iPROLEPSIS

D4.2 / The iPROLEPSIS patient and HCP apps (study version)

Editor	Contractual delivery date	Actual delivery date
Andreas Raptopoulos (WELLICS)	Dec 2023	Jun 2024
Deliverable type	**Dissemination level**	**Version - date**
OTHER	PU – Public	1.0 - 28/06/2024

Funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Health and Digital Executive Agency. Neither the European Union nor the European Health and Digital Executive Agency can be held responsible for them.

Deliverable ID

Project acronym	iPROLEPSIS
Project full title	Psoriatic arthritis inflammation explained through multi-source data analysis guiding a novel personalised digital care ecosystem
Grant Agreement ID	101095697

Deliverable number	D4.2
Deliverable title	The iPROLEPSIS patient and HCP apps (study version)
Work package	WP4 - Development of the iPROLEPSIS digital health ecosystem for personalised preventive care
Deliverable type	OTHER
Dissemination level	PU – Public
Version - date	1.0 - 28/06/2024

Contractual delivery date	Dec 2023
Actual delivery date	Jun 2024

Lead partner	WELLICS
Editor	Andreas Raptopoulos (WELLICS)
Contributors	WELLICS, AUTH, CERTH, INTRA, PLUX
Reviewed by	Dimitrios Konstantinidis (CERTH) Ilja Tchetverikov (CICERO)
Approved by	Leontios Hadjileontiadis (AUTH, Project Coordinator)

Keywords	Data Collection, Data Management System, Joint Landmarker Application, Keyboard, miPROLEPSIS Study Application, Smartphone Application, Wearables

Document history

Version	Date	Contributors	Action / status
0.1	20/11/2023	WELLICS	Document structure (table of contents) ready
0.3	31/05/2024	WELLICS, AUTH, INTRA, CERTH, PLUX	Completion of actual prototypes developments
0.5	10/06/2024	WELLICS, AUTH, INTRA, CERTH, PLUX	Preparation and provision of inputs for the deliverable
0.7	14/06/2024	WELLICS	Inputs integration, preparation of additional inputs and quality checks
0.8	18/06/2024	WELLICS	Updates on the prototypes and additional reporting
0.9	21/06/2024	WELLICS	Ready for internal review
0.10	26/06/2024	CERTH	Reviewed by Dimitrios Konstantinidis (CERTH)
0.11	26/06/2024	CICERO	Reviewed by Ilja Tchetverikov (CICERO)
0.12	27/06/2024	WELLICS	Document revised
0.13	28/06/2024	AUTH	Approved by the Project Coordinator
1.0	28/06/2024	AUTH	Submitted to the EC by the Project Coordinator

Contents

List of figures

List of tables

List of abbreviations

PsA	Psoriatic Arthritis
PsO	Psoriasis
PDPID	PsA Digital Phenotyping and Inflammation Drivers
WP	Work Package
FR	Functional Requirements
NFR	Non-functional Requirements
UR	User Requirements
DMS	Data Management System
OS	Operating System
HCPs	Healthcare Professionals
API	Application Programming Interface

Executive summary

The deliverable presents the miPROLEPSIS smartphone application, along its cloud-based back-end components, which are the main technical WP4 outcomes up to now and comprise the key technological tools for the implementation of iPROLEPSIS-PDPID study application. This report is complementary to the actual prototypes delivered along with it, and functions as a supporting document to the reader by providing information on how to use the various features, as well as technical details on their implementation.

More specifically, the main aspects covered in this document are: (1) the miPROLEPSIS PDPID smartphone application; (2) the miPROLEPSIS keyboard; (3) the miPROLEPSIS Joint Landmarker application for video assessment tests; (4) the cloud-based Data Management System; and (5) CI/CD tools and workflows. The PDPID study application is synthesized by all these features, integrated together into a single functional platform. The main scope of this platform is to function as a data collection tool, utilized during the PDPID study, in order to produce datasets used for model generation and training, as well as for the further evolution of the aforementioned features.

This deliverable is considered the first step (first development cycle) of the iPROLEPSIS digital ecosystem. The evolution of the system will be performed in an incremental manner divided in 4 development cycles, with 3 more prototypes scheduled until the end of the project. To this direction, the second development cycle of the project has already started, with its main outcomes scheduled for M23 of the project.

1 Introduction

The main objectives of iPROLEPSIS Work Package (WP) 4 are: (1) to design and develop the patient digital ecosystem, incorporating PsA monitoring, knowledge, targeted interventions and tailor AI-driven recommendations; (2) to implement the serious gaming suite for health and wellness; and (3) to design and develop the application for the Healthcare Professionals (HCPs), allowing them to remotely monitor patients and view predictions on their conditions. Focusing on the first objective, this deliverable presents the first version of the foreseen ecosystem (the PDPID study application), aiming at supporting the needs of the PDPID study namely: (1) to develop digital biomarkers related to PsA and (ii) to investigate (actionable) drivers (e.g., lifestyle) of inflammation in PsA.

1.1 Document scope

The document presents the iPROLEPSIS PDPID study application created to serve the needs of this study. The work described here was mainly performed in the context of tasks 4.3, 4.5 and 4.6. The type of the deliverable is characterized as “OTHER” combining both a prototype as well as a report. Thus, this document can be seen as the second part of this deliverable (report) and should be treated as complementary to the actual prototype. As such, it provides brief technical details on the application as well as a short manual regarding how to use the application.

The current version of the iPROLEPSIS application and overall system implements a subset of functionalities of the iPROLEPSIS digital ecosystem, as this was presented in deliverable D4.1. To this end, this document presents a status update on the user, functional and non-functional requirements covered by this application, providing a qualitative and quantitative indication of what is already in place (fully or partially) and what is expected to be realized in the future. Thus, this status is expected to further change as the developments progress and further updates will be provided in the next WP4 deliverables.

