth, calibrate, filter, and extract parameters and features from the acquired signals.

FR7. Signal visualization

The system must provide graphical visualization of the acquired signals, both in real time and in offline modes.

FR8. Session storage

The application must store acquired sessions locally.

FR9. Data export

The system must allow recorded sessions to be exported in standard formats for external analysis.

FR10. User configuration

The application must provide a settings module allowing the user to configure acquisition and visualization parameters.

FR11. Event annotation

The system should allow the user to annotate acquisition events for subsequent analysis.

2.1.2 Non-Functional Requirements

In addition to the core functionalities of the prototype, the system had to satisfy several non-functional requirements.

These requirements define the main quality attributes of the application, including reliability, efficiency, usability, and modularity. These requirements were defined because the application had to run on a smartphone, maintain BLE acquisition during prolonged sessions, and update plots without blocking the user interface.

NFR1. Real-time responsiveness

The application should provide responsive visualization during acquisition, with limited delay between data reception and graphical update.

NFR2. Reliability of communication

The BLE communication should be stable enough to support prolonged acquisitions with reduced packet loss.

NFR3. Computational efficiency

Since the application runs on a mobile device, the implemented algorithms should remain lightweight in terms of CPU, memory usage, and battery consumption.

NFR4. Modularity and extensibility

The architecture should be modular in order to facilitate the addition of new devices, sensing modalities, or processing functions.

NFR5. Usability

The interface should remain simple and intuitive, allowing the user to manage all operations without unnecessary complexity.

NFR6. Maintainability

The codebase should support easy updating of parameters, settings, and processing modules during future development stages.

2.1.3 Design Constraints

The development of the system was influenced by several design constraints that affected both the architectural choices and the implementation of the application.

DC1. Mobile platform limitations

The application was designed to run on a smartphone, in particular a Google Pixel 5, which imposes limitations in terms of memory, computational power, and energy availability.

DC2. Bluetooth Low Energy protocol

Communication had to rely on BLE, so the system had to work within the limitations of MTU size, packet fragmentation, and asynchronous transmission.

DC3. Heterogeneous devices

During the development phase, the application had to interface with various wearable prototypes characterized by different packet structures and sensing modalities.

DC4. Limited access to the final integrated hardware

Since the fully multisensorized wearable device was not yet available, part of the multimodal framework had to be developed and tested using simulated or reconstructed data.

DC5. Real-time execution requirements

Some processing strategies had to be simplified in online mode in order to guarantee real-time continuity of the acquisition and visualization.

DC6. Research-oriented prototype nature

The application was developed as a research prototype, so flexibility and configurability were prioritized over full commercial-grade optimization.

These requirements and constraints guided the definition of the application's software architecture.

2.2 Hardware Architecture

Although hardware development was not part of this thesis project, the application was designed to interface with a specific set of devices and datasets. For this reason, the hardware architecture is briefly introduced to contextualize the software development.

The hardware includes several sensing modules dedicated to temperature monitoring, bioimpedance acquisition, and inertial measurements. In addition, within the multimodal development, EMG and IMU data were considered to test the integration of muscular and kinematic information.

The data provided by these modules guided several software design choices, particularly with respect to communication, real-time visualization, storage, and signal processing.

2.2.1 Target Mobile Platform

The application was developed and tested on a specific target device, a Google Pixel 5.

This smartphone is equipped with a Qualcomm Snapdragon 765G processor, 8 GB of RAM, and 128 GB of internal storage. It has a 6-inch display with a refresh rate of 90 Hz, an aspect ratio of 19.5:9, and a resolution of 1080×2340 pixels.

The support for Bluetooth Low Energy 5.0 made it suitable for testing continuous

communication with wearable devices, while its hardware characteristics provided an appropriate reference platform for real-time visualization and local data management.

The target mobile platform used during the development and testing phase is shown in Figure 2.1.



Figure 2.1: Google Pixel 5 used as the target mobile platform for application development and testing.

2.2.2 Thermal Measurement Module

The device used for thermal data acquisition is not a simple point temperature sensor, but a multi-point wearable sensing array designed to estimate the spatial distribution of skin temperature around the knee. Through this device, the knee can be observed at multiple locations, allowing both the temporal evolution and the spatial distribution of temperature to be analyzed. The system was designed to detect thermal variations related to physical activity and, potentially, to inflammatory processes.

The sensing module uses 12 GA2.2K3A1iA thermistors. Each thermistor is a compact temperature sensor selected for skin temperature monitoring. The sensors

are connected through flat cables and positioned around the knee according to predefined anatomical landmarks, such as the femoral epicondyle, tibial condyle, head of the fibula, joint line, and patella, while avoiding muscle regions as much as possible.

The data are not directly acquired as temperature values. Instead, the ESP32 microcontroller reads the analog signals generated by the thermistor circuits through 12 analog input channels. Therefore, the raw data consist of ADC values from the 12 channels, which are later converted into degrees Celsius using calibration coefficients.

After acquisition, the data are transmitted wirelessly to the mobile application through Bluetooth Low Energy.

The thermal sensing module used for spatial temperature monitoring around the knee is shown in Figure 2.2.



Figure 2.2: Temperature Knee Sleeve used for spatial temperature monitoring around the knee.

2.2.3 Bioimpedance Measurement Module

The bioimpedance sensing module was designed to measure the electrical properties of the tissues surrounding the knee joint. The system operates by injecting a low-amplitude alternating current into the tissue and measuring the resulting voltage difference, allowing tissue impedance to be estimated.

The bioimpedance front-end adopted in the system is integrated on the main board, where it operates together with the microcontroller.

A multi-electrode configuration is used, with electrodes distributed around the knee and positioned along the joint line at predefined distances. This configuration improves the repeatability of the acquisitions and reduces variability related to electrode positioning.

More specifically, a tetrapolar configuration is adopted, in which two electrodes are dedicated to current injection and two electrodes are used for voltage sensing. This configuration was selected because it reduces the influence of the skin-electrode interface on the measured voltage, improving measurement accuracy and stability compared with bipolar acquisition.

After the current is injected, the measured voltage signal is alternating. Therefore, the analog front-end multiplies this signal by a known reference at the same frequency and extracts the real component, corresponding to resistance, and the imaginary component, corresponding to reactance.

Since tissue impedance is frequency-dependent, the excitation frequency plays a central role in bioimpedance measurements. At low frequencies, the current mainly flows through extracellular fluids, while at higher frequencies it can also pass through cell membranes. The operating frequency must therefore be selected by balancing penetration depth, sensitivity to the target tissue properties, and noise limitations.

Accordingly, the system can support both single-frequency acquisitions for continuous monitoring and multi-frequency measurements for a broader characterization of tissue electrical behavior.

The output of the system consists of digital values representing resistance, reactance, or impedance magnitude.

2.2.4 Inertial Measurement Module

The inertial measurement module was included in the system to provide complementary kinematic information alongside bioimpedance measurements. The aim of this module is to estimate the flexion–extension movement of the knee and to provide additional information about the subject’s motion during acquisition. This information is useful because it allows a better physiological interpretation of the bioimpedance signals and enables the identification of motion-related artifacts.

The device integrates two inertial measurement units placed on two different segments of the lower limb, namely the thigh and the shank, in order to capture the relative movement of the knee joint. Each sensing board is based on the TDK InvenSense ICM-20948, a low power 9 degrees of freedom IMU. The sensor integrates a tri-axial accelerometer, a tri-axial gyroscope, and a tri-axial magnetometer.

The IMUs communicate with the main microcontroller via a 400 kHz Fast-mode I²C interface. Accelerometer and gyroscope data are acquired synchronously, with full-scale ranges set to ± 2 g and ± 500 dps, respectively, and with internal low-pass filtering enabled. Under this configuration, both sensors operate at an output data rate of 1125 Hz. Magnetometer data are acquired at 100 Hz through the internal auxiliary I²C master. To optimize communication efficiency, sensor data are read using burst I²C transactions.

The system can operate in two different modes. In online mode, the microcontroller processes the IMU signals on board and directly transmits the flexion–extension angle together with the bioimpedance measurements. In offline mode, the raw inertial signals are preserved for subsequent post-processing.

The inertial measurement module used to estimate knee flexion–extension movement is shown in Figure 2.3.



Figure 2.3: BioZ Knee Sleeve used to estimate knee flexion–extension movement and bioimpedance.

2.2.5 Open-Source EMG-IMU Dataset

This dataset was selected because it includes synchronized thigh and shank IMU signals together with EMG recordings, allowing the Developer Mode pipeline to be tested. In particular, the IMU data from the thigh and shank provide useful information about the movement of the lower limb segments and the relative motion of the knee joint. At the same time, the EMG signal from the vastus medialis provides additional insight into muscle activation. The fact that these signals are synchronized makes the dataset especially useful for testing the proposed processing pipeline and for evaluating how kinematic and muscular information can be combined during locomotor tasks.

The dataset is an open-source resource developed by Camargo and colleagues at Georgia Tech [35]. It provides biomechanical data related to human locomotion across different environmental contexts and walking conditions. The dataset includes 22 healthy adult subjects, and the data were acquired under different walking conditions in terms of speed and terrain. From the available signals, the following data were selected for this work:

- IMU data from the shank
- IMU data from the thigh
- EMG signal from the vastus medialis

The IMUs placed on the shank and thigh are useful to describe the kinematics of the tibial and femoral segments, and therefore to indirectly estimate the motion of the knee joint. The EMG signal from the vastus medialis was selected because it represents a key muscle of the knee extensor compartment, providing information about the neuromuscular activity during walking.

The sensor locations selected from the dataset are shown in Figure 2.4. The black boxes indicate the IMU positions used in this work, while the red box indicates the EMG signal selected from the vastus medialis.

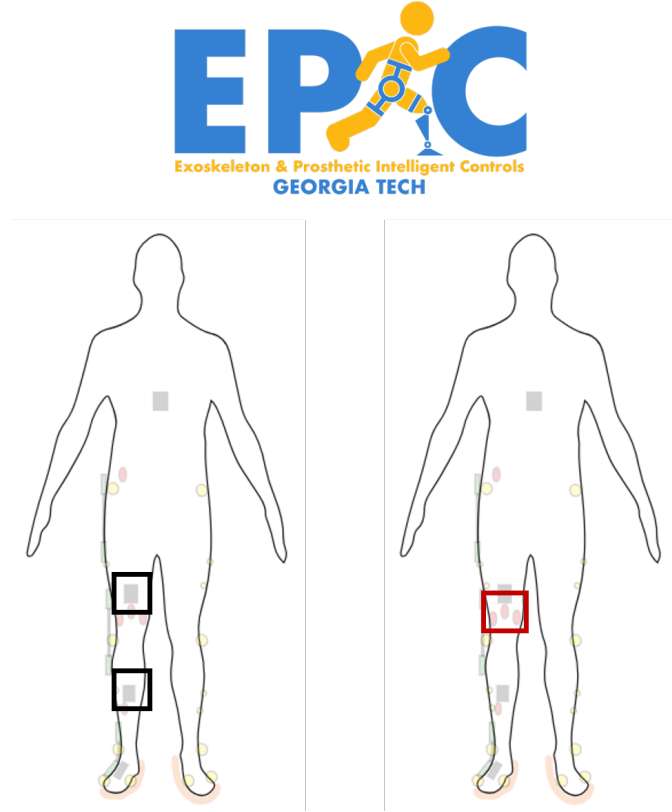


Figure 2.4: Sensor locations selected from the open-source EMG-IMU dataset. The black boxes indicate the IMU positions used in this work, while the red box indicates the EMG signal selected from the vastus medialis.

2.3 Software Architecture

The software architecture of the developed system was designed to support the integration of multiple wearable sensing modalities within a single mobile application. The architecture was organized to provide a coherent structure for the main functionalities, including device communication, data acquisition, visualization, and local data management. Particular attention was devoted to modularity and extensibility, in order to facilitate prototype development and future system evolution.

2.3.1 Overall System Design

The design of the mobile application was based on a modular structure organized around a central home interface and a set of feature-specific workflows.

The application was composed of three main functional sections: the Temperature Flow, the Bioimpedance Flow, and the Settings module. The home page acts as the main entry point of the system, allowing the user to access the available sensing modules and navigate between the different application areas.

After the user selects a sensing module, the application initiates Bluetooth Low Energy communication with the corresponding wearable device, configures the required characteristics, and redirects the user to the available operating modes. In this way, each sensing modality is managed through a dedicated workflow, while maintaining a coherent interaction logic across the entire system.

From the home page, the user can also access previously recorded data without establishing a Bluetooth connection. This enables offline visualization and analysis modes that do not require real-time communication with the wearable device.

The Temperature Flow and the Bioimpedance Flow were implemented as two separate high-level branches of the application. Each branch includes multiple operating modes tailored to the corresponding sensing modality. In the temperature module, these include real-time acquisition, grid-based visualization for thermal mapping, and offline visualization. In the bioimpedance module, the workflow includes an initial calibration phase followed by different operating modes, such as real-time acquisition, sweep analysis, and session visualization.

A shared Settings module centralizes the main configuration parameters of the application. This centralized layer ensures coherent behavior across the different workflows and simplifies the management of parameters used by multiple screens. The system follows a hub-and-module logic, in which the home page provides access to the main sensing workflows. This organization is consistent with the research-oriented nature of the prototype, as it supports the integration of heterogeneous wearable devices while maintaining a clear and extensible software structure. A block diagram summarizing the software structure and the relationships between the main application modules is shown in Figure 2.5.

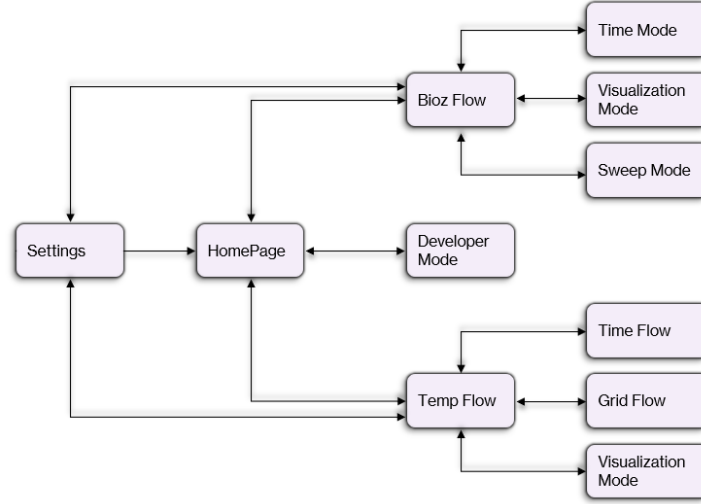


Figure 2.5: Block diagram of the software architecture, showing the main application modules and their organization into functional workflows.

2.3.2 Framework and Libraries

The software infrastructure of the developed system was based on React Native, a JavaScript framework widely used for the development of cross-platform mobile applications. This choice was motivated by the need to create a single application prototype that could be rapidly developed while maintaining a modular and maintainable architecture.

Expo was adopted as the development environment because it provides an efficient ecosystem for prototyping, testing, and iterative refinement. It supported the development process through tools for project configuration, dependency management, execution, and debugging, which were particularly useful during the early implementation stages.

Among the adopted tools, Expo CLI was used to initialize the project from the command line, run the development server, install compatible packages, and generate application builds. In addition, a development build was preferred over the standard Expo Go environment because it allowed the integration of custom native code and native modules required by the application. This was particularly important for functionalities related to Bluetooth Low Energy communication and advanced interaction with mobile device resources.

To support the main application functionalities, a set of external libraries was integrated into the React Native project. These libraries were selected according to the architectural requirements of the system, including Bluetooth communication with wearable devices, navigation between screens, state management, local data

storage, and real-time signal visualization. The main libraries adopted in the developed application and their respective roles are summarized in Table 2.1.

Table 2.1: Main software libraries adopted in the mobile application.

Library	Purpose
<code>react-native-ble-plx</code>	Bluetooth Low Energy communication with wearable devices.
<code>@react-navigation/native</code>	Core navigation infrastructure for screen management.
<code>@react-navigation/stack</code>	Stack-based navigation between application pages.
<code>react</code>	Component logic, context, and shared state management.
<code>@react-native-async-storage</code>	Persistent local storage of settings, metadata, and acquired data.
<code>react-native-svg-charts</code>	In-app visualization of sensor signals and temporal trends.

These libraries provided the basic software components required to implement wireless communication, screen navigation, data persistence, and graphical visualization within a single mobile application.

2.4 Bluetooth Low Energy Communication

2.4.1 BLE Communication Architecture

The management of wireless communication through the Bluetooth Low Energy protocol was implemented using a dedicated React Native library, namely *react-native-ble-plx*. This library relies on the creation of a *BleManager* object, which acts as the central communication layer of the application.

The *BleManager* object is responsible for device discovery, connection management, and data exchange. To avoid creating a new object in each screen, it is maintained at the main flow level and shared across the application through React contexts. In particular, two main contexts are defined: *BleCtx* for the temperature module and *CalibCtx* for the bioimpedance module.

Wireless communication is organized into two separate workflows: one dedicated to the Temperature Knee Sleeve and the other to the Bioimpedance Knee Sleeve. This separation allows each workflow to reuse an already initialized BLE connection, avoiding repeated connection procedures across different screens.

In the **temperature workflow**, the main flow manages the BLE connection and selects the characteristic of interest. All related screens access shared information such as device ID, connection state, and characteristic through the context, and are responsible only for starting or stopping the notification stream. In this case, the BLE communication is structured as a persistent shared connection focused on data streaming.

In the **bioimpedance workflow**, the *BioZFlow* exposes a context that includes both the BLE state and the calibration state. The modules reuse the same connection but introduce an additional configuration phase, including device setup and dynamic selection of characteristics. In this case, BLE communication is used not only as a data channel, but also as a control channel.

In summary, the temperature workflow relies on a persistent connection with a predefined characteristic, while the bioimpedance workflow extends this approach by including calibration procedures and dynamic control of the operating modes.

2.4.2 Device Discovery and Connection Establishment

The application was developed for devices running the Android operating system. Therefore, at the beginning of the connection workflow, a permission management routine is initialized. In particular, for Android 12 and later versions, the application requests the *BLUETOOTH_SCAN*, *BLUETOOTH_CONNECT*, and *ACCESS_FINE_LOCATION* permissions. For earlier Android versions, only the *ACCESS_FINE_LOCATION* permission is required.

In both workflows, the function *startDeviceScan* is called to initiate device discovery. The scan is performed by filtering devices according to their name, and a safety timeout of 10 seconds is applied. This means that the application does not scan indiscriminately for all available BLE devices, but explicitly searches only for the wearable devices of interest. This design choice improves usability by avoiding device selection screens that were present in earlier versions of the application.

In particular, the identifiers of the target devices are *TemperatureKneeSleeveDummy* for the thermal sleeve and *BIOZ-IMU* for the bioimpedance module.

Once a device is found, the application executes the *connect()* function, followed by *discoverAllServicesAndCharacteristics()*. This sequence allows the application to retrieve the GATT structure of the wearable device before enabling notifications or sending commands.

In the temperature workflow, after service discovery, the application attempts to request an MTU size of 247 bytes on Android, if supported, and increases the connection priority. The system then scans the available services and characteristics, automatically selecting a characteristic that contains *FF01* and supports notifications or indications.

In the bioimpedance workflow, the connection phase is simpler. After service discovery, the application stores the device ID and connection state in the shared context. Advanced configuration is handled in subsequent screens, where calibration procedures and operating mode selection are performed. Considering the workflow, BLE communication is managed as an ordered sequence of permissions, device discovery, service discovery, and configuration steps. This defines a structured workflow for establishing and maintaining communication with the wearable devices.

2.4.3 Device Configuration and Operating Modes

The bioimpedance module defines a set of specific operating modes, namely idle mode, real-time mode, instant mode, IMU calibration mode, and angle calibration mode. These modes are sent to the device by writing a base64-encoded byte to the characteristic *FFF3*, which acts as a control characteristic.

The bioimpedance workflow includes two sequential calibration steps:

1. IMU calibration
2. Angular calibration

During the first phase, the application writes the IMU calibration mode to the device, waits for the required duration, and then sets the device back to idle mode. The result of the calibration is then stored in *AsyncStorage*. The same procedure is repeated for the angular calibration phase.

Access to the operational modes of the bioimpedance sleeve is allowed only when the required calibration steps are successfully completed. The *BioZ* mode selector enables real-time acquisition and the other operating modes only if both calibration phases are valid. This ensures that the workflow remains consistent with the importance of the initial system state for both bioimpedance and angular measurements. In contrast, the temperature workflow does not require writing specific operating modes to the device. Once the connection is established, the user can directly select between acquisition and visualization modes without additional configuration steps.