1.2 Document structure

D4.2 is divided in the following sections:

Section 2 – The PDPID application: Presents the PDPID study application, including a short user manual, technical details and a status update on the URs, FRs and NFRs as these were defined in previous deliverables submitted in the context of WP2 and WP4.

Section 3 – The miPROLEPSIS keyboard: Presents the miPROLEPSIS custom keyboard for collecting typing dynamics, including a short user manual and technical information, as well as a status update on the FRs and NFRs covered in this deliverable.

Section 4 – The miPROLEPSIS Joint Landmarker app for video assessment tests: Presents the miPROLEPSIS Joint Landmarker application that analyses hand/body movements of patients through video assessment exercises. The section includes a short user manual of this application, technical details and a status update on the FRs and NFRs covered in this deliverable.

Section 5 – The iPROLEPSIS cloud-based data management system: Presents the back-end implementation for supporting the PDPID study application. The section also includes a status update on the FRs and NFRs covered in this deliverable.

Section 6 – System orchestration, monitoring tools and integration: Provides an overview of the orchestration and monitoring tools used for integration, system development and resource management purposes, covering not only the current version of the application, but also the future ones.

Section 7 – Conclusions and next steps: Concludes the deliverable and briefly presents the next steps towards the next version of the miPROLEPSIS application and WP4 in general.

2 The PDPID application

2.1 The scope of the application

The scope of the miPROLEPSIS PDPID study app is **threefold**, namely:

1. **Data Collection:**
    - Continuous and unobtrusive data collection (via bluetooth) from wearable devices (Garmin Vivoactive 5[1]) including: raw accelerometer data, heart rate, stress levels, wellness data (i.e., activity, intensity and classification and number of steps), sleep session data (duration, score), Beat-to-beat intervals (BBI) data, body battery.
    - Raw accelerometer and gyroscope data from smartphone's embedded sensors (accelerometer, gyroscope).
    - User generated inputs from questionnaires (upon notification).
    - User generated inputs through video and photo assessment tests, utilizing the smartphone embedded sensors (camera). (See Sections 2.2.11 and 4)
    - Keystroke dynamics data (metadata), through the use of customized smartphone keyboard. (See Section 3)
2. **User interaction:**
    - Answer a questionnaire and flare declaration.
    - Review a summary of daily data collected.
    - Keep simple (private) notes on a daily basis.
    - Receive notifications to trigger actions (answer a questionnaire, take the video and photo active tests).
    - Initiate the processes of deleting user's data, deleting user's account and getting the user's data collected.
    - Perform a pairing process between the wearable and the smartphone.
3. **Data Upload:**
    - Periodic (every 10 minutes) data uploading on miPROLEPSIS cloud-based data management system (DMS) including:
        - The data collected from the paired wearable.
        - The data collected by utilizing the smartphone's embedded sensors (accelerometer and gyroscope).
    - On-demand (based on corresponding notification) data uploading on miPROLEPSIS cloud-based data management system (DMS) including:

- The results of the video assessment tests (3-D coordinates of skeletal joints) (See Section 4).
- The results of the photo assessment tests.
- The questionnaires' answers.

To this end the connection service between the smartphone and the DMS is **always running in the background**, with no need for an active instance of the PDPID application, enabling the continuous and unobtrusive collection and transfer of the data mentioned above. The table below provides an overview of the data collected and the scope of this process:

Table 1 Data collection overview.

Type	From	Scope
Keyboard data	Keyboard application/features	Collecting typing dynamics is a key component for data collection related to hand function (fine motor skills)
Images (toes and nails)	Smartphone camera sensor (photo assessment test feature)	Assessing PsA symptoms related to fingers, nails and toes
Skeletal joints	Smartphone camera sensor (video assessment test application/feature)	Hand/body movement analysis for the assessment of range of motion of PsA patients
Accelerometer data	Smartphone accelerometer and gyroscope sensors and wearable accelerometer data	Predict / evaluate physical activity levels
User inputs	Questionnaires provided in the miPROLEPSIS patient app	Patient assessment on his/her condition and lifestyle (pain, flare, diet)
Physical activity (steps, etc.)	Wearable device (Garmin Vivoactive 5)	Assess mobility status and investigate the association with disease inflammation (complementary to accelerometer data)
Sleep (duration, stages, quality)	Wearable device (Garmin Vivoactive 5)	Assess sleep status and disorders and investigate the association with disease inflammation
Stress (summary, continuous data)	Wearable device (Garmin Vivoactive 5)	Investigate the association of stress with disease inflammation
Heart Rate and HRV	Wearable device (Garmin Vivoactive 5)	Assess body response to disease inflammation
Body Battery	Wearable device (Garmin Vivoactive 5)	Assess overall condition of the user and investigate the association with disease inflammation

2.2 Using the application

2.2.1 Introduction

This section presents a brief manual of use for the PDPID study application. Each function supported is presented in a stepwise approach, combined with snapshots visualizing the navigation flow. The functionalities included in this version of the application are aligned with the project's DoA and were developed in collaboration with iPROLEPSIS medical partners as well as a selected group of potential users who provided their inputs in a series of biweekly workshops performed between September and December 2023. The UIs follow the visual identity implemented in the context of WP6, having as a goal the development of a common brand that will further evolve through the different versions of the miPROLEPSIS application.

2.2.2 General

The miPROLEPSIS PDPID study application is currently available for Android and very soon will be available for iOS. The two versions will provide the same functionalities and features. Below, a list is found outlining the application packages needed to be installed for each OS version.

Android (will be available via Google Play very soon)[1]

- miPROLEPSIS Joint Landmarker app for video assessment tests
- miPROLEPSIS PDPID study app

iOS (will be available via AppStore very soon)

- miPROLEPSIS Joint Landmarker app for video assessment tests
- miPROLEPSIS PDPID study app

The applications are available in four (4) languages, namely Dutch, English, Greek and Portuguese. For the development of the application features, both cross-platform (Native REACT) as well as native approaches were used. This decision is mainly based on the limitations and rules imposed by the two operating systems (Android and iOS), related with: (1) internal scheduling and battery use, (2) system processes, (3) utilization of the smartphone's embedded sensors, and (4) application delivery.