2.4.4 Real-Time Data Streaming

In the case of the temperature sleeve, the acquisition and visualization modules first verify the availability of the BLE connection through the *connected*, *deviceID*, and *characteristic* objects. Only when all these elements are valid, the monitoring process is started using the function *monitorCharacteristicForDevice()*.

The value received from the characteristic is encoded in base64. It is first converted into raw bytes and then passed to a parsing pipeline that distinguishes three possible cases:

- ASCII header messages
- Binary payloads composed of multiple 26-byte frames
- Continuous byte streams delimited by STX/ETX markers

Incoming BLE notifications are therefore parsed according to three possible formats. ASCII messages are interpreted as header information, for example to extract parameters such as the sampling frequency. When the received binary payload length is an exact multiple of 26 bytes, the packet is directly decoded as one or more fixed-size frames, each composed of a temporal index and 12 sensor channels. If the payload does not match this condition, a buffered parser based on STX/ETX delimiters is used to reconstruct valid frames from the continuous byte stream. Thanks to this, the fixed-size frame decoding represents a fallback mechanism that allows correct parsing even in the absence of explicit delimiters.

In the case of the bioimpedance module, the data acquisition flow follows a structured sequence of operations:

1. The real-time mode is written to the device.
2. Buffers, timeline, and plot state are reset.
3. Session metadata are initialized.
4. An MTU request of 247 bytes is attempted on Android.
5. Service and characteristic discovery is performed.
6. Notifications are enabled on the selected data characteristic.

2.4.5 Data Transmission Protocol

For the temperature sleeve, the communication protocol is based on frames composed of 13 words of 16 bits. One word corresponds to the temporal index, while the remaining 12 correspond to the sensor channels. Since each word is encoded as a 16-bit integer in little-endian format, each frame has a total size of 26 bytes.

As previously described, the parser includes a fallback mechanism to ensure robustness. However, the preferred parsing strategy relies on the presence of two delimiters, STX = 0x02 and ETX = 0x03, within which a complete frame is expected.

For the bioimpedance module, the protocol is based on data transmitted in blocks

of 240 bytes. Each block contains 15 samples, each of 16 bytes. Within each sample, the data include flexion–extension angle, resistance, reactance, and a temporal counter. The first three quantities are encoded as *float32*, while the temporal value is encoded as *uint32*, all in little-endian format.

In the bioimpedance module, time is not transmitted as an absolute timestamp but as ticks, with a conversion factor of 1 tick = 10 microseconds. The application reconstructs a continuous relative timeline and handles possible rollovers or resets of the counter by updating a temporal offset.

Therefore, while both devices share the same BLE transport layer, they implement different application-level protocols.

2.4.6 Connection Monitoring and Robustness

Both workflows implement a listener based on *onDeviceDisconnected*. When a disconnection occurs, the BLE state is reset. This represents the first level of robustness in handling interrupted or inconsistent connections.

In addition, a periodic watchdog mechanism is implemented. In the temperature workflow, every 1.5 seconds the system verifies that the BLE connection is still valid. In the bioimpedance workflow, the watchdog checks the list of connected devices to ensure that the expected device is still present. In both cases, if the check fails, the BLE state is set to disconnected.

In the temperature module, if the provider indicates that the connection is no longer ready, namely when the *isReady* flag becomes false, the data stream is automatically stopped and the temporal series displayed in the acquisition screen is reset. This prevents the visualization of inconsistent data in case of connection loss.

In the bioimpedance module, in addition to disconnection handling, an internal mechanism ensures correct reconstruction of data blocks even when a single BLE notification does not correspond to a complete block. The parser expects blocks of 15 samples, each composed of 16 bytes, for a total of 240 bytes.

Since BLE notifications do not guarantee alignment with block boundaries, a notification may contain only a portion of a block or multiple concatenated blocks. For this reason, the parser does not immediately interpret each notification as a final block. Instead, incoming bytes are accumulated in an internal buffer. Once the buffer reaches at least 240 bytes, the first 240 bytes are extracted and decoded. This strategy ensures correct parsing of the data stream, prevents the processing of incomplete blocks, and avoids data loss due to fragmentation at the BLE transport level.

2.5 Data Management and Storage Architecture

2.5.1 Session Model & Organization

Each recorded session is not simply represented as a file, but as a logical acquisition unit that contains multiple elements:

- Metadata, such as device ID, sampling rate, and timestamp
- Time-series data
- Marks and events inserted by the user
- A time-based unique identifier

For the temperature module, the session is reconstructed at the moment of saving using a structured format that includes timestamp, device name, sampling rate, total number of packets, time data, channel data, marks, and a metadata block. The metadata block contains information such as the number of channels, the maximum number of points, and the temporal window.

The session name is generated using the format `acquisition_YYYY-MM-DDTHH_mm_ss|` allowing each acquisition to be uniquely identified and chronologically ordered.

For the bioimpedance module, sessions are organized differently, as they are designed for longer acquisitions. Each session contains two main signal blocks, namely bioimpedance data, including resistance and reactance, and knee flexion–extension angle. These data are stored together with the total duration, number of samples, and information related to the chunked storage process.

The session identifier is generated using the format `realtime_<sessionId>`, ensuring uniqueness across acquisitions.

In summary, a session represents a container for a recording, designed not only for real-time visualization but also for offline analysis and data export.

2.5.2 Local Persistence Strategy

For the temperature sleeve, local storage is implemented using AsyncStorage. The payload of each session is serialized in JSON format and saved with a dedicated key following the pattern `acquisition_<filename>`. In parallel, a lightweight index called `saved_acquisitions` is updated. This index contains only the minimal information required to display the list of stored sessions, such as filename, timestamp, device name, sampling rate, number of samples, and number of markers.

The full payload is used for reopening and analyzing a session, while the compact index is used for fast navigation within the local archive. This design avoids loading the full dataset every time the user accesses the session list, thereby improving

efficiency and responsiveness.

For the bioimpedance sleeve, data are not stored entirely in RAM until the end of the acquisition. Instead, samples are accumulated in temporary buffers and written in chunks of 1000 samples. Each chunk is saved with a dedicated key, while a separate metadata structure keeps track of the session ID, chunk size, and number of chunks already written.

At the end of the acquisition, all chunks are reassembled into a single payload, which is then indexed in *saved_realtime_sessions*. The use of chunked storage is not intended to segment the data logically, but to improve robustness during long and continuous acquisitions by reducing the dependency on a single large object stored in memory.

Each module enforces a maximum number of stored sessions, defined by the parameter *maxSaved*. This mechanism prevents issues related to the hardware limitations of the mobile device by avoiding excessive memory usage and requiring the user to delete older sessions before saving new ones.

2.5.3 Session Export and File Handling

After data acquisition, in the temperature visualization module, the application reads the *saved_acquisitions* index, sorts the sessions chronologically, and, when the user selects a session, retrieves the full payload using the key *acquisition_<filename>*.

At this point, the data can be exported in both CSV and JSON formats.

The CSV file contains the temporal axis (*time_s*), the 12 sensor channels, and an additional column for user-defined marks. The marks are associated with the closest sample in time with respect to the moment in which they were inserted, ensuring that event information remains aligned with the data timeline. The export of the CSV file depends on the current visualization state, meaning that the data are exported according to the representation mode selected at the time of export. The JSON file preserves a structure consistent with the locally stored session, making it suitable for further analysis and data reloading. The application also manages the export path according to the platform and verifies the required write permissions on Android.

For the bioimpedance module, sessions are instead exported in *.txt* format, using a simple tabular structure that includes impedance data and knee angle values, making the file easily readable and compatible with external analysis tools.

2.6 Signal Processing and Computation Methods

The acquired signals were processed using different procedures according to the characteristics of each sensing modality. Since the application was designed to

support both real-time visualization and offline analysis, the processing methods were selected by considering signal quality, computational cost, and temporal behavior.

For real-time visualization, lightweight filtering strategies were preferred, while zero-phase filtering procedures were used during offline processing, where the whole signal segment was available.

2.6.1 Temperature Signal Processing

Temperature signals are characterized by slow variations over time and may contain small fluctuations due to measurement noise or minor instability during acquisition. For this reason, a low-pass filtering approach was adopted to improve signal readability and obtain smoother temporal trends.

The filtering procedure was based on a Butterworth low-pass IIR filter. This type of filter was selected because it provides a smooth frequency response without oscillations in the passband, making it suitable for visualization purposes and for preserving the slow evolution of the temperature signal.

In the implemented system, the filter was realized using a digital biquad structure. The single-pass transfer function can be expressed as:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}.$$

For offline visualization and post-processing, the filter was applied in both the forward and backward directions. This method was adopted to reduce phase distortion and avoid temporal shifts of the signal. A standard causal filter may introduce phase delay, whereas forward-backward filtering compensates for this effect and produces a zero-phase response.

The effective filtering operation can therefore be written as:

$$H_{\text{eff}}(z) = H(z) H(z^{-1}).$$

As a result, the signal is smoothed while preserving the temporal alignment of its main variations. This processing step was mainly used to improve the readability of the temperature trends during offline visualization and analysis.

The general effect of the filtering procedure on the temperature signal is represented in Figure 2.6.

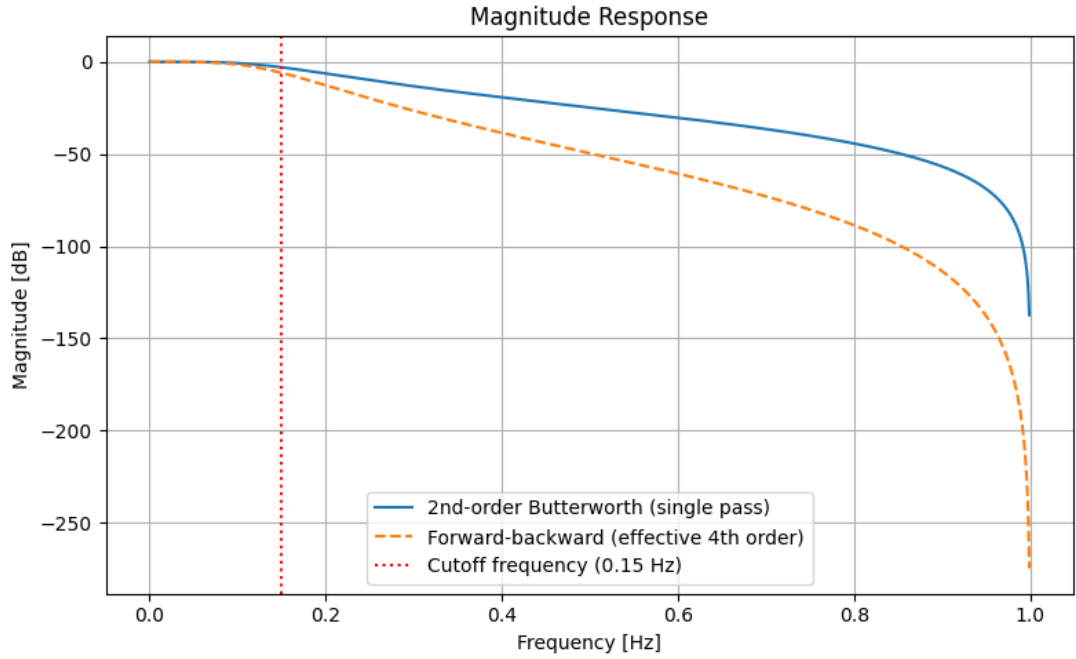


Figure 2.6: Magnitude response of the low-pass filtering applied to the temperature signal. The 2nd-order Butterworth filter and the effective forward–backward response are shown, with the cutoff frequency set at 0.15 Hz.

2.6.2 Bioimpedance Signal Processing

For the analysis of bioimpedance signals, and in particular for their visualization, an exponential moving average (EMA) filter was adopted. Bioimpedance signals can exhibit small rapid oscillations and visual jitter, which make the signal less readable during continuous acquisition.

The EMA filter was selected to improve graphical stability without introducing excessive computational complexity. This aspect is particularly important in a mobile application, where the processing method must remain lightweight in terms of memory usage and computational cost.

The EMA is a first-order low-pass IIR filter. It is suitable for real-time applications because the filtered value is updated using only the current sample and the previous filtered value. In this way, the signal can be smoothed sample by sample without storing an entire window of past data.

The filter can be expressed as:

$$H(z) = \frac{1 - \alpha}{1 - \alpha z^{-1}}.$$

Unlike a moving average filter based on a fixed window, the EMA assigns exponentially decreasing weights to past samples. As a result, more recent samples have a stronger influence on the output, while older samples progressively contribute less. The main parameter of the filter is α , which controls the trade-off between signal stability and responsiveness to variations. Higher values of α produce a smoother signal, while lower values allow the filter to follow rapid changes more closely. Therefore, the EMA filter was used as a lightweight smoothing strategy for real-time visualization of resistance, reactance, and impedance-related signals. The smoothing effect of the EMA filter on the bioimpedance signal is represented in Figure 2.7.

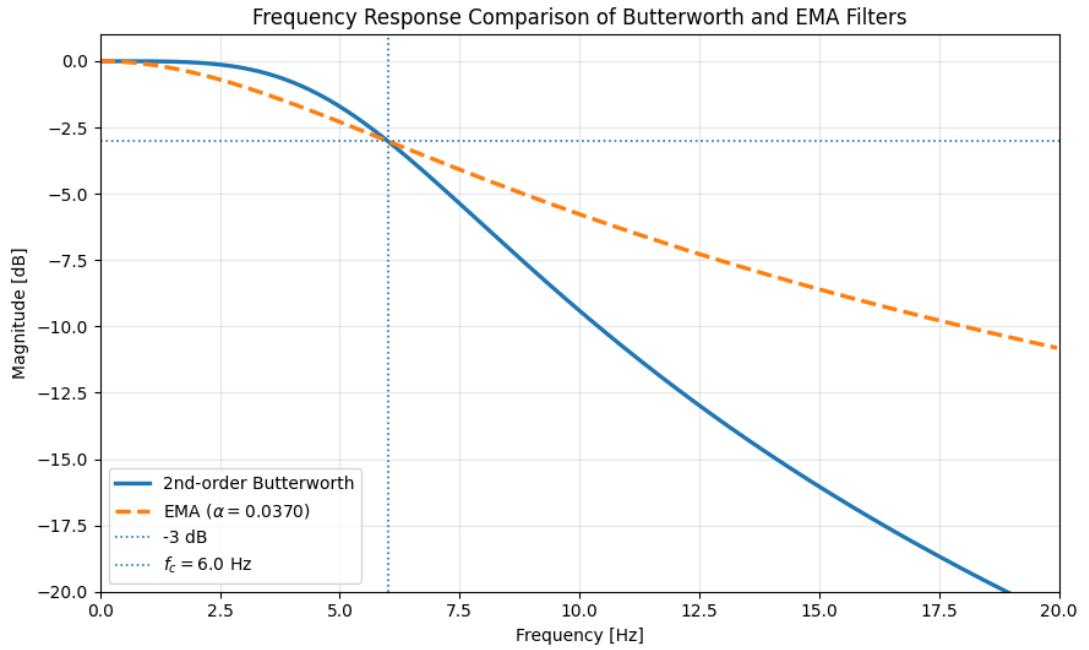


Figure 2.7: Frequency response comparison of the filters considered for bioimpedance signal preprocessing. The 2nd-order Butterworth filter and the exponential moving average filter are shown, with the -3 dB cutoff frequency set at 6 Hz.

2.6.3 EMG Signal Processing

The acquired EMG signal was processed through an offline pipeline aimed at extracting the signal envelope, which provides an indirect representation of muscle activity.

First, the raw signal was centered by removing the DC component through median subtraction. This choice was adopted to reduce the influence of outliers and

impulsive artifacts compared with a mean-based correction.

The centered signal was then filtered using a fourth-order Butterworth band-pass filter with cutoff frequencies of 20 Hz and 450 Hz. This step isolates the main EMG frequency content by reducing both low-frequency motion artifacts and high-frequency non-physiological components. Since the processing was performed offline, the filtering was applied in zero-phase mode using a forward-backward approach, avoiding temporal distortions of the signal.

After filtering, the signal was full-wave rectified by computing the absolute value of each sample. This step makes all amplitudes positive and allows the signal amplitude to be related to the intensity of muscle activity.

Finally, a fourth-order Butterworth low-pass filter with a cutoff frequency of 6 Hz was applied, also in zero-phase mode, to extract the EMG envelope. Through this processing pipeline, the slow-varying trend of the signal is extracted, while rapid fluctuations are reduced.

The transfer functions of the filters used to filter these signals are shown in Figure 2.8.

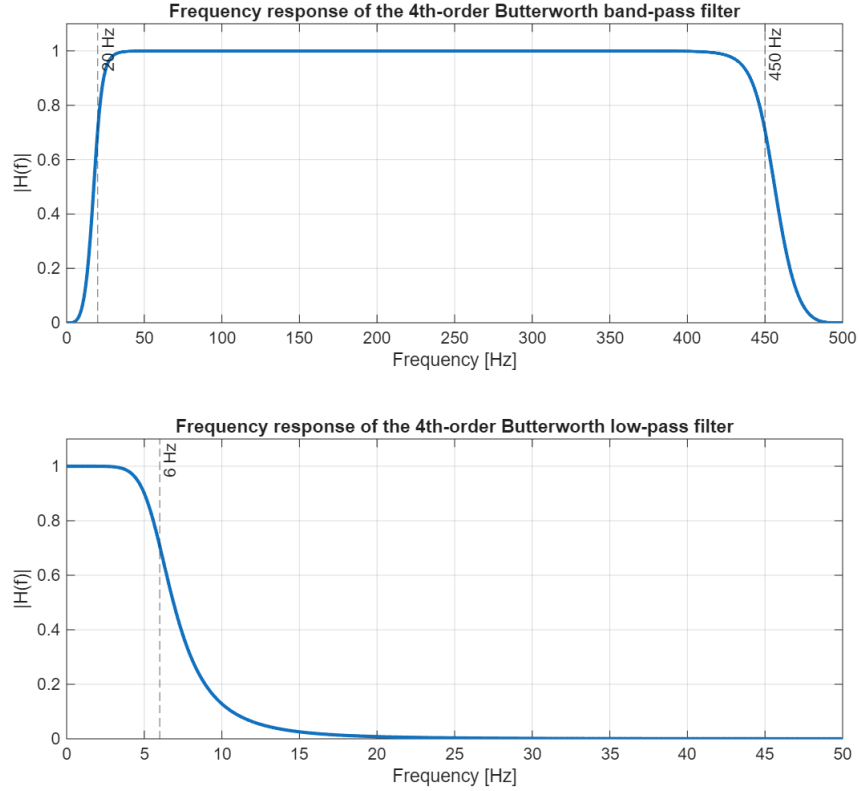


Figure 2.8: The top figure shows the bandpass filter, which is the first filter used to isolate the desired signal, while the bottom figure shows the low-pass filter used to isolate the envelope.

2.6.4 Gyroscope Signal Processing

Gyroscope data were used to support the analysis of knee flexion–extension movement during locomotor tasks. Since this movement is mainly represented by angular velocity variations along one dominant axis, a preliminary axis selection step was performed to identify the most informative gyroscope component.

To determine the most relevant gyroscope axis, the signal was first resampled to 50 Hz in order to obtain a uniform sampling frequency and reduce the computational cost of the subsequent processing steps. Subsequently, the median of each gyroscope axis was computed and subtracted in order to center the signal around zero.

This operation removes the DC offset or slow bias of the axis, allowing the signal to oscillate around zero. This step is important because it makes positive and negative angular velocity variations easier to compare and interpret.

After centering, the signal was filtered using a low-pass exponential moving average filter implemented in zero-phase mode, with an effective cutoff frequency of 8 Hz. Unlike the EMA used for real-time bioimpedance visualization, this filter was applied both forward and backward because the gyroscope processing was performed offline. This method reduces phase distortion and preserves the temporal position of the main movement-related variations.

The objective of this filtering step is to attenuate noise and rapid oscillations while preserving a clear and interpretable angular velocity profile. In this way, the processed gyroscope signal provides a more stable representation of the flexion–extension movement.

3 Results

3.1 Phase 1 – Multi modalities App Hub

3.1.1 Multi-Modality Application Hub

The first phase of the project was based in the development of a multi-modality application hub designed to manage different wearable devices within the same mobile environment.

The main functionalities implemented in this research prototype were aimed at providing a stable and simple BLE connection, supporting reliable data transfer with reduced packet loss, and allowing the user to configure selected parameters related to data acquisition and visualization. Since the prototype was designed for research purposes, protocol flexibility and configurability represented key aspects of the development process.

The application hub developed during the first phase is shown in Figure 3.1. The two main modules that have been developed are the temperature module and the bioimpedance module. In addition, a settings module has been implemented to manage the settings and parameters associated with the aforementioned modules. In this chapter, we will analyze all the modules, describe their functions within the overall ecosystem, and provide a detailed description of all the modes contained within each of these modules.