2.2.3 User Onboarding

Step 1: A list of predefined credentials (usernames and their passwords) will be distributed to each study site.

Step 2: Study team (nurse/physician) will assign each patient that wants to be enrolled with one of the credentials in the list, keeping track of which patient has been assigned each credential, and making sure that no two patients are assigned the same credentials. It is important for privacy reasons to safeguard the information of which patient corresponds to which username, as well as their passwords.

Step 3: The user will apply the assigned predefined usernames and passwords for logging in to the application.

2.2.4 Installing the application

Step 1: The application is available both on **Google Play** and **App Store**, under the name **miPROLEPSIS**, and it can be downloaded from there.

Step 2: Upon download the installation is performed automatically (similarly to all other smartphone applications).

2.2.4.1 Minimum Requirements

- **Smartphone:**
    - iOS: version 14 or higher
    - Android: version 11 (API level 30) or higher

[1] Currently, the Android version of the apps can be downloaded from here.

- **Wearable:** Garmin VivoActive 5
- **An account** (username and password) should be available, enabling the user to login into the application. (See Section 2.2.3)

2.2.5 Log in to the miPROLEPSIS PDPID study application

Step 1: Upon installing the application and by clicking on the **application icon**, the login screen appears.

Step 2: Upon the correct completion of the username and password, and by clicking Sign In, the user logs into the application and the miPROLEPSIS application home screen appears.

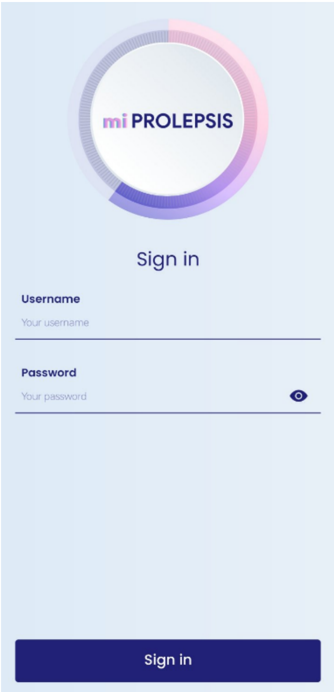

Figure 1 Login screen snapshot.

2.2.6 The miPROLEPSIS PDPID application home screen

The home screen of the miPROLEPSIS application and its available functionalities are presented in the figure below. From the home screen of the miPROLEPSIS application, the user of the app may:

- Access the application designated burger menu (See Section 2.2.7).
- Check if there are any open notifications for the availability of either a questionnaire or active photo and video tests (See Sections 2.2.10, 2.2.11 and 4).
- Access the “My Journey” section (See Section 2.2.12).
- View the day’s metrics that have been logged and synced from their paired Garmin Vivoactive 5 wearable (See Section 2.2.8).

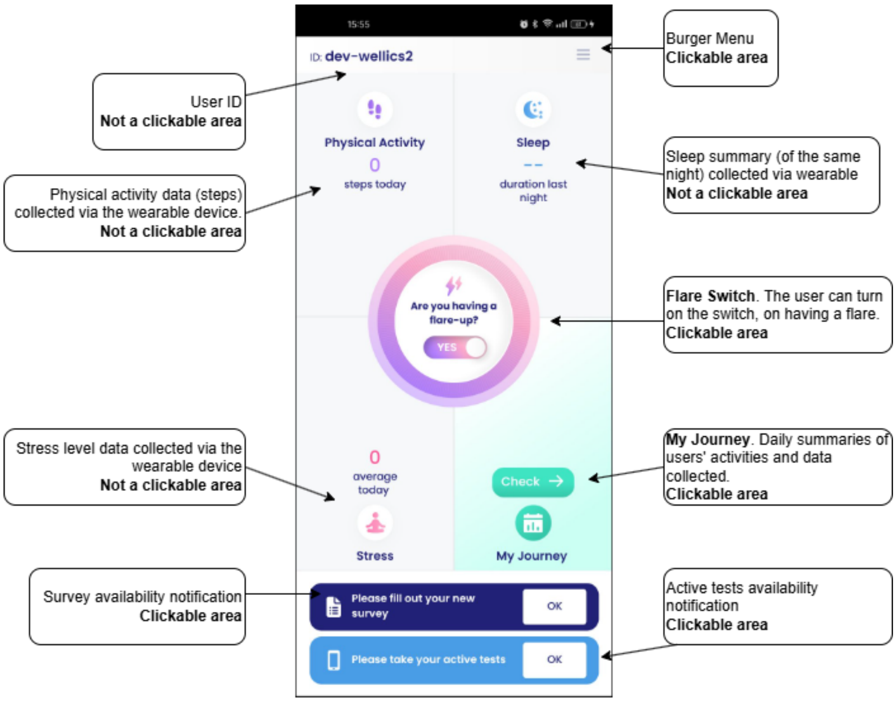

Figure 2 Home screen snapshot of the miPROLEPSIS application.

2.2.7 The miPROLEPSIS PDPID application burger menu options

Step 1: By clicking on to the Burger Menu, the menu opens. Currently, three options are available:

(1) the **Device** option: for connecting the wearable device with the application (See Section 2.2.8).

(2) the **Profile** option: for managing the settings of user profile (language, wearable location, perform data deletion, request access to data, and log out) (See Section 2.2.14).

(3) the **Active Tests** option: for the patient to be able to access the available photo and video based active tests the application offers (See Sections 2.2.11 and 4).

Note: Access to active tests from the burger menu is unobstructed, in the sense that the application will still present the active test selection screen and allow the user to take an active photo or video test, as long as they have not completed one on the same day. This behavior is in contrast to the access to active tests provided via the active test notification on the miPROLEPSIS home screen (See **Figure 3**).