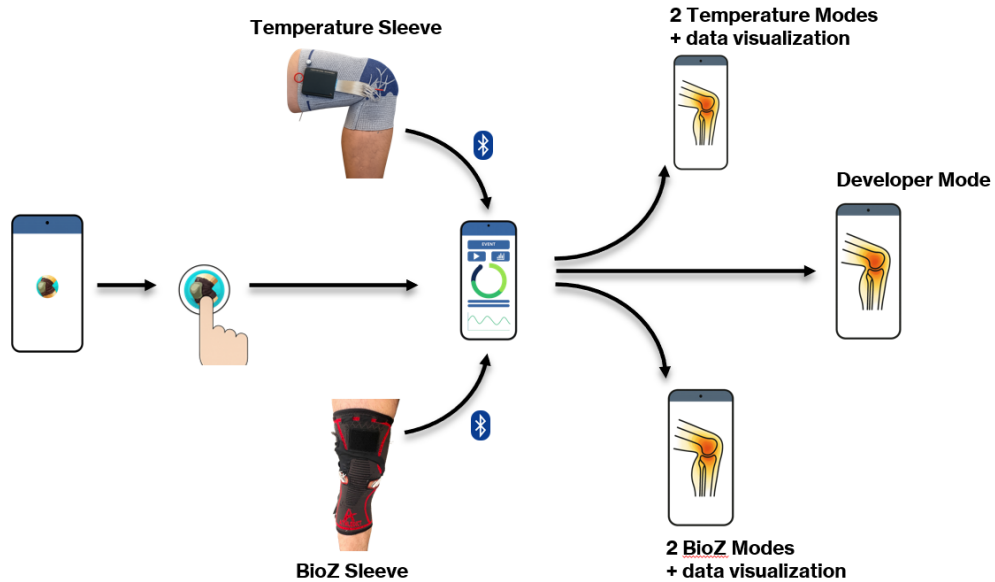


Figure 3.1: General structure of multi-modality application hub developed during the first phase of the project.

3.1.2 Temperature Module

The temperature module was implemented to manage the acquisition, visualization, and analysis of data coming from the 12 thermistors embedded in the Temperature Knee Sleeve.

The module includes two main operating modes. The first one is the Real-Time Mode, which allows the user to acquire and visualize the temporal evolution of the 12 temperature channels. During the initial acquisition phase, the data already stored in the sleeve from the moment it was turned on are transmitted to the application at the maximum speed allowed by the BLE communication. After this initial transfer, the visualization continues in real time at a frequency of 2 Hz.

The second operating mode is the Grid Mode, which provides a spatial visualization of the temperature distribution around the knee. In this mode, the temporal evolution of the thermal map can be observed together with additional parameters extracted during the acquisition.

Both modes work with a calibration dictionary, shown in Table 3.1, used to convert the ADC values collected by the microcontroller into temperature values expressed in degrees Celsius. The calibration file was obtained in collaboration with other collaborators at the Motion Analysis Lab, who specifically worked on the hardware development of the Temperature Knee Sleeve.

An overview of the temperature module and its main operating modes is shown in Figure 3.2.

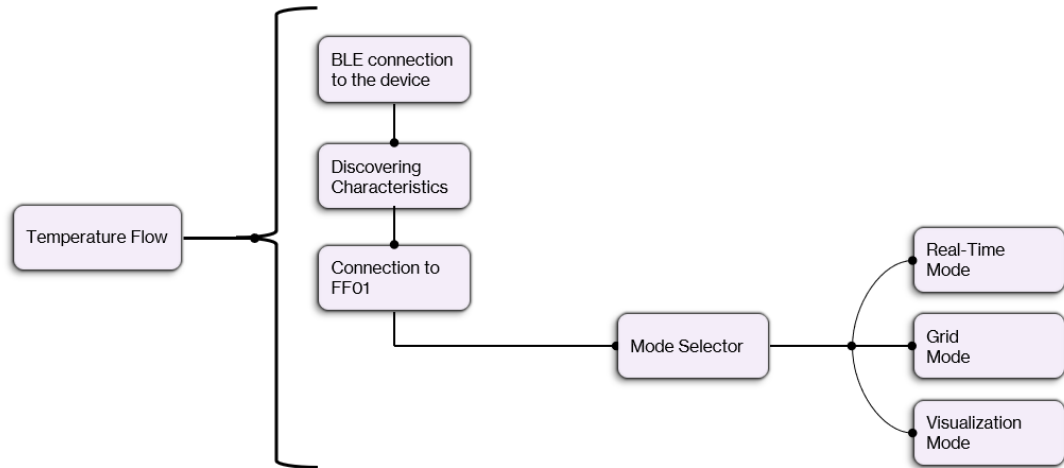


Figure 3.2: Overview of the temperature module, including the main operating modes implemented for acquisition and visualization.

The conversion from ADC values to temperature was performed using the following logarithmic calibration model:

$$T = a + b \cdot \ln(V_{\text{ADC}}) + c \cdot (\ln(V_{\text{ADC}}))^3 \quad (3.1)$$

where T is the estimated temperature, V_{ADC} is the acquired ADC-related value, and a , b , and c are the calibration coefficients associated with each channel.

Table 3.1: Calibration coefficients used to convert ADC values into temperature values for the 12 thermistor channels.

Index	Channel	Coefficient a	Coefficient b	Coefficient c
0	ADC1_0	120.3	−6.894	−0.097,530
1	ADC1_3	157.3	−14.4	−0.051,470
2	ADC1_6	116.9	−6.383	−0.098,220
3	ADC1_7	150.4	−12.55	−0.067,630
4	ADC2_0	163.5	−14.81	−0.054,150
5	ADC2_3	212.2	−23.55	−0.013,480
6	ADC2_4	216.7	−24.46	−0.007,484
7	ADC2_5	179.1	−17.78	−0.038,690
8	ADC2_6	221.5	−25.25	−0.004,378
9	ADC2_7	246.2	−30.66	0.031,940
10	ADC2_8	184.6	−18.9	−0.031,430
11	ADC2_9	171.9	−15.97	−0.052,810

3.1.2.1 Real-Time Mode

The Real-Time Mode was implemented to provide a temporal overview of the 12 channels corresponding to the thermistors embedded in the Temperature Knee Sleeve.

In this mode, incoming BLE packets are decoded using the same robust parsing strategy described in the previous chapter. The parser supports frames delimited by STX/ETX markers and includes a fallback mechanism based on fixed 26-byte blocks. Each valid frame contains the 12 sensor channels, encoded as 16-bit integers in little-endian format.

The acquired data are displayed using a dynamic windowing approach, in which a scrolling window of the last 60 seconds is shown. The interface provides a clean real-time visualization of the 12 channels, initially represented as ADC values collected by the microcontroller. When the acquisition is started through the dedicated button, the data are not only displayed in real time but also automatically recorded.

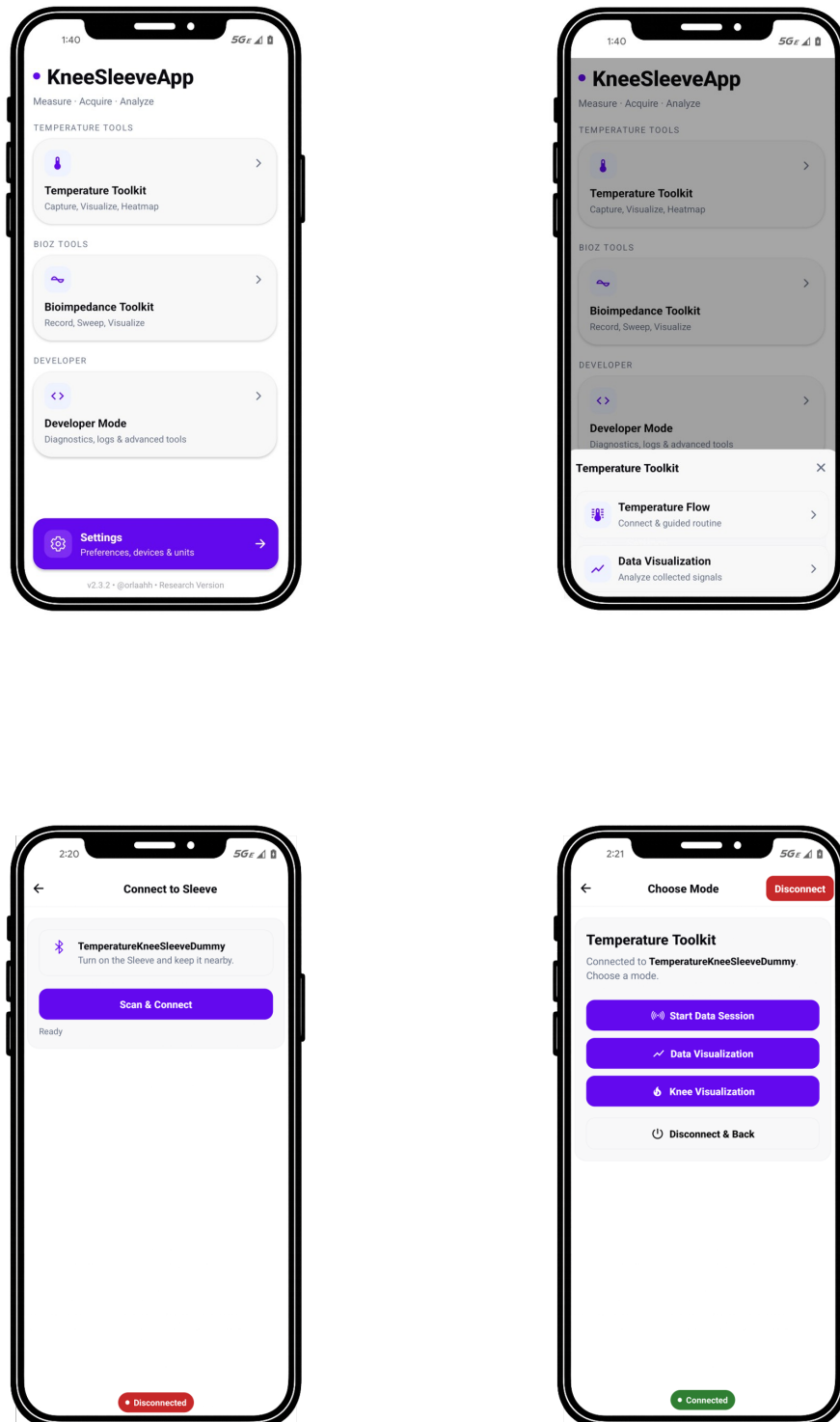


Figure 3.3: Temperature module interfaces showing the main visualization options and acquisition-related screens.

Several interaction options were implemented to support initial processing of the acquired signals. A dedicated button allows the user to switch the visualization from ADC values to temperature values expressed in degrees Celsius, according to the calibration coefficients previously introduced. A second option allows the user to visualize filtered data by applying a low-pass filter to the visible time window. The filtering option was designed to improve signal readability during visualization. Since the filter is applied to the displayed window, the approach prioritizes the absence of phase delay and preserves the temporal alignment of the signal variations. However, transient effects may appear at the edges of the moving window, especially for the most recently received samples, which do not yet have future samples available for the backward filtering step.

An additional visualization option was implemented to display the Heat Distribution Index(HDI), which provides a synthetic indication of the spatial variability of temperature across the 12 channels. For each time sample, the HDI is computed across the active temperature channels according to:

$$HDI(t) = 2 \sigma_{\text{channels}}(T_1(t), T_2(t), \dots, T_{12}(t)) \quad (3.2)$$

where $T_i(t)$ represents the temperature measured by the i -th channel at time t . When this option is enabled, the visualization of the individual channels is replaced by the HDI trend.

The mode also includes an event annotation function. Through a dedicated button, the user can mark a specific time instant during acquisition. The marked event is represented graphically on the plot and stored within the session recording for later analysis.

At the end of the acquisition, the application verifies that data are available and generates a unique file name based on the local timestamp. The entire session is then saved as a JSON file, including temporal and hardware-related metadata such as timestamp, device name, connected Bluetooth device, and actual sampling frequency. The data object is serialized into a JSON string and stored in AsyncStorage using the generated file name as a unique key.

To improve navigation across saved recordings, the application stores the acquisition payload separately from a lightweight history index. After saving the main data file, the system updates the *saved_acquisitions* list, which acts as a registry of the recorded sessions.

3.1.2.2 Offline Visualization and Data Export

An offline visualization module was implemented to allow the user to visualize and analyze the recordings generated during the Real-Time Visualization Mode.

After a recording is selected, the application loads the session and applies the same conversion procedure used during acquisition. The ADC values are converted

into temperature values using the calibration model previously introduced. The Heat Distribution Index (HDI) is then computed for each time instant as twice the standard deviation across the active temperature channels. This metric provides a synthetic indication of thermal variability, allowing the user to evaluate possible temperature imbalances without inspecting all twelve curves simultaneously.

The user can visualize the twelve temperature channels and selectively enable or disable specific signals in order to focus on selected sensor locations. As in the acquisition mode, filtered data can also be displayed to facilitate signal inspection. The filtering option is based on a second-order Butterworth low-pass IIR filter implemented as a digital biquad structure. The default cutoff frequency is set to $f_c = 0.15$ Hz, while the filter order is set to 2. The quality factor is defined as $Q = 1/\sqrt{2}$, corresponding to a Butterworth response. The sampling frequency is obtained from the session metadata when available; otherwise, it is estimated from the time vector, with a fallback value close to 2 Hz.

To avoid temporal shifts in the displayed signal, the filter is applied in a forward-backward configuration, following a zero-phase filtering approach. In this way, the filtered visualization improves signal readability while preserving the temporal alignment of the main signal variations.

To support smooth and interactive exploration of recorded signals, the module uses a vector-based rendering system optimized for mobile devices. Since cross-platform applications may suffer from delays caused by continuous data exchange between the application logic and the graphical engine, touch interactions such as scrolling and pinch-to-zoom gestures are handled directly by the native user interface thread. This solution allows the graphical interface to remain responsive during navigation across the dataset, maintaining fluid transitions during user interaction. When large recordings are loaded, the number of points to be rendered can become very high. To avoid excessive memory usage and preserve responsiveness, the system applies a downsampling strategy that limits the number of points displayed on screen while preserving the shape of the curves.

For long recordings, a static downsampling strategy is applied over the entire temporal domain. The maximum number of renderable vertices, defined as $N_{\max} = 10,000$, is uniformly distributed across the full duration of the recording during the loading phase. This allows the application to maintain stable performance even when long sessions are visualized.

In this analysis mode, the time axis does not start only from zero, but can also be reconstructed as an absolute temporal reference. The system uses the name of the saved file, which corresponds to the timestamp of the end of the acquisition, to estimate the actual time at which the recording started. In this way, the recorded data can be aligned with external logs or video recordings of the experiment.

Event markers generated during acquisition are also reconstructed in the offline visualization. The timestamp associated with each marker is used to locate the

closest sample in the dataset by computing the minimum temporal difference between the marker timestamp and the sampled time values.

Finally, the module allows the user to export the dataset as a CSV file generated directly from the internal data matrix. The exported file includes the relative time axis, the reconstructed absolute time axis, the sensor values, and the event markers represented as a categorical column.

Examples of the offline visualization and export interface are shown in Figure 3.4 and Figure 3.5.

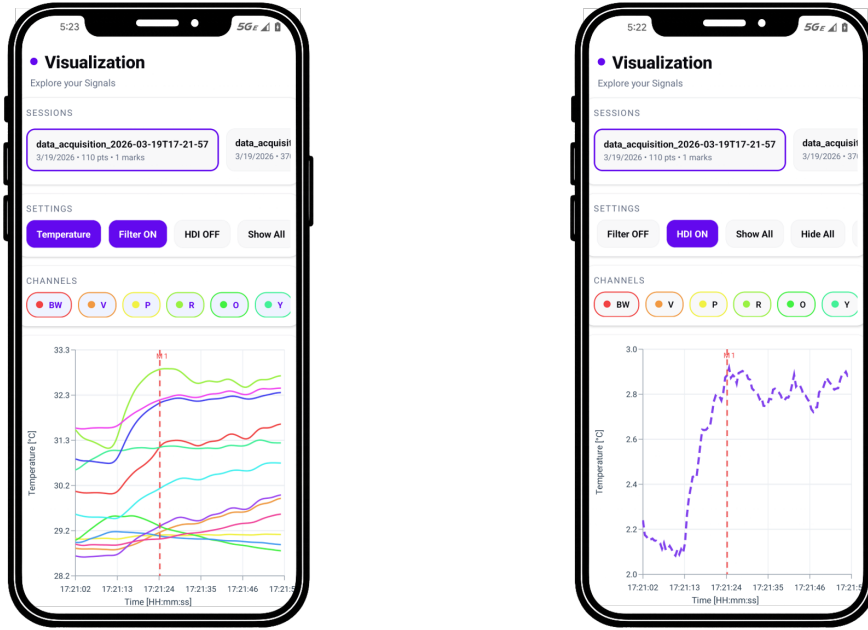


Figure 3.4: Offline visualization of recorded temperature sessions, including channel selection and signal inspection tools.

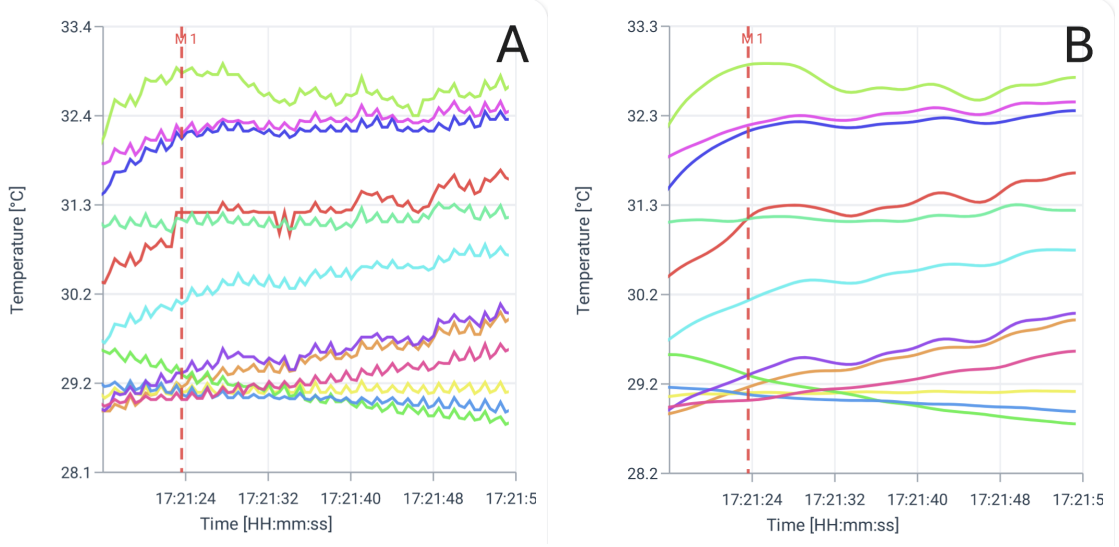


Figure 3.5: Example of recorded temperature data visualization before and after interactive signal processing.

3.1.2.3 Grid Mode

The Grid Mode was implemented to visualize the spatial evolution of knee temperature over time. The main objective of this mode was to provide a real-time thermal representation of the knee while maintaining a low computational cost on the mobile device.

The final visualization is obtained by combining two graphical layers. The first layer is a static knee map stored as a JSON file and included among the application resources. This file provides the geometric structure of the knee used by the interface and allows the anatomical reference shape to be rendered with minimal memory and computational requirements.

The second layer is a precomputed ROI map, also stored as a JSON file. This file contains a matrix of pixels belonging to the Region of Interest (ROI), corresponding to the area where the temperature sensors are positioned and where the thermal evaluation is performed. The spatial map was generated offline according to the sensor positions. First, a Voronoi-based association was used to relate each pixel to the closest sensor. Then, to obtain a smoother spatial representation, a local spatial window was applied around each pixel to estimate the contribution of the neighbouring sensors.

Since the sensor positions are fixed inside the knee sleeve, this spatial computation was performed only once offline. During application execution, the system only applies the precomputed weights to the current temperature values, avoiding the

need to recalculate the spatial map in real time. This design reduces the computational load and allows the heatmap to be updated directly on the mobile device. When the mode is activated, the application starts the acquisition and receives raw data through the BLE connection. The incoming packets are parsed using the same framing and decoding strategy adopted in the Real-Time Visualization Mode. After decoding, the ADC values from the 12 channels are converted into temperature values using the calibration lookup table.