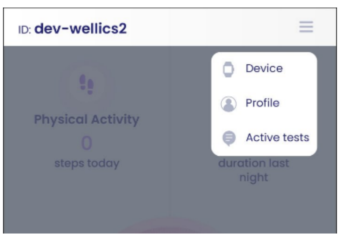

Figure 3 Burger Menu snapshot.

2.2.8 Connecting a wearable

Step 1: By clicking on the miPROLEPSIS application burger menu and then by clicking on the **Device** option the following screen appears, indicating that there is no connected device.

Figure 4 Wearable Connection initiation screen snapshot.

Step 2: By clicking **Start Search** a search for an available wearable device is initiated, and the available devices appear on screen (**Figure 5**). By **selecting** one of the devices and clicking **Connect** the pairing process is performed. By clicking **Cancel** (upper right screen corner) the connection procedure stops.

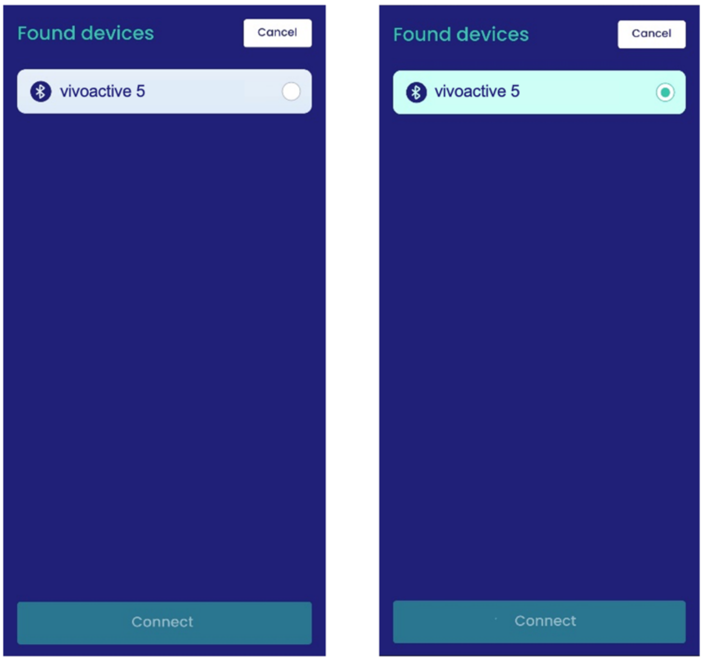

Figure 5 Available devices for pairing screen snapshot.

Note: Before pairing with the wearable, the Garmin SDK which is hosted in the application requires some personal data, so that it can perform its self-initialization (**Figure 6**).

Fill this form to be able to perform the pairing (**Figure 6**). Personal data are used solely for the initialization of the connection with the wearable and are neither stored locally or transmitted elsewhere for processing or storage.

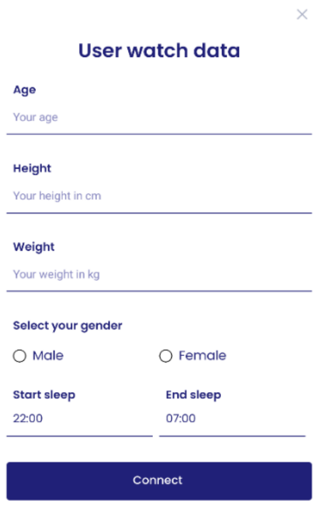

Figure 6 Wearable personal data - Initialization of connection.

Step 3: Upon successful completion a screen appears indicating that the connection process has been successfully **completed**. By clicking the "**<**" in the upper left screen corner, the user exits this process, and the home screen appears. By clicking "**Edit**" the user has the ability to disconnect the device (**Figure 7**).

Figure 7 Successful connection completion screen snapshot.

After the successful pairing of the Garming Vivoactive 5 wearable device, the data collection from the wearable begins running periodically and in the background. By clicking the Manual sync button at the bottom, the user has the option to trigger a manual sync between the application and the wearable.

Step 4: Unpairing the device can be performed in the device screen (**Figure 8**): **(1)** Press the **Edit** button; **(2)** Click on the **Remove device** button; and **(3)** The unpairing process is completed.

Figure 8 Unpairing a wearable device snapshot.

Prerequisites: To connect the wearable with the smartphone device (pairing process), the following prerequisites must be met:

(1) Make sure that the Garmin Vivoactive 5 is charged properly and turned on

(2) Make sure that the **smartphone's Bluetooth connection** is active.

- User can either turn on the Bluetooth of their device on their own or
- Rely on the application functionality itself, which asks the user to turn their Bluetooth on, after they have successfully logged in the application. User must select the **"Allow"** option.

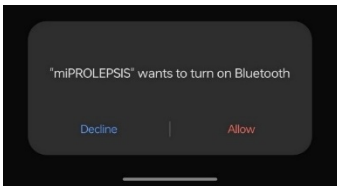

Figure 9 miPROLEPSIS asking to turn Bluetooth on.

2.2.9 Reporting a flare

Step 1: Switch the flare button ON when you (the user) are experiencing a flare. For **as long as** the button is on, the event is continuously being recorded and it is considered as a continuous experience of a flare.

By **turning the flare button OFF**, it is considered that a flare is no longer experienced (user's subjective opinion) and the flare will stop being recorded.

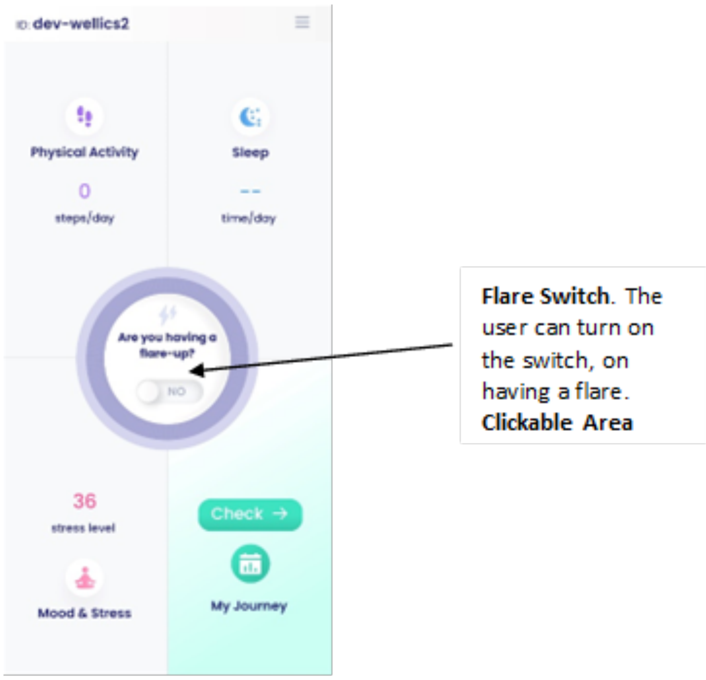

Figure 10 Using the flare button.

Step 2: Upon switching ON the flare button, a corresponding **notification** is generated at the bottom of the main dashboard screen, indicating that **a new questionnaire is available** for the user to fill in (**Figure 10**).

For as long as the flare button is turned on and being reported, the notification for the patient to fill in the specific questionnaire will be made available daily.

Step 3: Click **on the notification button ("OK")**, to proceed to fill in the survey for the flare (**Figure 11**).

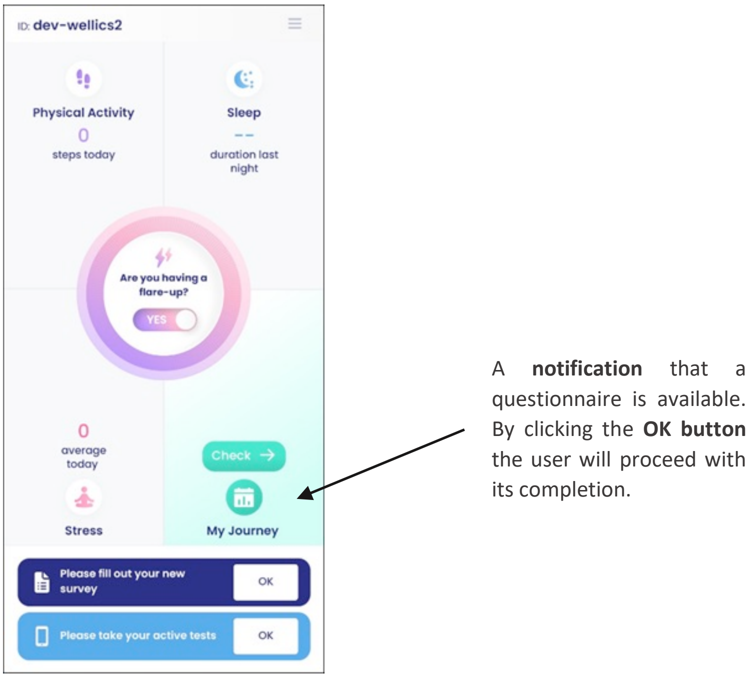

Figure 11 Questionnaire notification upon switching on the Flare button snapshot.

2.2.10 Answering the flare questionnaire

2.2.10.1 Introduction

There are **three types** of questions available in the specific questionnaire:

(1) Thermometer questions, where the user shall evaluate a parameter, by swiping (up or down) (**Figure 12**). These questions assess: pain and fatigue levels, sleep quality, skin problems, morning stiffness, functional capacity, and mood

(2) Single Select questions, where the user may select a single option from the available ones (**Figure 13**).

(3) Mannequin questions, where the user (un-) selects by clicking on certain body parts to indicate the points of discomfort (**Figure 14**).

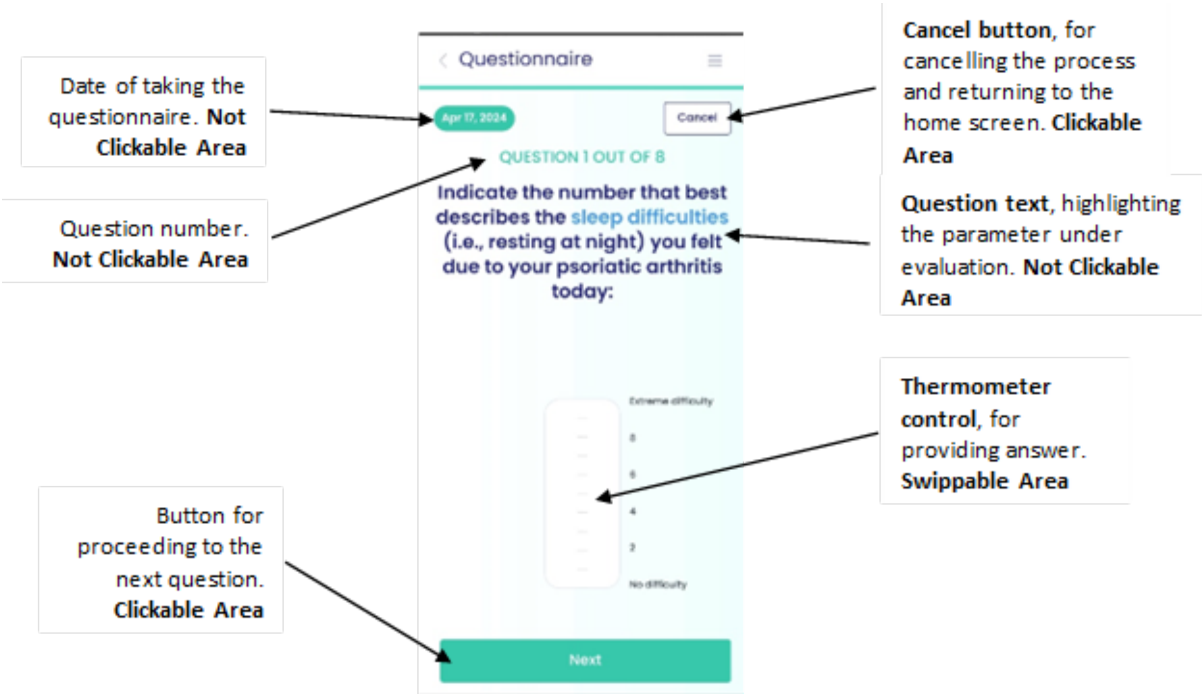

Figure 12 Snapshot of a thermometer question type.