For each pixel belonging to the ROI, the application computes a weighted combination of the 12 sensor temperatures. Each pixel is associated with 12 precomputed weights, one for each sensor channel. The temperature value assigned to a pixel is therefore calculated as:

$$T(\mathbf{p}, t) = \sum_{i=1}^{12} w_i(\mathbf{p}) T_i(t) \quad (3.3)$$

where $\mathbf{p} = (x, y)$ represents a pixel belonging to the ROI, $T_i(t)$ is the temperature measured by the i -th sensor at time t , and $w_i(\mathbf{p})$ is the precomputed weight representing the contribution of sensor i to pixel $\mathbf{p}$.

This operation generates an instantaneous heatmap of the knee region. A new heatmap frame is generated every 500 ms and displayed with a smooth fade transition, creating the perception of continuous real-time visualization while keeping the computational load limited.

The color mapping pipeline used to transform normalized temperature values into heatmap colors is summarized in Figure 3.6.

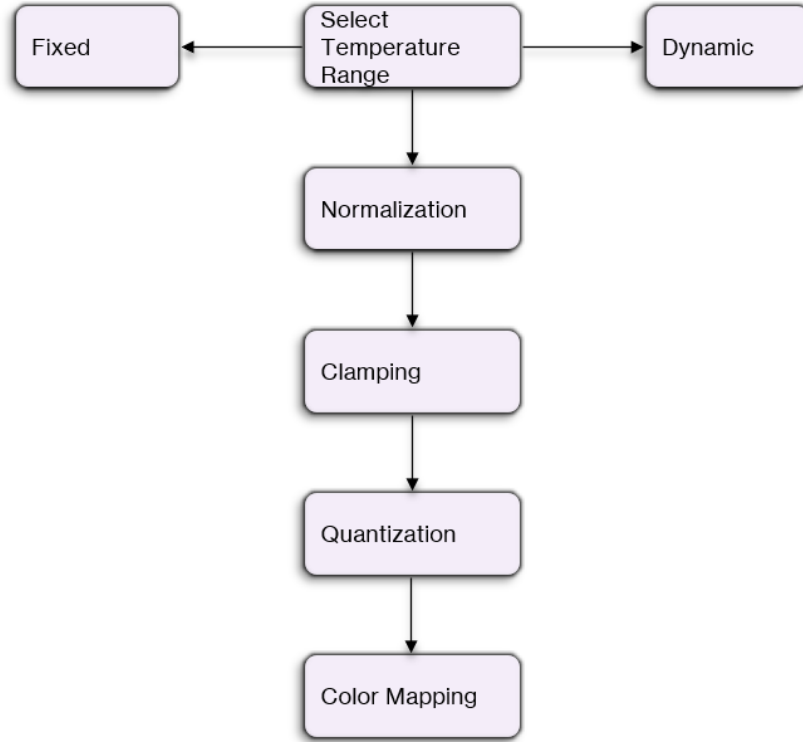


Figure 3.6: Color mapping pipeline used in the Grid Mode. The temperature range is selected either in fixed or dynamic mode, then the values are normalized, clamped, quantized, and finally mapped to the corresponding heatmap colors.

In addition to the heatmap, the Grid Mode also displays synthetic parameters that support the interpretation of the thermal distribution. The HDI value is computed using the same approach described in the previous mode and is shown together with a three-level graphical classification: Mild, Moderate, and Severe. This classification was included only as a visual indicator in the prototype and does not represent a clinically validated scale.

The application also displays the average temperature of the knee, computed over the active ROI. This value is represented by a moving indicator on a temperature bar,

whose limits correspond to the minimum and maximum values of the visualization scale. The same limits are used to define the color range of the heatmap.

The temperature scale can operate with either fixed or dynamic limits, according to the configuration selected by the user in the settings page. Once the minimum and maximum temperatures are defined, the pixel values are normalized between 0 and 1 and then clamped to avoid values outside the selected range. A quantization step is then applied to discretize the interval into a finite number of color levels. In the current implementation, the scale is divided into 20 levels, corresponding to 20 different color shades.

The final color is obtained through a piecewise gradient function. Lower values are represented with blue and cyan tones, intermediate values move toward green and yellow, and higher values gradually transition toward red. The generated heatmap is rendered with slight transparency, allowing the underlying knee map to remain visible while highlighting the spatial temperature distribution.

Examples of the final Grid Mode interface and spatial heatmap visualization are shown in Figure 3.7.

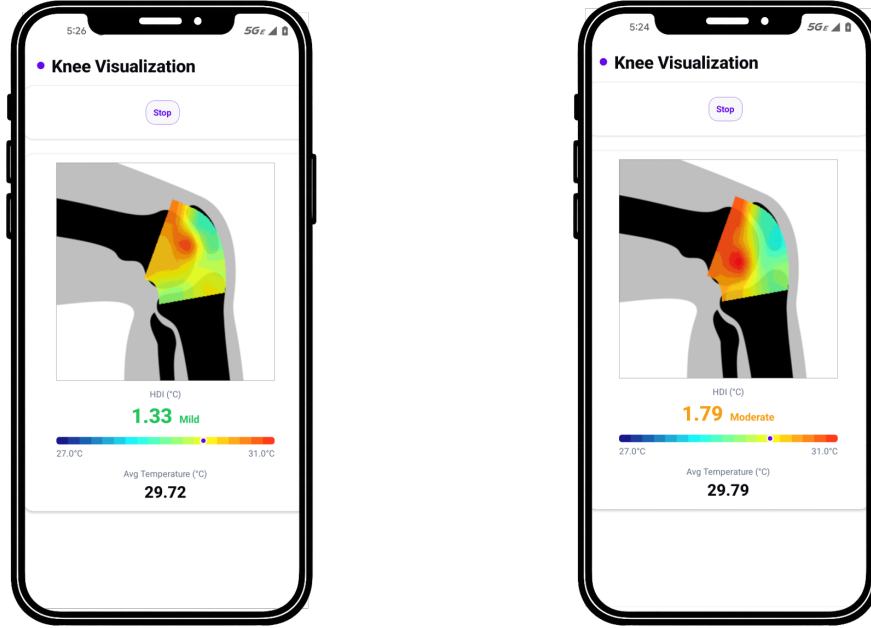


Figure 3.7: Grid Mode interface showing the spatial heatmap visualization and the additional temperature-related indicators.

3.1.3 Bioimpedance Module

The bioimpedance module was implemented to connect the mobile application to a knee sleeve able to of acquiring different types of data from multiple sensors embedded in the device.

The main functionality of this module is the acquisition of bioimpedance data, obtained by measuring the voltage response of the tissues after the injection of a low-amplitude current through the knee. Two acquisition modes were implemented for this purpose: a continuous mode and a sweep mode.

In addition to bioimpedance data, the module also receives kinematic information from two IMUs placed on the sleeve. In this configuration, the microcontroller processes the raw inertial signals on board and transmits the knee flexion–extension angle, which represents the main kinematic parameter visualized in the application. The collected data are displayed through dedicated plots, allowing the user to observe the evolution of resistance, reactance, and knee angle during acquisition or

during offline analysis.

An overview of the bioimpedance module and its main operating modes is shown in Figure 3.8.

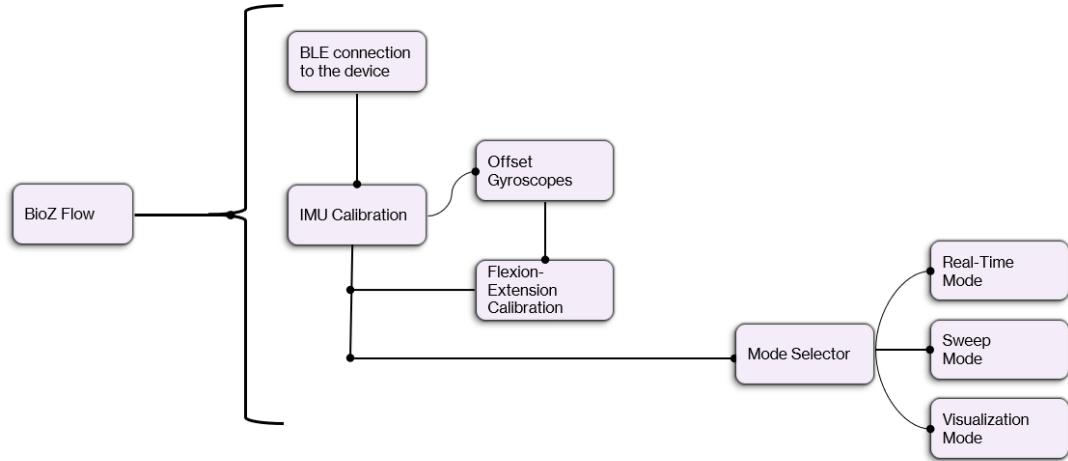


Figure 3.8: Overview of the bioimpedance module, including the main operating modes implemented for calibration, continuous acquisition, sweep acquisition, and data visualization.

3.1.3.1 Calibration Mode

When the user accesses the bioimpedance module, the first interface displayed is the calibration page. This page was implemented to ensure that the inertial measurements are initialized before accessing the operating modes of the sleeve. The calibration procedure is divided into two sequential phases, each lasting five seconds. In the first phase, the user can trigger the estimation and removal of the gyroscope offset of the IMUs. In the second phase, a dedicated button allows the calibration of the knee flexion–extension angle.

During the entire calibration procedure, the Bluetooth connection is automatically checked every three seconds. This verification was included to ensure that the connection with the device remains stable while the calibration process is being performed.

Once both calibration phases are completed, the application enables access to the mode selection page. From this page, the user can select one of the available operating modes: continuous mode, sweep mode, or visualization mode.

The implemented calibration workflow and the transition to the mode selection page are shown in Figure 3.9.

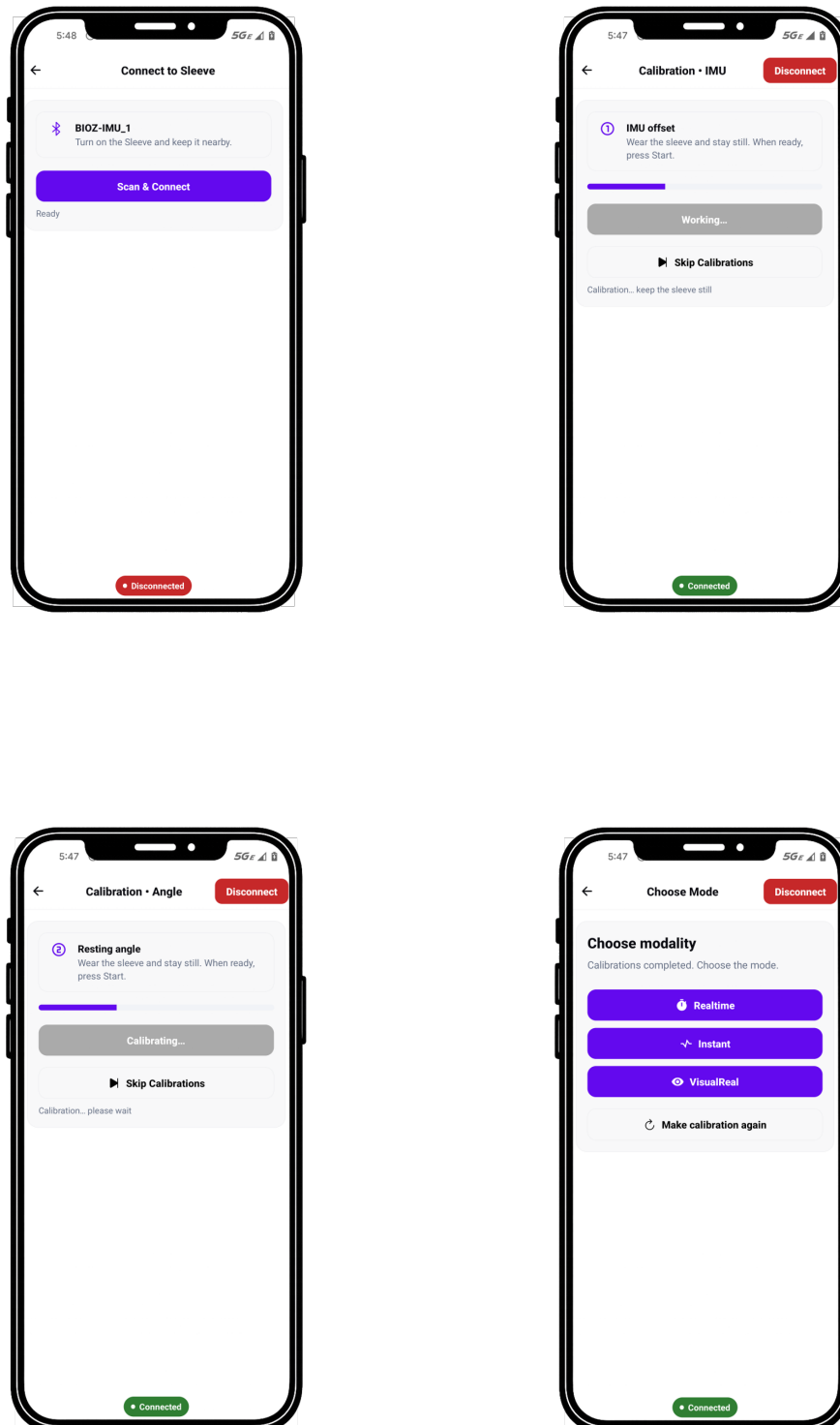


Figure 3.9: Calibration workflow of the bioimpedance module, including IMU offset calibration, flexion–extension angle calibration, and access to the mode selection page.

3.1.3.2 Real-Time Continuous Mode

The Real-Time Continuous Mode was implemented to acquire and visualize bioimpedance and kinematic data during continuous acquisition. In this mode, the acquisition is started by sending a command value equal to 1 to the service *FFF0* and to the control characteristic *FFF3*, while the incoming data are read from the streaming characteristic *FFF2*.

Once transmission is enabled, the application expects data samples with a fixed size of 16 bytes. Each sample is composed of four bytes for the flexion–extension angle, four bytes for resistance, four bytes for reactance, and four bytes for the timestamp.

The data are not transmitted as isolated samples, but as aggregated frames composed of 15 samples. Since each sample has a size of 16 bytes, each frame contains 240 bytes of useful payload. This value is close to the negotiated BLE MTU of 247 bytes and allows the communication channel to be used more efficiently.

The application asynchronously reassembles incoming fragments and decodes the data stream in order to avoid losses caused by packet fragmentation. The use of aggregated frames also reduces the communication overhead, since fewer BLE notifications are required to transmit the same number of samples.

After packet reception and parsing, the decoded data are visualized through two dedicated plots. The main characteristic of this mode is the separation between data ingestion and graphical rendering. The BLE parser extracts incoming packets in the background and stores the decoded values in memory buffers at the maximum available speed. In parallel, an asynchronous timer updates the SVG plots at fixed intervals, retrieving only the most recent portion of the buffers. This implementation prevents blocking of the user interface even when the incoming data traffic is high.

The same time axis is used to represent both the flexion–extension angle and the bioimpedance measurements. The impedance plot uses two vertical axes, one for resistance and one for reactance. The reactance is displayed with inverted sign because, in physiological conditions, this parameter is expected to assume negative values in human tissues. This graphical choice makes the visualization easier to interpret.

The time axis is reconstructed from the internal tick counter of the microcontroller, where one tick corresponds to ten microseconds. To reduce the computational load during visualization, a static downsampling strategy is applied to the displayed samples. The maximum number of points rendered in the plot is limited to 300 samples, keeping the CPU and GPU load approximately constant regardless of the duration of the displayed temporal window.

Once the acquisition is completed, the data can be saved. The storage mechanism is based on a chunked strategy: samples are temporarily accumulated in a memory

buffer and, when 1000 samples are reached, the block is written to non-volatile memory using AsyncStorage. This mechanism allows the RAM to be periodically freed during long acquisitions.

A fallback mechanism is also implemented. If new packets arrive while the previous block is still being written to memory, the storage operation is placed in a queue in order to prevent packet loss. At the end of the recording session, all stored blocks are reassembled by the application into a single dataset that can be exported.

The implemented real-time continuous acquisition interface is shown in Figure 3.10.

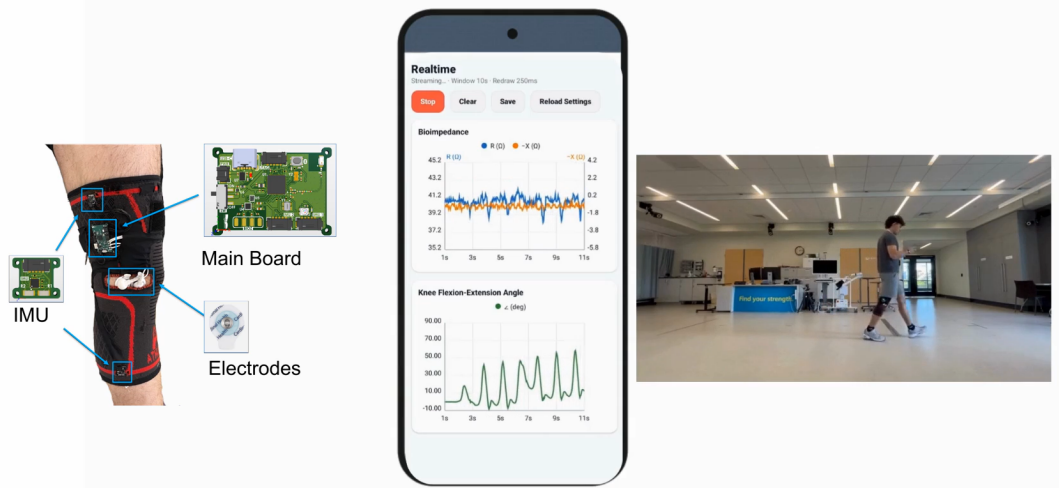


Figure 3.10: Real-time continuous mode of the bioimpedance module, showing the simultaneous visualization of knee flexion–extension angle, resistance, and reactance during acquisition.

3.1.3.3 Sweep Mode

The Sweep Mode was implemented to provide an on-demand bioimpedance acquisition across multiple excitation frequencies. In this mode, the acquisition is activated by sending the command value 2 to the service *FFF0* through the write characteristic *FFF3*, while the incoming data are read from the streaming characteristic *FFF2*.

During each sweep, the microcontroller transmits a total of seven packets. Each packet contains two *Float32* values: the first value represents resistance, while the second value represents reactance. The received packets are accumulated in an array and then displayed in a Cole–Cole plot.

The frequencies acquired in this modality are 5000 Hz, 10 kHz, 32 kHz, 50 kHz, 100 kHz, 200 kHz, and 300 kHz. This frequency sweep allows the system to obtain immediate information about the frequency-dependent electrical response of the tissue, supporting a broader characterization of bioimpedance behavior compared

with single-frequency acquisition.

A dedicated button was implemented to open a modal window showing the raw frequency of each packet together with the corresponding resistance and reactance values. The values are displayed in a scrollable table that can be closed by the user. The acquisition process is controlled by a watchdog timer of 1000 ms. If the microcontroller takes more than one second to transmit all the frequencies of the sweep, the application detects the delay and forces the device back into an idle state.

The SVG-based vector rendering engine implements an adaptive auto-scaling function to provide a stable visualization of the Cole–Cole arc. The system identifies the minimum and maximum values within the dataset, computes the geometric center of the points, and evaluates the signal excursion, referred to as the span. A proportional padding of the graphical area is then applied to preserve the morphology of the Cole–Cole arc and avoid visual distortions, especially when the difference between impedance values is large.

The implemented sweep acquisition interface and Cole–Cole visualization are shown in Figure 3.11.

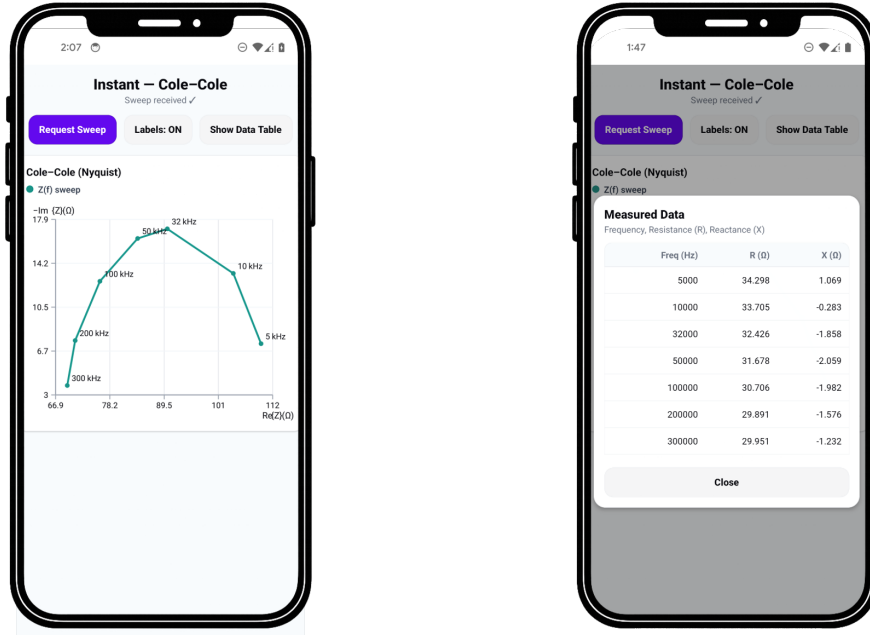


Figure 3.11: Sweep Mode interface of the bioimpedance module, showing the Cole-Cole plot and the table of resistance and reactance values acquired at different frequencies.