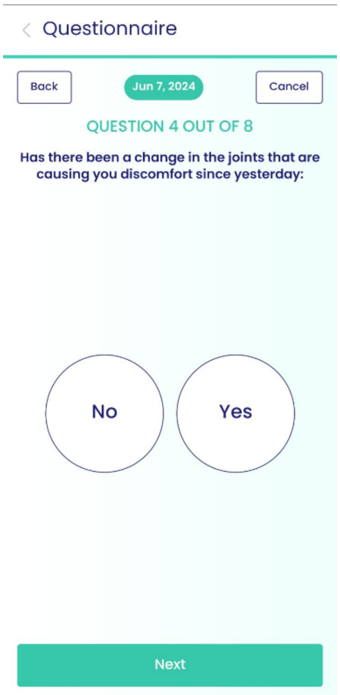

Figure 13 Snapshot of a single select question type.

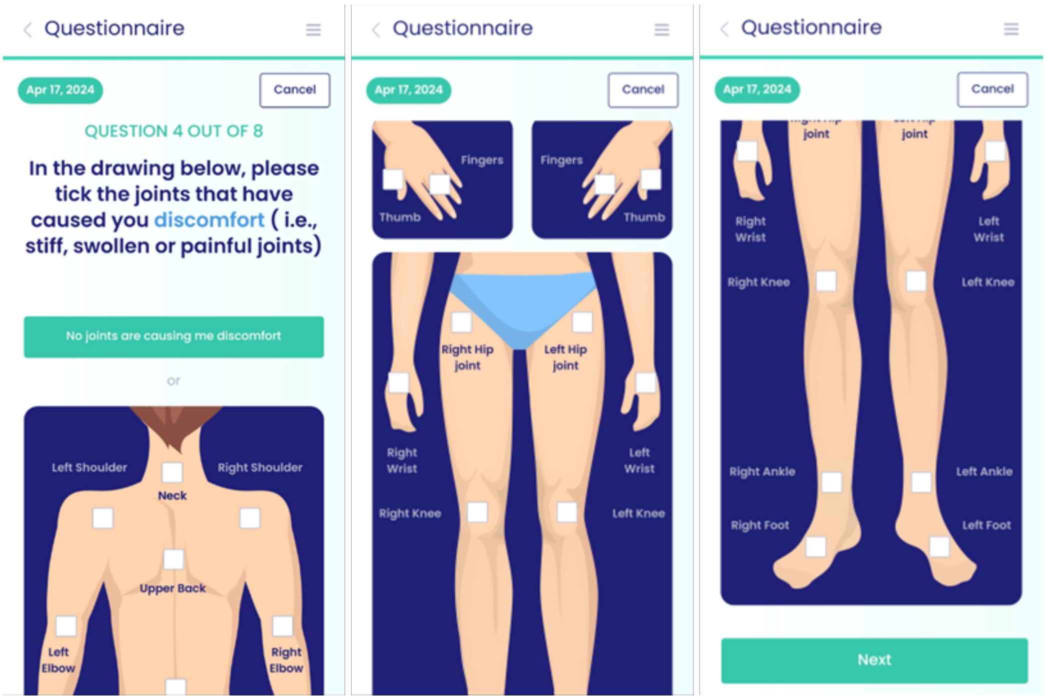

Figure 14 Snapshots of mannequin questions.

2.2.10.2 Answering process

Step 1: By clicking the "OK" button found in the notification generated upon reporting a flare, the flare questionnaire appears.

Step 2: By clicking Next, the user moves to the next question in sequence. In the case of clicking Next without providing an answer the question is considered skipped, and a corresponding notification appears for the user to confirm this action (**Figure 15**).

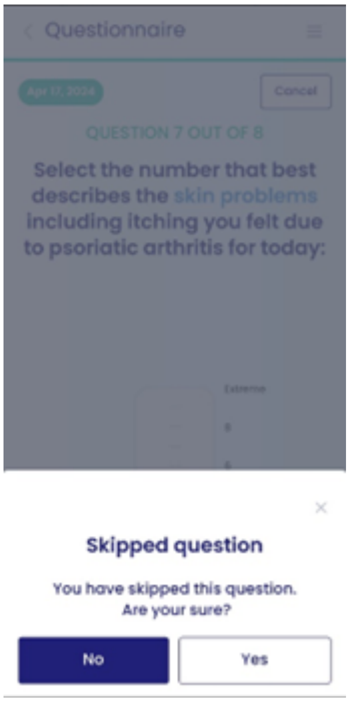

Figure 15 Message for confirming skipping of a question.

Step 3: Upon completion of a questionnaire, **a summary** of the questions and the answers provided appears to the user (**Figure 16**). By clicking on a question, the user may revisit the question and update their answer.

Step 4: Click on **Submit** to finalize and submit the questionnaire (**Figure 16**). After this step, the user cannot change their answers anymore.