3.1.3.4 Offline Visualization and Data Export

An offline visualization mode was implemented for the bioimpedance module to allow the user to inspect the recordings generated during the Real-Time Continuous Mode.

From this interface, the user can scroll through the list of saved recordings and select a specific acquisition session. Once a recording is selected, the application loads the stored dataset and displays the evolution of three parameters: resistance, reactance, and knee flexion-extension angle. These signals are represented using two plots sharing the same time domain, allowing the user to compare bioimpedance variations with the corresponding knee movement.

The graphical visualization is supported by an adaptive sample management strategy. Each time the user zooms into the plot, the selected temporal window is evaluated and a rapid downsampling procedure is applied according to the

maximum number of points that can be displayed, set to 1000 samples. In this way, the displayed resolution adapts to the selected window: when the user zooms in, the effective resolution increases, making it possible to observe signal details that would otherwise remain hidden.

To make the recordings usable for external post-processing, this mode also provides an export function. The recorded datasets can be exported in both *.txt* and *.csv* formats. The export process is based on the serialization of the JSON files that contain the original session payload, allowing the acquired data to be analyzed with external tools. The offline visualization interface for saved bioimpedance recordings is shown in Figure 3.12.

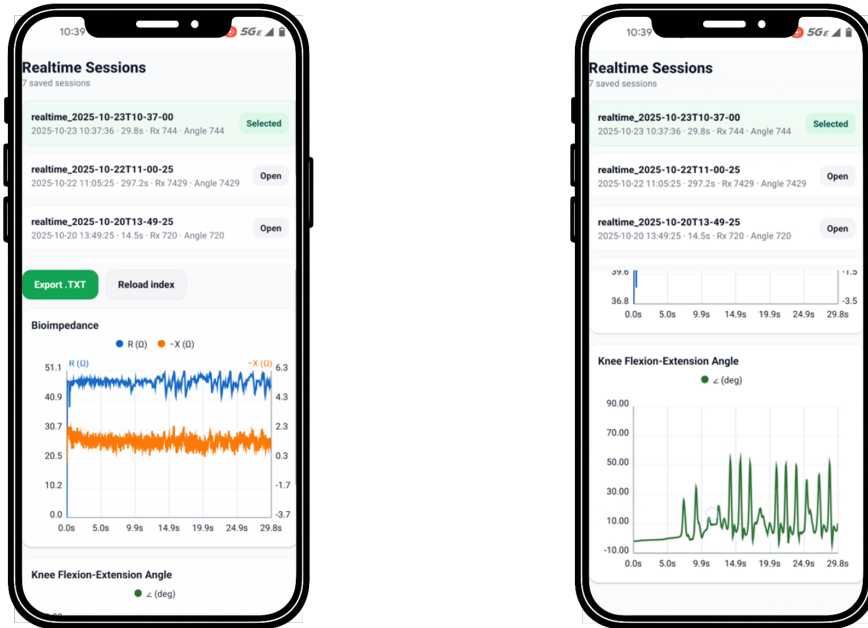


Figure 3.12: Offline visualization mode of the bioimpedance module. The interface allows the user to select saved sessions and visualize knee flexion–extension angle together with resistance and reactance over the same time domain.

3.1.4 Settings Module

The Settings Module represents one of the main results of the prototype, as it provides a centralized way to configure the behavior of the different application modes. Through this module, the user can define, modify, and adjust several parameters related to data acquisition, visualization, filtering, storage, and Bluetooth communication.

Since the system was designed as a research-oriented prototype, configurability was an important requirement. The Settings Module allows the user to adapt the application behavior to different devices, acquisition conditions, and visualization needs without modifying the source code.

The system is based on a persistent configuration strategy that relies on the internal storage of the mobile device through AsyncStorage. When a parameter is modified, either by pressing a button or by inserting a value through a text field, the value is first clamped to ensure that it remains within predefined limits.

The validated value is then passed to the `update()` function. This function performs two operations: one synchronous and one asynchronous. The synchronous operation immediately updates the user interface, making the change visible to the user. In parallel, the asynchronous operation triggers the `save()` function, which serializes the configuration object into a JSON string and stores it in AsyncStorage using the key `app_settings_v1`. This solution was adopted because JavaScript objects cannot be directly stored in the mobile storage system.

When any application page or operating mode is opened, the first operation performed by the application is to read the AsyncStorage content. The updated JSON configuration file is loaded, and the corresponding variables are overwritten with the stored parameter values. This ensures that the selected configuration remains consistent across the different modules of the application.

The settings available in the final version of the prototype are summarized below:

BLE Settings Connectivity and hardware parameters:

- Auto-scan (`autoScanOnOpen`)
- Scan timeout (`scanTimeoutMs`)
- Auto-reconnect (`autoReconnect`)
- MTU request (`requestMtu`)

Acquisition Settings Standard and non-real-time acquisition parameters:

- Window width (`windowSec`)
- Soft redraw interval (`redrawSoftMs`)
- Hard redraw interval (`redrawHardMs`)

Realtime Settings High-frequency bioimpedance acquisition parameters:

- Realtime window (`windowSec`)

Chart Settings Visualization and filtering parameters:

- Default unit (`yMode`)
- Temperature range mode (`tempRange.mode`)
- Minimum temperature range (`tempRange.minC`)
- Maximum temperature range (`tempRange.maxC`)
- Show legend (`showLegend`)
- Enable filtering (`showFiltered`)
- Low-pass filter cutoff frequency (`filterFc`)
- Low-pass filter order (`filterOrder`)
- Show HDI (`showHDI`)

Storage Settings Data persistence parameters:

- Auto-save on stop (`autosaveOnStop`)
- Maximum number of stored files (`maxSaved`)

The implemented settings interfaces are shown in Figure 3.13.

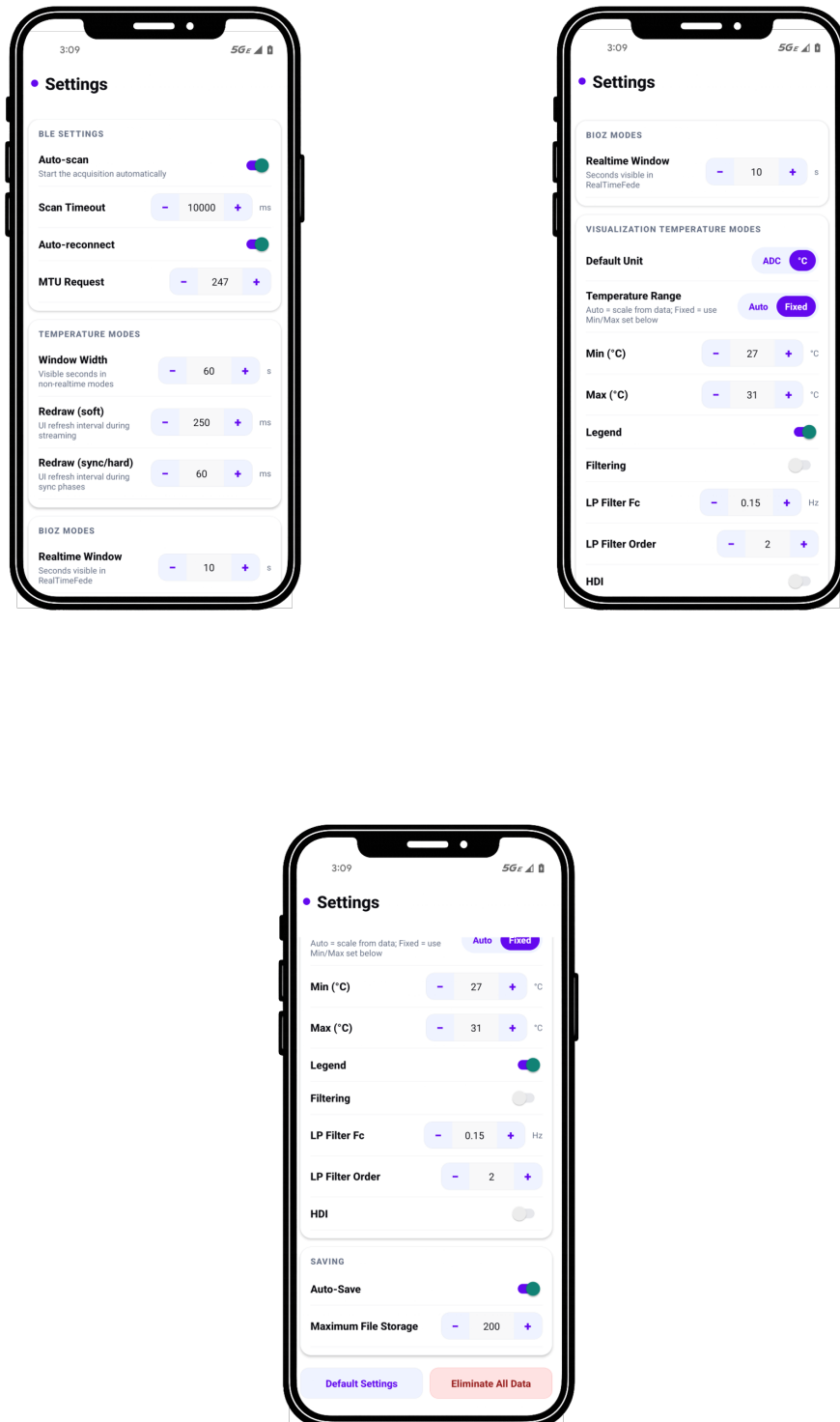


Figure 3.13: Settings Module interfaces showing the configurable parameters for BLE communication, acquisition, visualization, filtering, and storage.

3.2 Phase 2 – Multimodal Developer Mode

In the second phase of the project, a new section of the application, referred to as *Developer Mode*, was implemented. This mode was designed to provide a prototype environment for testing the future integration of multiple sensing modalities within a single multisensorized knee sleeve.

Since the final integrated device was not yet available, the development followed a two-step strategy. First, the multimodal data structure, parsing logic, synchronization procedures, and signal processing algorithms were developed and tested offline in Python using a constructed dummy dataset. Then, the same logic was translated and integrated into the React Native mobile application built in the Phase 1 of the project.

The goal of this phase was therefore not to validate a final hardware system, but to prepare a functional software framework capable of receiving, decoding, processing, and visualizing multimodal data expected from a future version of the knee sleeve. The long-term objective is to progressively integrate the sensing solutions previously implemented in separate sleeves into a single device capable of handling multisignal acquisition and processing. The initial workflow of the Developer Mode, including Bluetooth scanning, acquisition management, and analysis initialization, is shown in Figure 3.14.

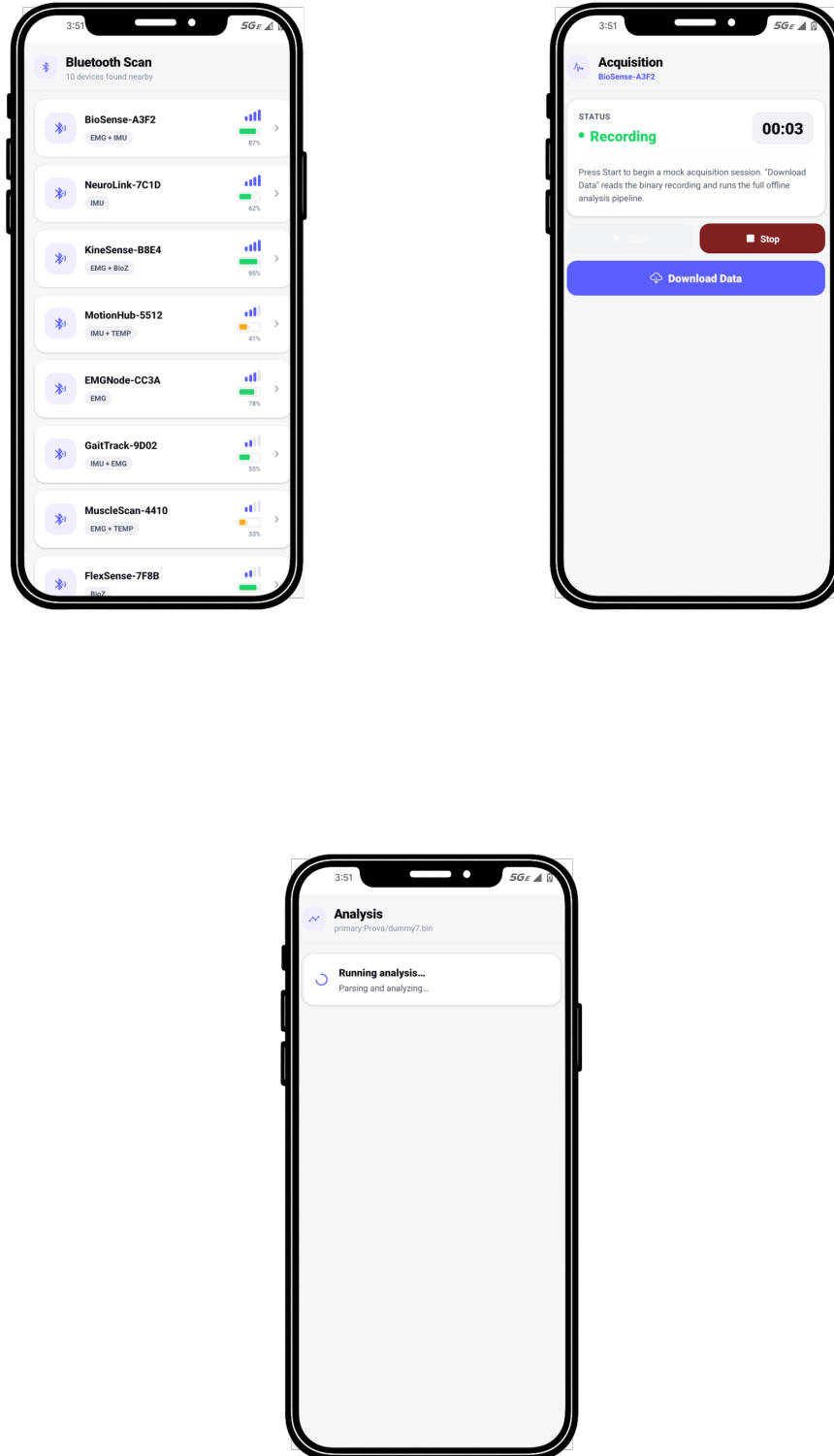


Figure 3.14: Initial workflow of the Developer Mode. The interface includes Bluetooth device scanning, acquisition management, and initialization of the multimodal analysis pipeline.

3.2.1 Dummy Multimodal Dataset Construction

Since the final multisensorized knee sleeve was not yet available, a dummy multimodal dataset was constructed to test the signal processing and visualization pipeline. The dataset was first generated and processed offline in Python, and was then used as a reference to implement the same logic inside the React Native Developer Mode.

The constructed dataset had a duration of approximately 10 seconds and was designed to simulate a short walking sequence composed of multiple gait cycles. It was generated by combining acquisitions from different sources, each providing a specific sensing modality.

IMU data from the thigh and shank, together with synchronized EMG data from the vastus medialis, were obtained from the external open-source dataset described in the Materials and Methods chapter. Bioimpedance data were instead collected using the Bioimpedance Knee Sleeve during two separate ground-level running sessions.

In the first bioimpedance session, the voltage response was recorded using an excitation frequency of 5 kHz, while in the second session an excitation frequency of 100 kHz was used. Since these two recordings were acquired in separate sessions, a synchronization procedure was required before combining them.

The synchronization was performed using the flexion–extension angle streamed by the microcontroller of the Bioimpedance Knee Sleeve. This signal was available together with the bioimpedance data in both recordings and was therefore used as a reference for temporal alignment. First, the two flexion–extension angle signals were normalized by subtracting the mean and dividing by the standard deviation. Then, cross-correlation was applied by sliding the 100 kHz signal over the 5 kHz signal, and the lag corresponding to the maximum correlation was identified.

Once the lag was computed, the two bioimpedance recordings were temporally aligned. A linear interpolation step was then applied to ensure that the samples acquired at the two excitation frequencies referred to the same time instants.

After synchronization, a mixing phase was performed to emulate the expected behavior of the future integrated device. In this simulated transmission logic, five samples acquired at one frequency are followed by five samples acquired at the other frequency, alternating between the two frequencies over time.

Temperature data were obtained from the Temperature Knee Sleeve during a running session. After all types of data were collected and synchronized, they were combined to build a multimodal packet stream structured according to the expected packet architecture of the future multisensorized device.

3.2.2 Implemented Packet Architecture and Parsing

Packet Structure

A configurable packet architecture was implemented to emulate the data stream expected from the future multisensorized knee sleeve. Each packet is encapsulated in a binary frame with a fixed structure, allowing the receiver to identify, validate, and decode heterogeneous sensor data within the same stream.

The first 4 bytes represent the sync word, defined as the hexadecimal constant 0xAABBCCDD. This field is used by the receiver to detect the beginning of a new packet within the continuous binary data stream.

The sync word is followed by a 1-byte Packet ID, which specifies the type of data contained in the payload. A 2-byte sequence number is then included as an incremental counter to support the detection of potential packet loss during transmission. The following 4 bytes contain the timestamp, representing the acquisition time expressed in clock ticks, based on a reference frequency of 32.768 kHz.

After this common header, the payload contains the actual sensor data. Its size depends on the specific type of packet being transmitted. Before transmission, physical values are converted into digital format through a 12-bit quantization process. The floating-point values are first scaled according to the sensor full scale (FS) and then encoded into signed integers using two's-complement representation. The quantized samples are packed using little-endian byte order. Finally, each packet ends with a 4-byte CRC32, computed over the packet content, including header and payload. This field allows the parser to detect corrupted or incomplete packets before data conversion.

The general structure of the implemented packet architecture is shown in Figure 3.15.

4 BYTES	1 BYTE	2 BYTES	4 BYTES	N BYTES	4 BYTES
SYNC	ID	SEQ	TO	PAYLOAD	CRC32

Figure 3.15: General structure of the binary packet implemented for the multimodal Developer Mode. The packet includes a sync word, packet identifier, sequence number, timestamp, payload, and CRC32 integrity field.

The supported packet types differ in terms of payload content, payload size, and transmission frequency. The payload structure is summarized in Table 3.2, while the expected transmission frequency of each packet type is reported in Table 3.3.

Table 3.2: Payload structure for the supported packet types.

Payload type	Content	Payload bytes	Total bytes
EMG	1× EMG	2	17
IMU	12× Accel/Gyro	24	39
TEMP	12× ADC1/ADC2	24	39
Z5	2× R/X	4	19
Z100	2× R/X	4	19

Table 3.3: Transmission frequency defined for each supported payload type.

Payload type	Frequency [Hz]
EMG	1000
IMU	50
TEMP	1
Z5	50
Z100	50

Parsing of Data

The parsing procedure was first developed and tested offline in Python using the previously generated dummy dataset. Once the pipeline was tested locally, the same logic was translated into React Native and integrated into the mobile application. This two-step strategy made it possible to verify the decoding process in a controlled offline environment before implementing it in the Developer Mode of the application.

The main objective of the parsing stage is to convert the raw binary stream into structured physical data. This operation is implemented in the `parse_bin` function, which relies on a dynamic configuration defined in a JSON file. This configuration describes the packet structure, including header fields, payload format, and data types, making the parser flexible and adaptable to different packet definitions.

The parsing process is performed through the following sequential steps:

1. Reading of the global parameters from the settings file.
2. Sync word search.
3. Packet identification.
4. Length evaluation.
5. Integrity check using CRC32.
6. Extraction of header and payload data.
7. Conversion into physical units through dequantization.

The parser operates as a state machine. Since packets may have different lengths, the first step consists of searching for the sync sequence. The incoming stream is scanned one byte at a time until the start sequence is detected. Once the sync word is found, the next byte, corresponding to the Packet ID, is read in order to determine the total packet length and the number of additional bytes that must be acquired.

After all packet components have been identified, the packet is checked for corruption before processing the data. The header and payload are used to compute a CRC32 checksum, which is then compared with the last 4 bytes of the packet. If the checksum matches, the packet is accepted; otherwise, it is discarded.

Once the packet is validated, the raw payload is converted back into physical values. The incoming data, received as 16-bit integers, first undergo a bitmask operation to ensure correct sign reconstruction. The resulting value, expressed in ADC units, is then converted into its physical unit according to:

$$V_{out} = \left(\frac{S}{Q_{max}} \right) \cdot FS \quad (3.4)$$

where S denotes the signed two's-complement value, $Q_{max} = 2047$ is the maximum quantization level for the adopted 12-bit representation, and FS is the full-scale value associated with the sensor.

After parsing, a linear synchronization step is applied. Since each data stream may start at slightly different times, the minimum absolute timestamp among all the packets is identified and subtracted from every timestamp. In this way, all signals are temporally aligned and the reconstructed plots start from a common reference time, namely $t = 0$.

The temporal order of the packets within the simulated multimodal stream is illustrated in Figure 3.16. The figure shows the arrival sequence of the different packet types according to their transmission frequencies, highlighting the interleaved structure of the multimodal data stream.

EMG	IMU	TEMP	BIOZ	EMG	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	EMG	EMG	IMU	BIOZ	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	EMG	EMG	EMG	EMG	EMG
EMG	IMU	BIOZ	EMG	EMG	EMG
EMG	EMG	EMG	EMG	EMG	...

Figure 3.16: Example of packet arrival sequence in the simulated multimodal stream. Each cell represents an incoming packet, ordered according to the transmission frequency of the corresponding sensing modality (e.g., IMU, EMG, TEMP, Bioimpedance 5kHz (BIOZ), and Bioimpedance 5kHz (BIOZ)).

3.2.3 Multimodal Signal Processing

After parsing and synchronization, each signal was processed according to its sensing modality. The processing logic was first developed and tested offline in Python on the constructed dummy dataset and was then translated into React Native for integration into the Developer Mode. In this way, the same processing strategy tested in the offline environment was reproduced inside the mobile application.

Temperature

For the temperature channels, the dequantized value does not directly correspond to temperature in degrees Celsius, but to an ADC-related value following a non-linear relationship. For this reason, the logarithmic calibration model already used in Phase 1 was applied using the corresponding calibration coefficients. This step converts the dequantized measurements into temperature values expressed in degrees Celsius.

IMUs

For the inertial signals, the first processing step consists of gyroscope bias calibration. To estimate this bias, the first 0.8 seconds of the recording are considered, assuming that during this initial interval the sensor remains still.

For each gyroscope axis, both for the shank and thigh units, the mean angular velocity measured during this interval is computed. This mean value represents the constant offset affecting the sensor and is subtracted from all subsequent samples of the corresponding axis, obtaining a corrected angular velocity signal.

This correction is important because even small gyroscope offsets can accumulate over time when the signal is integrated, leading to drift in the estimation of the angle.

After bias removal, a linear interpolation is applied to obtain a uniform sampling frequency of 50 Hz, which is required by the following processing steps.

Since the axis that best captures the walking movement is not known in advance, an automatic axis selection procedure was implemented. For each axis, the median is first removed as an estimate of the residual bias. Then, a zero-phase exponential moving average filter with an effective cutoff frequency of 8 Hz is applied to reduce noise and rapid oscillations without introducing temporal shifts.

After filtering, the standard deviation of each axis is computed, and the axis with the highest standard deviation is selected as the most informative component for the subsequent gait analysis.

EMG

The EMG processing pipeline begins with the removal of the DC component by subtracting the median of the signal. The centered signal is then filtered using a fourth-order Butterworth band-pass filter with cutoff frequencies of 20 Hz and 450 Hz. The filtering is applied in zero-phase mode using a forward-backward approach, in order to avoid temporal distortion.

After band-pass filtering, the signal is full-wave rectified. Finally, a fourth-order Butterworth low-pass filter with a cutoff frequency of 6 Hz is applied, also in zero-phase mode, to extract the EMG envelope.

Bioimpedance

Bioimpedance data were represented at two excitation frequencies, namely 5 kHz and 100 kHz. Each measurement includes a real component, corresponding to resistance, and an imaginary component, corresponding to reactance. The data are collected in bursts of five samples for each frequency. To reduce quantization noise and rapid fluctuations, each burst of five samples is reduced to a single representative value by computing the arithmetic mean.

3.2.4 Extracted Indicators and Reporting

After the multimodal signals were parsed, synchronized, and processed, several indicators were extracted in order to generate a detailed visual report for the user. These indicators were designed to summarize different aspects of the acquisition, including gait timing, muscle activation, spatial temperature variability, and frequency-dependent bioimpedance variations.

As for the previous processing steps, the feature extraction logic was first developed and tested offline in Python and was then translated into React Native for integration into the Developer Mode.

Gait Segmentation

The first indicator extracted from the multimodal pipeline was gait segmentation, aimed at identifying the main phases of the gait cycle from the processed gyroscope signal. To obtain this information, the selected gyroscope axis was transformed into a state-based representation by applying a hysteresis rule with a symmetric threshold of ± 15 deg/s. In this way, each sample was associated with a discrete state value, as summarized in Table 3.4.

After the state representation was obtained, peaks of the gyroscope signal were detected to identify the mid-swing phase. An adaptive threshold was implemented for peak detection. First, the Median Absolute Deviation (MAD) of the signal was computed. Then, the deviations of the signal from the median were calculated, and their median value was multiplied by a scaling factor to obtain the final threshold used for peak detection.

Table 3.4: Hysteresis-based state determination with symmetric threshold $h = 15$ deg/s.

Condition on $g[n]$	State $s[n]$
$g[n] > +h$	+1
$g[n] < -h$	-1
$-h \leq g[n] \leq +h$	0 (neutral zone)

Each detected peak was interpreted as a mid-swing event. Around each peak, two temporal windows of 0.3 seconds were selected, one before and one after the peak. Since the signal was resampled at 50 Hz, each window corresponded to 15 samples. Within the window preceding the peak, the Toe-Off event was detected as the first transition of the state from -1 to $+1$. Within the window following the peak, the Heel Strike event was detected as the first transition of the state from $+1$ to -1 . The strides were then defined as the intervals between consecutive Heel Strike events. Only strides with a duration between 0.35 seconds and 1.8 seconds were accepted, corresponding approximately to step frequencies between 0.55 Hz and 2.86 Hz.

Once the gait events were identified, the stance percentage was computed as:

$$\text{Stance}_{\%} = 100 \cdot \frac{t_{TO} - t_{HS}}{t_{HS}^{next} - t_{HS}} \quad (3.5)$$

where t_{HS} is the time of the Heel Strike event, t_{TO} is the time of the Toe-Off event, and t_{HS}^{next} is the time of the following Heel Strike.

EMG Activation

Another indicator extracted from the multimodal pipeline was the percentage of EMG activation during the gait cycle. After computing the EMG envelope as described in the previous section, an adaptive threshold was estimated to detect muscle activation within each stride.

The baseline was defined as the median of the lowest 10% of the envelope values within each stride. Then, the Median Absolute Deviation (MAD) was computed as the median of the absolute deviations of the envelope from its median value. The final activation threshold was defined as the baseline plus the MAD multiplied by a scaling factor.

Once the threshold was defined, the algorithm counted the number of samples above the threshold. This allowed the activation percentage to be computed for the entire stride and separately for the stance and swing phases. The stance phase was defined as the interval from Heel Strike to the subsequent Toe-Off, while the swing phase was defined as the interval from Toe-Off to the following Heel Strike.

HDI Index

The HDI index was computed from the calibrated temperature channels as a measure of the spatial dispersion of temperature over the knee surface. For each time sample, the standard deviation across the twelve temperature sensors was calculated and multiplied by two. Since temperature data were available at 1 Hz, this procedure produced one HDI value every second.

The index was computed as:

$$HDI(t) = 2 \sigma_{\text{channels}}(T_1(t), T_2(t), \dots, T_N(t)) \quad (3.6)$$

where $T_i(t)$ represents the temperature measured by the i -th sensor at time t , and N is the number of active temperature channels.

H_α Index

After collapsing the bioimpedance bursts at the two excitation frequencies, the H_α index was computed. This index is a dimensionless ratio that combines the variations in bioimpedance measured at 5 kHz and 100 kHz.

The rationale behind this index is that the electrical response of biological tissues depends on frequency. According to the Cole–Cole model, tissues with different water content and different intra- and extracellular fluid distribution may respond differently to electrical stimulation at different frequencies. For this reason, for each stride, the difference between the maximum and minimum resistance value was first calculated separately for each frequency. Then, the ratio between the variation measured at 100 kHz and the variation measured at 5 kHz was computed. Values of H_α greater than 1 indicate that the resistance variation at the higher frequency is larger than the variation at the lower frequency. In this prototype, the index was used as an exploratory indicator of frequency-dependent tissue response rather than as a clinically validated marker. The index was calculated for each stride and plotted over time to monitor its evolution during walking.

$$\Delta R_f(k) = \max_{t \in [HS_k, HS_{next_k}]} R_f(t) - \min_{t \in [HS_k, HS_{next_k}]} R_f(t) \quad (3.7)$$

$$H_\alpha(k) = \frac{\Delta R_{100 \text{ kHz}}(k)}{\Delta R_{5 \text{ kHz}}(k)} \quad (3.8)$$

3.2.5 Developer Mode Implementation

In order to provide a unified environment for advanced multimodal signal analysis, all the implemented processing algorithms and extracted metrics were integrated into a dedicated application modality referred to as *Developer Mode*.

This modality was designed as an advanced interface that allows the user to access

the processing procedures previously developed and tested offline in Python, and later translated into React Native for integration within the mobile application. In this way, the application reproduces inside the mobile environment the same signal analysis logic that had already been verified in a controlled offline setting.

All the components previously described, including gait segmentation, EMG activation analysis, temperature-derived indices such as HDI, bioimpedance-derived indices such as H_α , and raw multimodal data visualization, are combined into a single synchronized framework. This integration enables the simultaneous observation of multiple physiological components sharing the same temporal axis.

From a graphical perspective, the interface is organized into multiple sections, each dedicated to a specific signal or metric. The layout allows the user to visualize processed signals in a structured way and to compare the behavior of different sensing modalities during the same simulated acquisition. In addition, the system provides the possibility to export raw data, enabling further offline analysis with different processing configurations.

Therefore, Developer Mode represents the practical implementation of the multimodal processing pipeline, integrating previously separate analytical structures into a unified and interactive analysis tool.

The implemented Developer Mode interfaces, including the visualization of the processed signals and extracted indicators, are shown in Figures 3.17 and 3.18.

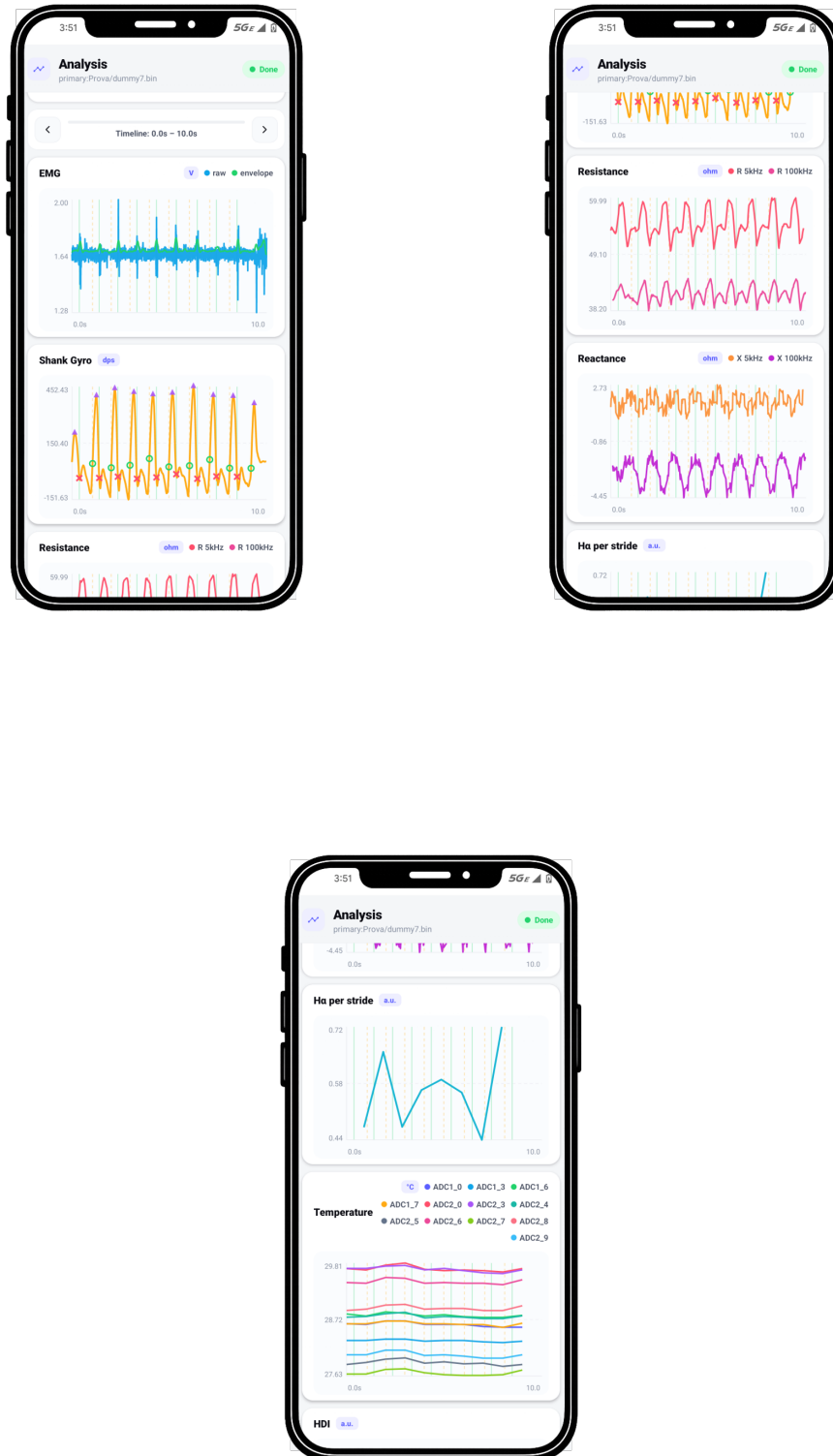


Figure 3.17: Developer Mode interfaces showing the visualization of multimodal signals and extracted indicators, including gait-related parameters, EMG activation, and synchronized signal trends.

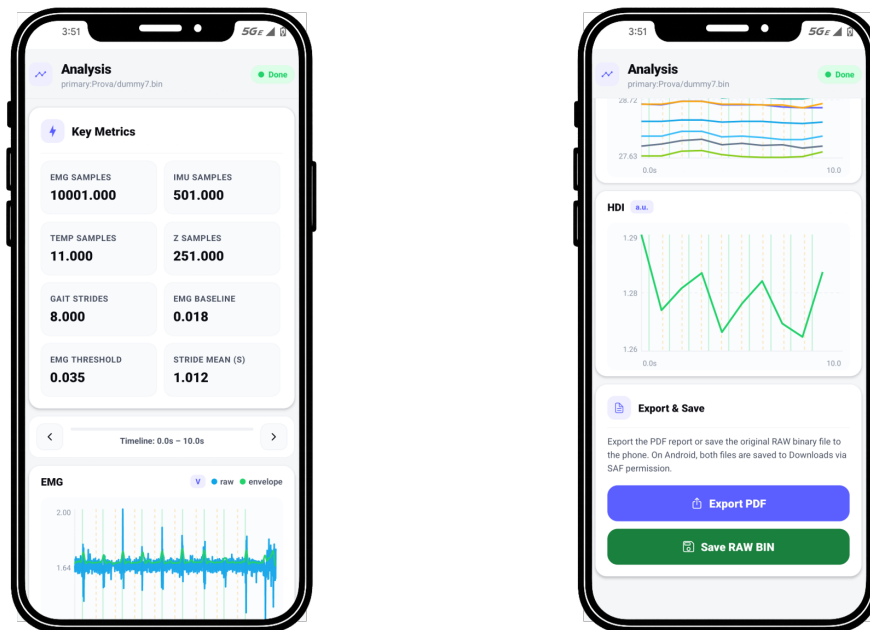


Figure 3.18: Additional Developer Mode interfaces related to multimodal analysis and data export, showing the integration of the different processing outputs within the mobile application.

4 Discussion

4.1 Interpretation of Results

4.1.1 Feasibility of a Multimodal Monitoring System

BLE Communication and MTU Optimization

One of the main challenges during development was the integration of Bluetooth Low Energy communication across different wearable devices and microcontrollers. This choice proved to be suitable for the prototype, as the communication logic was designed to exploit an increased BLE MTU and compact binary packet structures. In the temperature module, fixed-size frames were used to transmit one temporal index and twelve sensor channels. In the bioimpedance module, aggregated blocks were adopted to transmit multiple samples within the same BLE notification whenever the negotiated MTU allowed it. This organization reduced protocol overhead and improved transmission efficiency, supporting continuous acquisition and real-time visualization within the mobile application. The use of BLE also represented an advantage from the hardware point of view, since it enabled low-power wireless communication while maintaining adequate throughput for the tested acquisition scenarios. As a result, the same application hub was able to connect to different wearable systems, such as the Bioimpedance Knee Sleeve and the Temperature Knee Sleeve, while remaining compatible with low-power wearable operation.

Connection Stability

The stability of the BLE communication was supported by a persistent connection management strategy. The application continuously verified the connection state, started the data stream only when all the required BLE resources were available, and safely removed active subscriptions when the device was disconnected.

Real-Time Data Acquisition

Another difficulty that emerged during the design phase was the management of real-time data acquisition. The application had to handle two tasks at the same time: on one side, incoming data had to be received and displayed through real-time plots; on the other side, the same data had to be buffered in memory to preserve

the acquisition and allow the generation of a detailed report for the user. To address this issue, a bounded in-memory buffering strategy was adopted. The acquired samples were continuously stored in dedicated data structures, while the graphical interface was updated at controlled time intervals instead of being refreshed for every received packet. Thanks to this approach it was possible to reduce the computational load and avoid overloading the user interface. Even though a true multithreaded implementation was not explicitly used, the separation between data ingestion, buffering, and rendering proved suitable for the prototype. In particular, this solution allowed the system to maintain a smooth visual stream while preserving the acquired data for subsequent storage and analysis.

Packet Parsing

Particular attention was devoted to packet parsing, since the application had to manage different packet structures depending on the sensing modality and acquisition mode. The implemented logic was able to decode both explicitly delimited frames and aggregated fixed-length payloads. In the second phase of the project, the offline parser extended this approach by introducing sync-word detection, packet identification, endian-aware decoding, CRC32 integrity verification, and conversion into physical units.

This made the acquisition pipeline more robust to heterogeneous packet structures and possible transmission irregularities, supporting both the single-modality workflows and the simulated multimodal data stream used in Developer Mode.

Multi-Level Buffering Strategy

Finally, the buffering process was implemented at different levels of the acquisition pipeline. Raw bytes were first accumulated until complete packets could be reconstructed, while decoded samples were stored in bounded in-memory structures for real-time plotting and subsequent saving.

In the real-time bioimpedance mode, an additional chunk-based storage strategy was introduced to periodically offload data to local storage during long acquisitions. This choice contributed to maintaining a more stable acquisition process and a continuous visual stream without overloading the user interface.

4.1.2 Robustness of the Signal Processing Pipeline

The robustness of the signal processing pipeline represents one of the key aspects of the application development. Signal analysis was addressed through different strategies to reduce the influence of noise, motion artifacts, and variability between acquisitions. The pipeline was therefore designed to support the feature extraction and real-time analysis under the tested conditions.

Signal Stabilization

Before extracting information, the signals were processed in order to reduce variability and improve readability. Low-pass filtering applied to temperature signals reduced high-frequency noise, while IMU smoothing helped attenuate rapid oscillations and spurious peaks. In the case of EMG, the computation of the envelope transformed a highly variable signal into a more interpretable representation of muscle activation.

Signal transformations such as rectification and envelope extraction were also important because they did not only reduce noise, but also transformed the raw signals into quantities that were more suitable for interpretation and feature extraction.

Feature Design to improve Robustness

A key contribution to robustness comes from the design of the extracted features, which were intentionally made less dependent on absolute signal values. Bioimpedance features during gait were computed as the difference between maximum and minimum values within each stride. This strategy reduces the dependence on absolute impedance values and may increase robustness against offset drift and calibration errors.

The Heat Distribution Index (HDI) was designed to measure spatial dispersion rather than absolute temperature. Since it is computed across multiple temperature channels, it provides an indication of thermal variability that is less dependent on the value of a single sensor.

The H_α index, defined as a ratio between bioimpedance variations at two frequencies, was introduced to reduce scaling effects and highlight frequency-dependent bioimpedance behavior. Similarly, EMG activation was computed as the proportion of time in which the signal exceeded an adaptive activation threshold, resulting in a time-normalized measure rather than an absolute amplitude-based value.