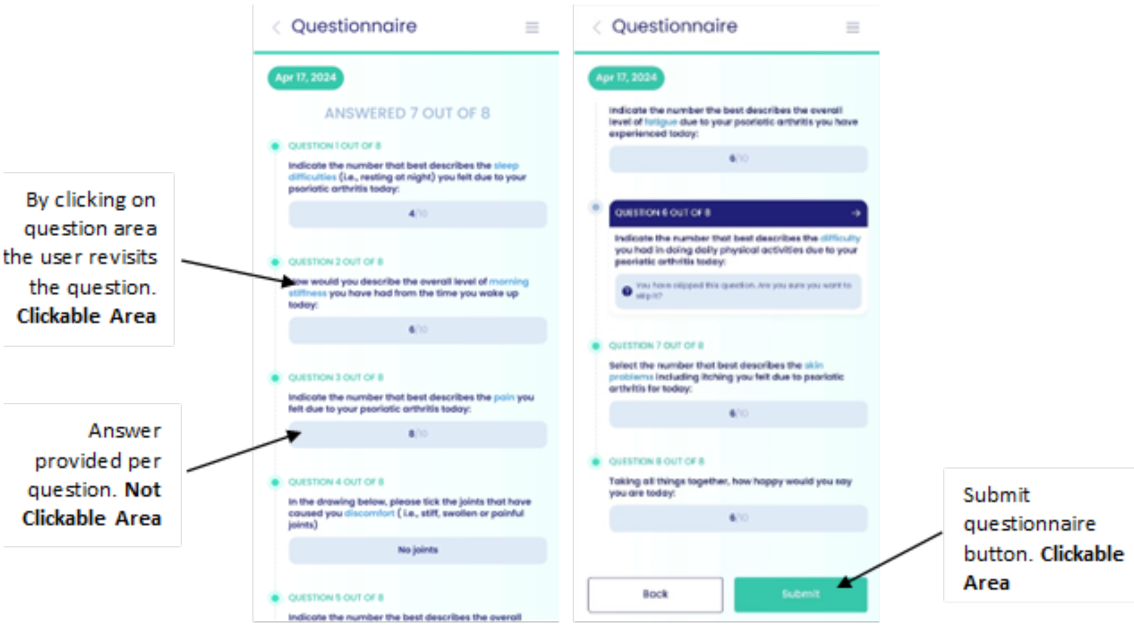

Figure 16 Questionnaire summary snapshot.

Note: The question of the flare questionnaire which is of **single-select type**, controls the navigation to the Mannequin specific question. The question specifically asks for a change from the previous day, so it only appears from the second day of switching the flare button ON and onwards, up until the flare is turned OFF.

2.2.11 Performing a photo assessment test

2.2.11.1 Introduction

This questionnaire is comprised of **two types** of questions:

(1) Single Select questions on flare, where the user may select **a single option** from the available ones (**Figure 17**).

(2) Negative to Positive Scale question on health, where the user may select **a single value** between a certain **range** (**Figure 17**).

The questions of this questionnaire are presented in **Figure 17Figure 14**.

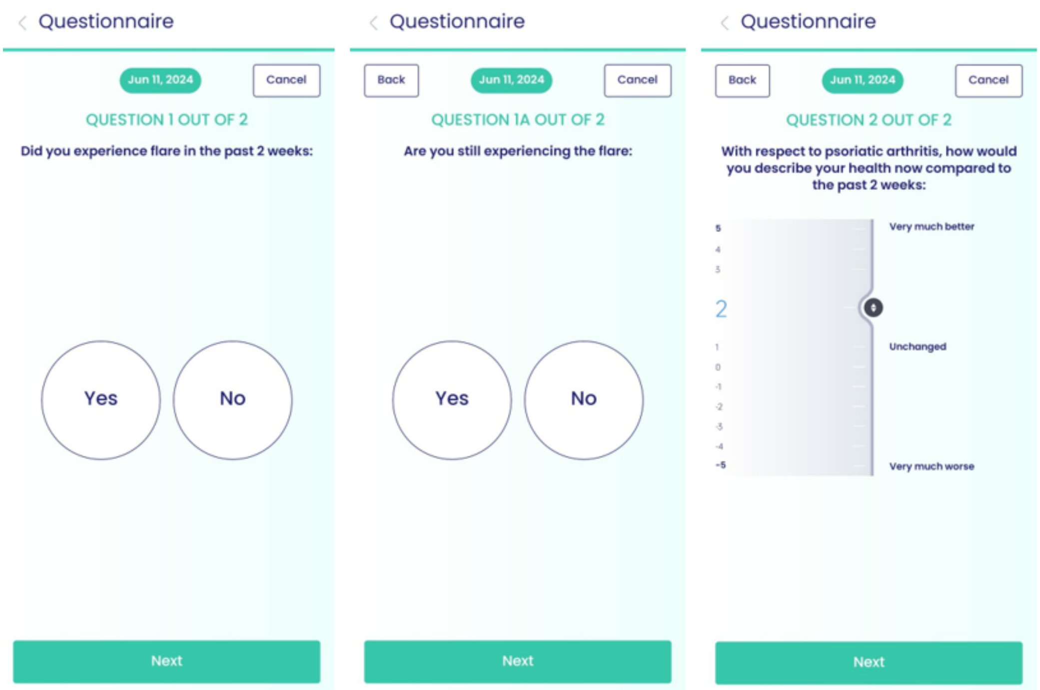

Figure 17 The 14-day flare off questionnaire.

2.2.11.2 Answering Process

Step 1: After **14 consecutive days** of turning the flare button OFF, an additional questionnaire is made available to the patients via the miPROLEPSIS application. This questionnaire is made available to identify whether there has been a change in the patient's condition.

Step 2: Depending on the user's answer in **question 1** (**Figure 17**), an additional question may appear in between to establish whether the patient is still experiencing a flare, but they have missed reporting it in the miPROLEPSIS application.

2.2.12 The miPROLEPSIS PDPID application "My Journey"

Step 1: By clicking **on the "Check" button** in the My Journey area of the home screen (**Figure 2**), the My Journey functionality opens. This is a calendar where the user can check the summaries of the data collected up to the current date (**Figure 18**).