Temporal Segmentation and Consistency

In the offline analysis, temporal normalization was obtained by segmenting the signals into strides, which represent repeated physiological units of gait. This allowed the extracted indicators to be compared across different gait cycles.

Segmentation also reduced the variability that would otherwise be present when analyzing continuous signals over the entire recording. As a result, EMG, bioimpedance, and temperature-related indicators could be evaluated with respect to the same gait events.

So basically, segmentation acted as a key mechanism to transform continuous data streams into more interpretable physiological units, making the extracted features more comparable across cycles.

Multimodal Complementarity

Robustness does not derive only from signal processing, but also from the complementarity of the different sensing modalities. After segmentation, each gait cycle can be described using temporal information from the IMUs, muscle activation information from EMG, frequency-dependent tissue response from bioimpedance, and thermal distribution from temperature measurements.

This integration provides a comprehensive overview of the monitored condition and supports a more meaningful interpretation of the acquisition than a single sensing modality alone.

Robustness of Real-Time and Offline Pipelines

The robustness of the pipeline is expressed differently in real-time and offline modalities, reflecting two complementary objectives. The real-time pipeline was designed to be lightweight and computationally efficient in order to support continuity of data streaming and stable visualization on a mobile device. In this case, processing operations were intentionally simplified, prioritizing stability and responsiveness over computationally expensive analysis.

On the other hand, the offline pipeline takes advantage of access to the complete recorded signal to apply more advanced processing techniques, including zero-phase filtering, temporal reconstruction, and synchronization across modalities.

As a result, the system combines two levels of robustness: operational continuity during real-time acquisition and improved analytical consistency during offline analysis.

4.1.3 Physiological Meaning of the Extracted Indicators

The extracted indicators can provide useful information related to different aspects of human movement and local tissue behavior. However, their interpretation must be considered exploratory, since the system was developed and tested as a prototype and was not clinically validated.

EMG signals allow the characterization of muscle activation during gait, providing information not only about the presence of activation but also about its temporal distribution within the gait cycle. In particular, the analysis of activation duration can support the evaluation of muscle involvement across different gait phases, especially for muscles such as the vastus medialis, which plays an important role in knee extension and stabilization.

Bioimpedance measurements, together with the H_α index, provide information about the frequency-dependent electrical response of the tissues during movement. Since tissue impedance is influenced by several factors, including fluid distribution and tissue composition, these signals may support the observation of dynamic changes in local tissue behavior. Nevertheless, in this work they should be interpreted as exploratory indicators rather than direct markers of specific physiological mechanisms.

Temperature measurements, combined with the Heat Distribution Index (HDI), provide insight into the spatial distribution and variability of thermal patterns around the knee. This can help identify localized variations or asymmetries in temperature distribution. However, possible associations with tissue condition or inflammatory processes require further experimental and clinical validation.

The integration of these indicators supports a multimodal characterization of the monitored condition, combining information related to muscle activation, tissue electrical response, and thermal distribution within a unified framework.

4.2 Limitations and Critical Considerations

4.2.1 Use of Simulated Multimodal Data

For the development of the multimodal Developer Mode, dummy data were introduced because the final multisensorized knee sleeve was not yet available. These data were obtained by combining signals acquired from different devices, with the aim of emulating the packet stream expected from the future integrated system.

The constructed dataset represented a useful tool for experimentation and software development. It allowed the signal processing and visualization pipeline to be tested offline in Python before being translated into React Native and integrated into the mobile application. However, this dataset does not fully reproduce real acquisition conditions.

In particular, bioimpedance, IMU, EMG, and temperature signals were acquired in different sessions and under different conditions. As a consequence, walking speed and movement execution were not perfectly consistent across recordings, and synchronization had to be performed a posteriori. This may introduce temporal misalignments and potential errors in the analysis. Considering this scenario, the gait segmentation can partially reduce the effect of these discrepancies by organizing the data into repeated gait cycles, but it cannot fully compensate for the absence of a truly synchronous acquisition.

Another important limitation is the limited variability of the constructed dataset. For development purposes, the dataset had a short duration, which reduces the natural variability that would be observed in longer gait recordings.

Therefore, while the dummy dataset supported controlled testing and implementation of the processing pipeline, the performance of the system under real integrated acquisition conditions may differ from what was observed during offline testing.

4.2.2 Limited Experimental Validation

Another important limitation concerns the limited experimental validation of the system. The evaluation was performed on a restricted number of acquisitions, without a large-scale study or clinical validation. Consequently, it is not possible to generalize the results or to fully assess the reliability of the extracted indicators across different populations and acquisition conditions.

A further limitation is related to the algorithms implemented in the second phase of the project. The amount of data used to test the proposed processing methods was not sufficient to fully validate the adopted methodological choices. The implemented approaches were able to identify gait phases and extract the related metrics in the tested dataset, but their robustness cannot be guaranteed without more extensive experimental validation.

A larger dataset involving multiple subjects, repeated sessions, and different walking speeds would be necessary to assess the consistency and generalizability of the results.

It is important to underline that the main objective of this prototype was not to provide a fully validated analytical framework, but rather to develop a multimodal software environment capable of acquiring, processing, visualizing, and exporting data. Therefore, the focus of the work was on demonstrating technical feasibility and practical applicability, rather than exhaustive validation.

4.2.3 Simplified Physiological Interpretation

The physiological interpretation of the extracted indicators represents another limitation of this project. Although features such as the H_α index, the Heat

Distribution Index (HDI), and EMG activation were introduced to describe different aspects of tissue electrical response, thermal distribution, and muscle activity, their interpretation remains simplified.

The relationship between these indicators and the underlying physiological processes is not direct and may be influenced by several factors, including sensor placement, movement execution, tissue characteristics, and acquisition conditions. Therefore, caution is required when associating these metrics with specific physiological or pathological conditions.

In this work, the extracted indicators were used as exploratory descriptors rather than clinically validated biomarkers. A direct correlation between these indicators and specific physiological conditions would require further experimental and medical validation.

4.2.4 Computational and Mobile Constraints

Finally, one of the intrinsic limitations of the project is related to the platform on which it was developed, namely a mobile device. The application was tested on a Google Pixel 5, which imposed constraints in terms of memory usage, computational load, and real-time rendering.

These limitations required specific architectural choices to manage the trade-off between performance and usability. In particular, real-time acquisition had to remain lightweight and rely on efficient algorithms, limiting the use of more advanced and computationally intensive techniques during online visualization.

As a result, some processing steps were simplified during real-time operation, potentially reducing the accuracy of the extracted metrics. However, this limitation was partially mitigated in offline analysis, where the recorded signal was available and more advanced processing could be applied.

4.3 Future Work

Integration with a Fully Multisensor Device

Future development of this project should primarily focus on integration with the final hardware device, capable of embedding a fully multisensorized system within a single knee sleeve and communicating directly with the mobile application. In this context, it will be essential to transition from a multimodal prototype framework to a fully integrated wearable monitoring system.

The implementation of a multisensor knee sleeve capable of acquiring temperature, bioimpedance, inertial data, and EMG signals from a single device would eliminate the need for simulated data derived from different external devices. This would allow the proposed packet structure and parsing pipeline to be validated under real acquisition conditions.

At the same time, a unified hardware platform would provide an important tool for debugging and validation, enabling the verification of temporal consistency of the received data, the quality of the extracted features, and their comparison with gold-standard gait analysis systems.

Experimental Validation and Dataset Expansion

Another key aspect of future work concerns comprehensive experimental validation. The proposed system should be tested on a larger number of subjects in order to assess the reliability and repeatability of the adopted signal processing methodologies.

The system should be evaluated under different acquisition conditions, including varying walking speeds, repeated sessions, and, ideally, patients affected by knee osteoarthritis. Such validation would improve the reliability of the extracted metrics and the robustness of gait segmentation.

Given the context of digital health and telerehabilitation, this step is fundamental to move from technical feasibility toward real biomedical applicability.

Physiological Interpretability of the Metrics

Another important direction concerns the physiological interpretability of the extracted metrics. Future work should investigate the relationship between the Heat Distribution Index (HDI), H_α , EMG activation, and established biomechanical or clinical reference measures.

In this way, the proposed indicators could be investigated as potential interpretable biomarkers related to muscle function, tissue condition, and local thermal variations, with possible relevance in the context of personalized monitoring.

Extension toward Remote Monitoring Systems

Finally, the mobile platform itself could be extended into a more complex remote monitoring system. The application already supports configurable acquisition settings, session storage, visualization modules, and data export functionalities, providing a solid foundation for future improvements.

Potential developments include automatic report generation, integration with cloud-based systems, and longitudinal tracking of patient data. The introduction of clinician-oriented dashboards for remote rehabilitation follow-up could further enhance the usability of the system.

In this perspective, the proposed platform could evolve into a structured m-health solution for continuous and personalized knee monitoring.

5 Conclusion

5.1 Summary of the Work

This thesis presented the design and development of a mobile application for a sensorized knee sleeve system intended to support the monitoring of knee osteoarthritis within the framework of digital health and rehabilitation. The work was motivated by the need for technological solutions able to support a more continuous observation of the knee condition through wearable devices and mobile tools.

From a technical perspective, the proposed system was conceived as a research-oriented mobile platform capable of interfacing with wearable devices, managing Bluetooth Low Energy communication, and supporting the acquisition of physiological data. In the first phase of development, the application was structured as a modular hub with separate workflows for temperature and bioimpedance monitoring. This architecture made it possible to implement the main functionalities defined by the project requirements and to create a coherent and extensible software environment.

The second phase of the work focused on extending the platform toward a future scenario involving a fully multimodal knee sleeve. Since the final integrated hardware was not yet available, a prototype environment based on simulated multimodal data was developed. This phase made it possible to define and test a complete processing pipeline, including packet architecture design, binary parsing, CRC-based integrity verification, synchronization, and multimodal feature extraction. All the algorithms were first developed and tested offline in Python and were then integrated into the mobile application through a Developer Mode.

In summary, the work demonstrated the technical feasibility of a mobile solution for multimodal knee monitoring. At the same time, the thesis acknowledges that the proposed system is still a prototype and presents several limitations, including the use of simulated multimodal data and limited experimental validation. For this reason, the developed platform should be considered as a technological basis for future developments rather than as a fully validated clinical monitoring system.

5.2 Main Contributions

The first main contribution of this thesis is the development of a modular mobile application with an architecture able to support different wearable sensing systems for knee monitoring within a single platform. This result is important because it provides a flexible software structure that can be reused and extended in future developments. At the same time, the application already supports fundamental tasks such as Bluetooth communication, real-time visualization, session management, and data export.

The second contribution is the implementation of communication and data management strategies specifically designed for mobile wearable applications. In particular, the work addresses important technical aspects such as MTU optimization, packet parsing, buffering, and local storage. These elements are essential for building a system able to manage continuous data acquisition in a stable and efficient way on a mobile device.

A third contribution is the development of the signal processing pipeline introduced for the implemented acquisition modalities. The proposed methods make it possible to extract and visualize parameters from thermal, bioimpedance, inertial, and EMG-related signals. Although their physiological interpretation still requires further validation, these indicators represent a first step toward a more analytical characterization of the knee condition during physical activity.

Another important contribution is the development of a prototype multimodal framework for the future integration of a fully multisensorized knee sleeve. Through dummy data simulation, packet design, parsing, synchronization, and offline testing in Python followed by integration in React Native, the work anticipates the requirements of a future device capable of integrating all these signals. At the same time, it provides a practical environment in which these solutions can already be tested, refined, and improved. In this sense, the work does not only address current software needs, but also prepares the basis for future hardware developments and possible clinical applications.

Finally, this thesis contributes by moving knee monitoring closer to a tangible digital health application. Indeed, this work shows how wearable and mobile technologies can support more continuous and personalized monitoring strategies.

Taken together, these contributions define a complete prototype framework. Although further validation is required, the developed system provides a technical foundation for future studies on multimodal knee monitoring.

6 Appendix: Digital Platform for Nutrition Support and Remote Follow-Up in Patients with Brain Injuries

Although this thesis is primarily focused on the development of a mobile application for a sensorized knee sleeve system, the work carried out during the visiting period also included another software development activity in digital health.

This appendix was added to briefly describe that additional project. The project focused on developing a digital platform for nutrition support and remote follow-up for patients with brain injuries.

Therefore, the clinical context differed from that of knee osteoarthritis, but the type of work remained aligned with the general themes of this thesis. Indeed, this project also includes aspects related to mobile health, remote monitoring, patient support, and data management.

This activity was part of the overall experience conducted during the research period and represents a useful, complementary example of digital health application development. In particular, the clinician-oriented dashboard developed within this platform can be seen as a possible reference for future extensions of the primary application, where remote access to patient data, follow-up management, and long-term monitoring could become important components. For this reason, it is presented in the appendix to provide a more complete view of the work carried out during the research period.

6.1 Clinical Context and Project Rationale

Traumatic brain injury (TBI) is a major cause of long-term disability and may lead to chronic health problems that persist beyond the acute phase of care [36].

After TBI, overweight and obesity can represent important clinical issues, as weight-related problems may be associated with hypertension, diabetes, poorer

self-reported health, and increased morbidity in this population [37, 38]. For this reason, nutritional support and lifestyle management may play an important role in long-term rehabilitation and follow-up.

Structured lifestyle interventions adapted for people with TBI have shown promising results. For example, previous work on the GLB-TBI program, a modified version of an evidence-based diabetes prevention intervention, reported weight loss and improvements in metabolic and behavioral outcomes [38].

However, access to in-person support is not always easy, especially when patients live far from rehabilitation centers or experience transportation and accessibility barriers. From this perspective, telehealth approaches may help extend support beyond the clinical setting and improve continuity of care [39].

The development of a digital platform for nutrition support and remote follow-up was therefore considered a useful response to both clinical and organizational needs.

6.2 Objectives of the Platform

The main goal of the platform was to support nutrition management and remote follow-up in patients with brain injuries.

The system was designed to connect the patient side and the clinician side within the same digital workflow, making daily monitoring, communication, and follow-up easier to organize.

On the patient side, the aim was to provide a mobile application for meal tracking, calorie monitoring, water intake monitoring, access to nutritional information, reminders, and scheduled remote meetings for education or remote follow-up.

On the clinician side, the aim was to provide a web-based dashboard for managing nutrition plans, monitoring patient data, and organizing Zoom follow-up calls.

In summary, the platform was conceived as a telerehabilitation tool able to combine patient data collection with clinician monitoring.

6.3 Overall System Architecture

The platform was developed as a multi-layer system composed of three main components: a mobile application for patients, a web dashboard for healthcare professionals, and a backend infrastructure connecting the two interfaces, indeed the backend managed authentication, data storage, synchronization, and communication between the different parts of the system.

The general workflow followed a bidirectional logic. Patients entered daily data through the mobile application, while clinicians reviewed these data and managed follow-up activities through the dashboard. The backend ensured continuity between the two sides and supported a single workflow for monitoring and remote support.

6.4 Mobile Application for Patients

The mobile application represented the patient-facing component of the platform. It was designed to support daily nutrition-related self-management through a simple and accessible interface.

Through the app, patients could log meals, monitor caloric intake in relation to assigned goals, track water consumption, and visualize their progress through indicators and progress displays. The application also included a calendar section and direct access to scheduled Zoom meetings for remote follow-up. The main patient-side interfaces are shown in Figures 6.1 and 6.2.

Food registration was supported through both text-based search and barcode recognition. These functions were implemented through the FatSecret API at the server level. In practice, the mobile application sent requests to the backend, which managed the external API calls and returned the corresponding nutritional information to the app.

This choice reduced the complexity of the mobile application and avoided exposing external API credentials on the client side.

A reminder system was also implemented to support patients in recording their meals more consistently. Specifically, using the expo-notifications library, the application sends a notification every six hours starting at 8:00 a.m., reminding patients to log their breakfast, lunch, and dinner throughout the day. This feature was designed to help patients maintain a more regular daily routine and reduce the likelihood of forgetting to enter important information. This is particularly relevant for individuals with traumatic brain injury (TBI), as the literature highlights memory impairment as one of the main cognitive difficulties experienced by this group of patients. Therefore, the reminder system represents a practical way to support patients in their daily self-monitoring activities and to make the application more accessible and useful for their specific needs[40].



Figure 6.1: Mobile application interfaces. The left screen shows the calendar page with a scheduled Zoom appointment, while the right screen shows the nutrition dashboard with calorie tracking, water intake monitoring, food search, and logged meals.

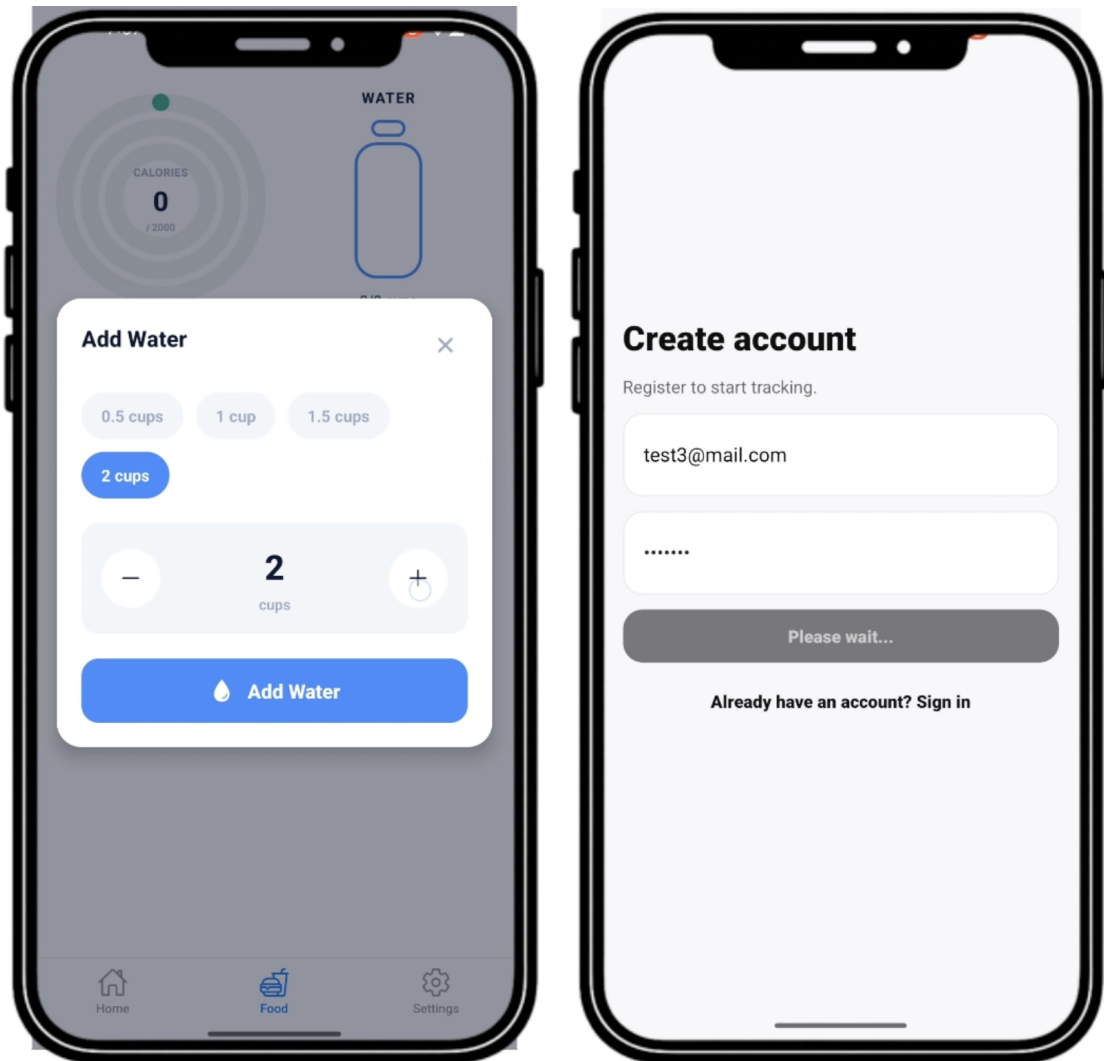


Figure 6.2: Additional mobile application interfaces. The left screen shows the modal for adding water intake, while the right screen shows the account registration page.

6.5 Web Dashboard for Healthcare Providers

The web dashboard was developed to help clinicians manage patients, define nutrition plans, schedule follow-up meetings, and monitor recorded data.

It included three main pages: meeting management, nutrition plan setup, and patient data monitoring. Access was protected through an admin key. After validation, the dashboard communicated with the backend through dedicated admin routes, keeping clinician-side operations separate from the patient login flow.

The meeting management page allowed the clinician to select a registered patient by email, enter a meeting title and Zoom link, and define the date, start time, and duration of the appointment. After submission, the backend stored the meeting in the PostgreSQL database and made it available in the patient's mobile calendar. This page is shown in Figure 6.3.

The nutrition plan page allowed the clinician to define the patient's daily calorie target, hydration goal, and macronutrient distribution. The interface converted the selected percentages into grams of proteins, carbohydrates, and fats according to the total daily calories. A visual bar displayed the macronutrient balance before saving the plan. Once submitted, the backend stored the updated goals and made them available to the mobile application. This page is shown in Figure 6.4.

The patient monitoring page displayed the nutritional data recorded by users, including daily calories, macronutrients, water intake, and last update time. It also included a CSV export function, allowing clinicians to download the stored data for further organization, analysis, or reporting. This interface is shown in Figure 6.5.

The screenshot shows a web browser window with the URL `https://fatsecret-backend2-production1.up.railway.app/admin/`. The page is titled "Meeting Manager" and has a "NEW / ACTIVE" status indicator. It features a "Login" button and a "Select User" dropdown menu with "test3@mail.com" selected. Below this, there are input fields for "Meeting Title" (containing "test") and "Zoom URL" (containing "testlink.com"). There are also fields for "Date" (8/2/24/2024), "Start Time" (09:00 AM), and "Duration (min)" (75). A "SUMMARY" section shows the meeting details: "Start: 2024-02-14 at 09:00 AM" and "End: 20:15 AM (7h 15m)". At the bottom, there is a blue "Schedule Meeting" button.

Figure 6.3: Web dashboard page for meeting management. The clinician can select a patient, define date, time, duration, title, and Zoom link, and save the follow-up appointment.

The screenshot shows a web browser window with the URL `https://fatsecret-backend2-production1.up.railway.app/admin/set_goals.html`. The page is titled "Nutrition Plan" and has a "Login" button. It features a "Select User" dropdown menu with "test3@mail.com" selected. Below this, there are sliders for "Daily Calories (kcal)" (set to 2000), "Water Goal (cups)" (set to 8), and "Macronutrient Split (%)". The macronutrient split is shown as a horizontal bar with three segments: Protein (30% - 150g), Carbs (40% - 200g), and Fats (30% - 47g). A "Total:" label is at the bottom of the bar. At the bottom, there is a blue "Update User Goals" button.

Figure 6.4: Web dashboard page for nutrition plan management, including calorie target, hydration goal, and macronutrient distribution.

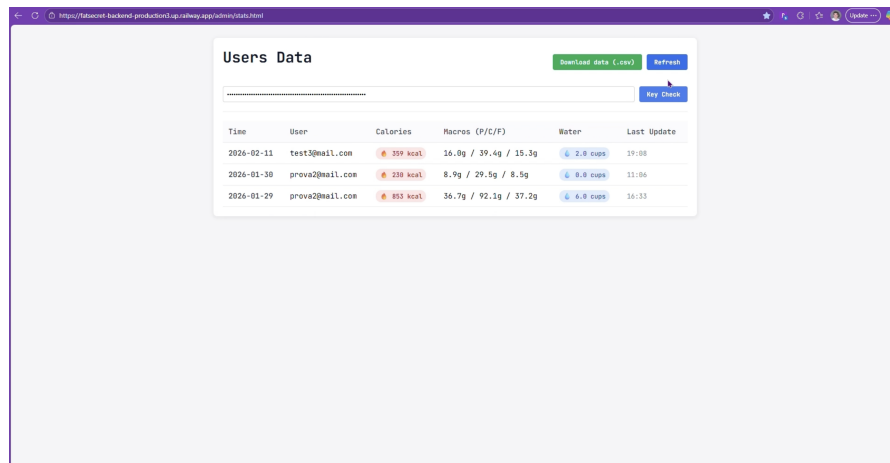


Figure 6.5: Web dashboard page for patient monitoring, showing nutritional data, water intake, last update time, and export options.

6.6 Backend, Authentication, and Data Synchronization

The backend was implemented as a Node.js server written in JavaScript and deployed on Railway. The same Railway environment was used to host a PostgreSQL database, which represented the central storage layer of the platform.

At startup, the server initialized the main database tables required by the system: `users` for authentication, `meetings` for Zoom appointments, `user_goals` for personalized nutrition targets, `user_daily_stats` for daily nutritional data, and `user_stats_log` for historical logging.

Authentication was managed through two different mechanisms. Patient registration and login were handled using bcrypt for password hashing and JWT for session management. During registration, the password was hashed before being stored in the database. During login, the entered password was compared with the stored hash and, if valid, the server generated a signed JWT token.

Clinician-side routes were instead protected through an `x-admin-key` header. This implementation separated dashboard operations from the authentication flow used by patients in the mobile application.

The backend also managed synchronization between the mobile application and the dashboard. The mobile app could retrieve personalized nutrition goals, send updated daily values for calories, macronutrients, and water intake, and retrieve scheduled meetings. Nutritional data sent from the app were stored both in a daily

summary table and in a historical log table, preserving current-day values as well as records over time.

The web dashboard was built on the same backend logic. Its pages used backend routes to create Zoom meetings, update patient goals, retrieve monitoring data, and export CSV files. In this way, the server handled the database operations and the exchange of data required by both interfaces.

Another important component was the FatSecret API integration. The mobile app did not communicate directly with FatSecret, instead, the backend acted as an intermediate layer for food search, barcode lookup, and nutritional data retrieval. This solution simplified the client-side implementation and avoided exposing external API credentials inside the mobile application.

6.7 Potential Impact and Future Developments

The developed platform represents a practical example of how mobile and web technologies can support remote nutrition management in patients with brain injuries.

From the patient side, it may support daily routines through meal tracking, hydration monitoring, reminders, and access to remote appointments. From the clinician side, it brings together nutrition goals, patient data, meeting management, and data export within the same workflow.

By collecting daily data through the mobile application and making them accessible through the web dashboard, the platform may help reduce information gaps between patients and healthcare providers. At the same time, it should be considered a prototype and not a clinically validated tool. Further testing is needed to evaluate its reliability, usability, and clinical impact.

Future developments could include integration with wearable devices and activity tracking systems. Through Health Connect, the platform could collect additional information from compatible devices and applications, potentially including platforms such as Garmin, Oura Ring, Fitbit, or Apple Watch. This would allow nutrition-related data to be combined with other health indicators, such as physical activity, heart rate, sleep, and energy expenditure.

In this perspective, the platform could evolve into a more complete digital health tool for remote monitoring and personalized rehabilitation support.

Bibliography

- [1] Jaimie D. Steinmetz et al. «Global, regional, and national burden of osteoarthritis, 1990–2020 and projections to 2050: a systematic analysis for the Global Burden of Disease Study 2021». In: *The Lancet Rheumatology* 5.9 (2023), e508–e522. DOI: 10.1016/S2665-9913(23)00163-7 (cit. on p. 3).
- [2] David J. Hunter and Sita Bierma-Zeinstra. «Osteoarthritis». In: *The Lancet* 393.10182 (2019), pp. 1745–1759. DOI: 10.1016/S0140-6736(19)30417-9 (cit. on pp. 3–6, 8).
- [3] World Health Organization. *Osteoarthritis*. <https://www.who.int/news-room/fact-sheets/detail/osteoarthritis>. Accessed: 2026-05-01. 2023 (cit. on p. 3).
- [4] M. M. Alabdullah, Z. N. Murrell-Smith, S. Q. Xie, and A. Liu. «Wearable sensing technology for knee joint monitoring and functional assessment: a systematic review». In: *IEEE Journal of Biomedical and Health Informatics* (2026). Early access. DOI: 10.1109/JBHI.2026.3679290 (cit. on pp. 3, 8).
- [5] Su'an Tang et al. «Osteoarthritis». In: *Nature Reviews Disease Primers* 11.1 (2025), p. 10. DOI: 10.1038/s41572-025-00594-6 (cit. on p. 3).
- [6] Di Chen, Jie Shen, Weiwei Zhao, Ting Wang, Leilei Han, Jennifer L. Hamilton, and Hee-Jeong Im. «Osteoarthritis: toward a comprehensive understanding of pathological mechanism». In: *Bone Research* 5 (2017), p. 16044. DOI: 10.1038/boneres.2016.44 (cit. on pp. 3, 4).
- [7] Vilim Molnar et al. «Cytokines and Chemokines Involved in Osteoarthritis Pathogenesis». In: *International Journal of Molecular Sciences* 22.17 (2021), p. 9208. DOI: 10.3390/ijms22179208 (cit. on p. 4).
- [8] Narcis B. Tudorachi, Eugenia E. Totu, Adrian Fifere, Victor Ardeleanu, Veronica Mocanu, Constantin Mircea, Ismail Isildak, Katarina Smilkov, and Elena M. Cărauşu. «The Implication of Reactive Oxygen Species and Antioxidants in Knee Osteoarthritis». In: *Antioxidants* 10.6 (2021), p. 985. DOI: 10.3390/antiox10060985 (cit. on p. 4).

- [9] El-Sayed E. Mehana, Asmaa F. Khafaga, and Samar S. El-Blehi. «The role of matrix metalloproteinases in osteoarthritis pathogenesis: An updated review». In: *Life Sciences* 234 (2019), p. 116786. DOI: 10.1016/j.lfs.2019.116786 (cit. on p. 4).
- [10] F. Héraud, A. Héraud, and M. F. Harmand. «Apoptosis in normal and osteoarthritic human articular cartilage». In: *Annals of the Rheumatic Diseases* 59.12 (2000), pp. 959–965. DOI: 10.1136/ard.59.12.959 (cit. on p. 5).
- [11] Hyun Sook Hwang and Hyun Ah Kim. «Chondrocyte apoptosis in the pathogenesis of osteoarthritis». In: *International Journal of Molecular Sciences* 16.11 (2015), pp. 26035–26054. DOI: 10.3390/ijms161125943 (cit. on p. 5).
- [12] Gillian A. Hawker, L. Stewart, M. R. French, J. Cibere, J. M. Jordan, L. March, M. Suarez-Almazor, and R. Gooberman-Hill. «Understanding the pain experience in hip and knee osteoarthritis: an OARSI/OMERACT initiative». In: *Osteoarthritis and Cartilage* 16.4 (2008), pp. 415–422. DOI: 10.1016/j.joca.2007.12.017 (cit. on p. 5).
- [13] Ans Van Ginckel, Kim L. Bennell, Penny K. Campbell, Tim V. Wrigley, David J. Hunter, and Rana S. Hinman. «Location of knee pain in medial knee osteoarthritis: patterns and associations with self-reported clinical symptoms». In: *Osteoarthritis and Cartilage* 24.7 (2016), pp. 1135–1142. DOI: 10.1016/j.joca.2016.01.986 (cit. on p. 5).
- [14] David J. Hunter, Jason J. McDougall, and Francis J. Keefe. «The symptoms of osteoarthritis and the genesis of pain». In: *Rheumatic Disease Clinics of North America* 34.3 (2008), pp. 623–643. DOI: 10.1016/j.rdc.2008.05.004 (cit. on p. 5).
- [15] Abhishek Abhishek and Michael Doherty. «Diagnosis and clinical presentation of osteoarthritis». In: *Rheumatic Disease Clinics of North America* 39.1 (2013), pp. 45–66. DOI: 10.1016/j.rdc.2012.10.007 (cit. on p. 5).
- [16] Michael A. Clynes, Karen A. Jameson, Mark H. Edwards, Cyrus Cooper, and Elaine M. Dennison. «Impact of osteoarthritis on activities of daily living: does joint site matter?» In: *Aging Clinical and Experimental Research* 31.8 (2019), pp. 1049–1056. DOI: 10.1007/s40520-019-01163-0 (cit. on pp. 5, 7).
- [17] Patricia A. Parmelee, Caitlan A. Tighe, and Natalie D. Dautovich. «Sleep disturbance in osteoarthritis: linkages with pain, disability, and depressive symptoms». In: *Arthritis Care & Research* 67.3 (2015), pp. 358–365. DOI: 10.1002/acr.22459 (cit. on p. 5).

- [18] Patrick H. Finan et al. «Discordance between pain and radiographic severity in knee osteoarthritis: findings from quantitative sensory testing of central sensitization». In: *Arthritis & Rheumatism* 65.2 (2013), pp. 363–372. DOI: 10.1002/art.34646 (cit. on p. 6).
- [19] National Institute for Health and Care Excellence. *Osteoarthritis in over 16s: diagnosis and management*. NICE guideline NG226. Accessed: 2026-05-01. 2022. URL: <https://www.nice.org.uk/guidance/ng226> (cit. on pp. 6, 7).
- [20] Sharon L. Kolasinski et al. «2019 American College of Rheumatology/Arthritis Foundation guideline for the management of osteoarthritis of the hand, hip, and knee». In: *Arthritis Care & Research* 72.2 (2020), pp. 149–162. DOI: 10.1002/acr.24131 (cit. on pp. 6, 7).
- [21] Raveendhara R. Bannuru et al. «OARSI guidelines for the non-surgical management of knee, hip, and polyarticular osteoarthritis». In: *Osteoarthritis and Cartilage* 27.11 (2019), pp. 1578–1589. DOI: 10.1016/j.joca.2019.06.011 (cit. on pp. 6, 7).
- [22] Belinda J. Lawford, Michelle Hall, Rana S. Hinman, Martin van der Esch, Alison R. Harmer, Lisa Spiers, Anna Kimp, Andrea Dell’Isola, and Kim L. Bennell. «Exercise for osteoarthritis of the knee». In: *Cochrane Database of Systematic Reviews* 12 (2024), p. CD004376. DOI: 10.1002/14651858.CD004376.pub4 (cit. on p. 6).
- [23] Kim L. Bennell et al. «Effectiveness of an Internet-Delivered Exercise and Pain-Coping Skills Training Intervention for Persons With Chronic Knee Pain: A Randomized Trial». In: *Annals of Internal Medicine* 166.7 (2017), pp. 453–462. DOI: 10.7326/M16-1714 (cit. on p. 6).
- [24] J. F. M. Holla, M. van der Leeden, D. L. Knol, L. D. Roorda, W. K. H. A. Hilberdink, W. F. Lems, M. P. M. Steultjens, and J. Dekker. «Predictors and outcome of pain-related avoidance of activities in persons with early symptomatic knee osteoarthritis: a five-year followup study». In: *Arthritis Care & Research* 67.1 (2015), pp. 48–57. DOI: 10.1002/acr.22381 (cit. on p. 7).
- [25] M. F. Pisters, C. Veenhof, G. M. van Dijk, and J. Dekker. «Avoidance of activity and limitations in activities in patients with osteoarthritis of the hip or knee: a 5 year follow-up study on the mediating role of reduced muscle strength». In: *Osteoarthritis and Cartilage* 22.2 (2014), pp. 171–177. DOI: 10.1016/j.joca.2013.12.007 (cit. on p. 7).
- [26] Grace H. Lo et al. «Symptom assessment in knee osteoarthritis needs to account for physical activity level». In: *Arthritis & Rheumatology* 67.11 (2015), pp. 2897–2904. DOI: 10.1002/art.39271 (cit. on p. 7).

- [27] Nirali Shah, Kerry Costello, Akshat Mehta, and Deepak Kumar. «Applications of Digital Health Technologies in Knee Osteoarthritis: Narrative Review». In: *JMIR Rehabilitation and Assistive Technologies* 9.2 (2022), e33489. DOI: 10.2196/33489. URL: <https://rehab.jmir.org/2022/2/e33489> (cit. on pp. 7, 8).
- [28] M. Reza Naeemabadi, Hesam Fazlali, Samira Najafi, Birthe Dinesen, and John Hansen. «Telerehabilitation for Patients With Knee Osteoarthritis: A Focused Review of Technologies and Teleservices». In: *JMIR Biomedical Engineering* 5.1 (2020), e16991. DOI: 10.2196/16991. URL: <https://biomedeng.jmir.org/2020/1/e16991> (cit. on p. 7).
- [29] Axel Georg Meender Schäfer, Christoff Zalpour, Harry von Piekartz, Toby Maxwell Hall, and Volker Paelke. «The Efficacy of Electronic Health-Supported Home Exercise Interventions for Patients With Osteoarthritis of the Knee: Systematic Review». In: *Journal of Medical Internet Research* 20.4 (2018), e152. DOI: 10.2196/jmir.9465. URL: <https://www.jmir.org/2018/4/e152> (cit. on p. 8).
- [30] Sarah Lang, Judith McLelland, David MacDonald, and David F. Hamilton. «Do digital interventions increase adherence to home exercise rehabilitation? A systematic review of randomised controlled trials». In: *Archives of Physiotherapy* 12 (2022), p. 24. DOI: 10.1186/s40945-022-00148-z. URL: <https://archivesphysiotherapy.biomedcentral.com/articles/10.1186/s40945-022-00148-z> (cit. on p. 8).
- [31] Dylan Kobsar, Zaryan Masood, Heba Khan, Noha Khalil, Marium Yossri Kiwan, Sarah Ridd, and Matthew Tobis. «Wearable Inertial Sensors for Gait Analysis in Adults with Osteoarthritis: A Scoping Review». In: *Sensors* 20.24 (2020), p. 7143. DOI: 10.3390/s20247143. URL: <https://www.mdpi.com/1424-8220/20/24/7143> (cit. on pp. 8, 9).
- [32] Andrea Cafarelli et al. «Usability Assessment of Technologies for Remote Monitoring of Knee Osteoarthritis». In: *IEEE Open Journal of Engineering in Medicine and Biology* 5 (2024), pp. 476–484. DOI: 10.1109/OJEMB.2024.3407961. URL: <https://doi.org/10.1109/OJEMB.2024.3407961> (cit. on pp. 8, 9).
- [33] Annalisa Na and Thomas S. Buchanan. «Validating Wearable Sensors Using Self-Reported Instability among Patients with Knee Osteoarthritis». In: *PM&R* 13.2 (2021), pp. 119–127. DOI: 10.1002/pmrj.12393 (cit. on p. 8).
- [34] Wu Xiang, Jun-Yu Wang, Bing-jin Ji, Li-Jun Li, and Han Xiang. «Effectiveness of Different Telerehabilitation Strategies on Pain and Physical Function in Patients With Knee Osteoarthritis: Systematic Review and Meta-Analysis».

- In: *Journal of Medical Internet Research* 25 (2023), e40735. DOI: 10.2196/40735. URL: <https://www.jmir.org/2023/1/e40735> (cit. on p. 8).
- [35] Jonathan Camargo, Aditya Ramanathan, Will Flanagan, and Aaron Young. «A comprehensive, open-source dataset of lower limb biomechanics in multiple conditions of stairs, ramps, and level-ground ambulation and transitions». In: *Journal of Biomechanics* 119 (2021), p. 110320. DOI: 10.1016/j.jbiomech.2021.110320. URL: <https://www.sciencedirect.com/science/article/pii/S0021929021001007> (cit. on p. 20).
- [36] Centers for Disease Control and Prevention. *Facts About TBI*. Accessed 2026-05-23. 2025. URL: <https://www.cdc.gov/traumatic-brain-injury/data-research/facts-stats/index.html> (cit. on p. 84).
- [37] Laura E. Dreer, Jessica M. Ketchum, Thomas A. Novack, Jennifer Bogner, Elizabeth R. Felix, John D. Corrigan, Doug Johnson-Greene, and Flora M. Hammond. «Obesity and Overweight Problems Among Individuals 1 to 25 Years Following Acute Rehabilitation for Traumatic Brain Injury: A NIDILRR Traumatic Brain Injury Model Systems Study». In: *Journal of Head Trauma Rehabilitation* 33.4 (2018), pp. 246–256. DOI: 10.1097/HTR.0000000000000408 (cit. on p. 85).
- [38] Simon Driver et al. «Efficacy of the Diabetes Prevention Program Group Lifestyle Balance Program Modified for Individuals with TBI (GLB-TBI): Results from a 12-month Randomized Controlled Trial». In: *Annals of Behavioral Medicine* 57.2 (2023), pp. 131–145. DOI: 10.1093/abm/kaac036 (cit. on p. 85).
- [39] Simon Driver, Evan Elizabeth McShan, Monica Bennett, Stephanie Calhoun, Librada Callender, Chad Swank, and Rosemary Dubiel. «A randomized controlled trial protocol for people with traumatic brain injury enrolled in a telehealth delivered diabetes prevention program (tGLB-TBI)». In: *Contemporary Clinical Trials Communications* 35 (2023), p. 101191. DOI: 10.1016/j.conctc.2023.101191 (cit. on p. 85).
- [40] Luqmanul Hakim Abdul Razak, Tedd Denis, Yoghaanjaly Murugiah, Weng Kei Yoong, Zamzuri Idris, and Mohd Harizal Senik. «The Effect of Traumatic Brain Injury on Memory». In: *Malaysian Journal of Medical Sciences* 31.3 (June 2024), pp. 52–74. DOI: 10.21315/mjms2024.31.3.4. URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC11229567/> (cit. on p. 86).