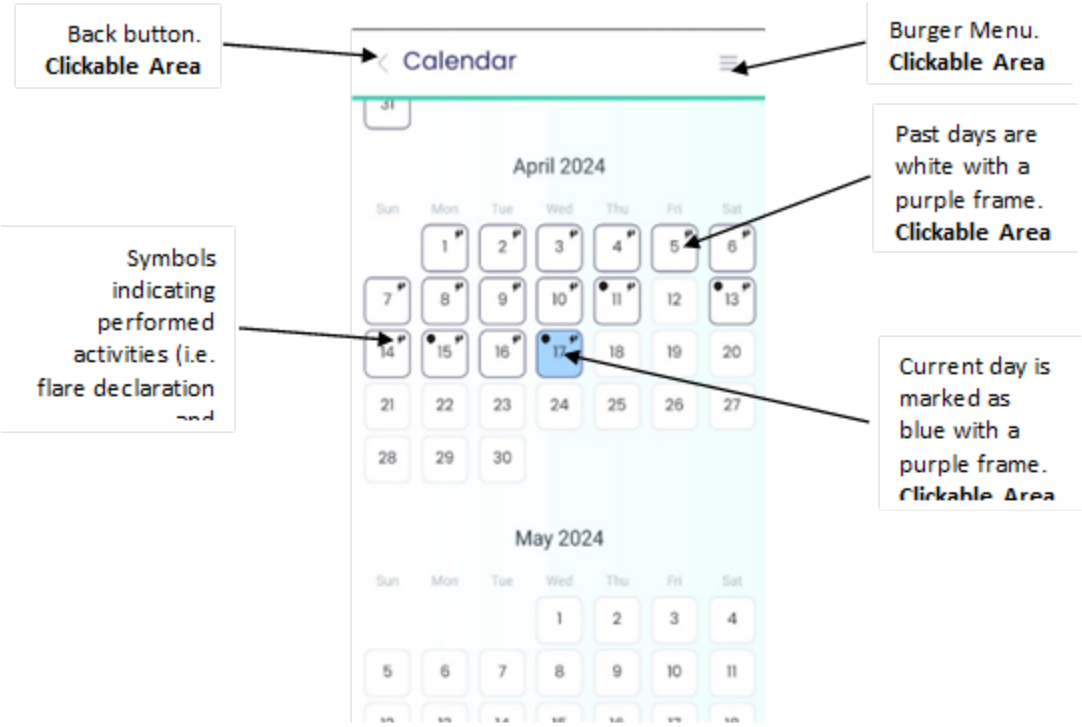

Figure 18 My Journey calendar area snapshot.

Step 2: By clicking on **a date** (up to the current date), a screen with details for the day appears (Figure 19), where the user can get a summary for the selected day as well as include a short note (user input) (Figure 19).

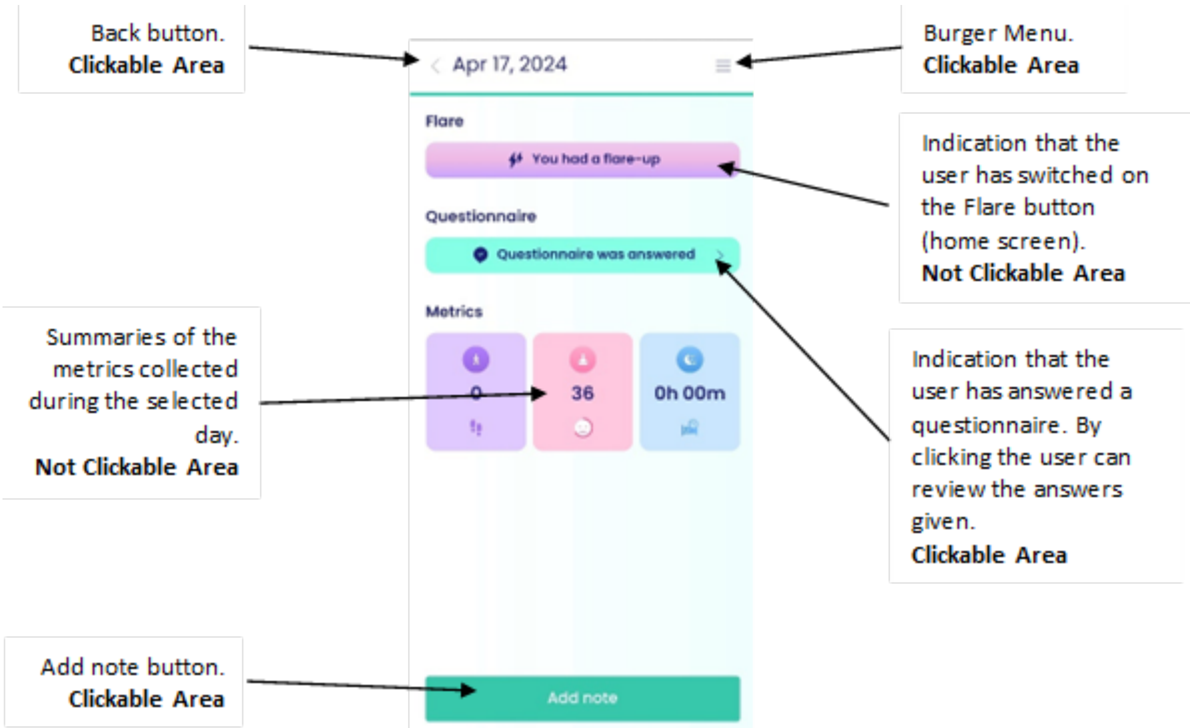

Figure 19 Daily summary of the selected day snapshot.

2.2.13 The miPROLEPSIS PDPID application notifications

2.2.13.1 miPROLEPSIS Dashboard: Notification area

At the bottom of the main miPROLEPSIS dashboard screen lies the miPROLEPSIS Notification area (**Figure 20**).

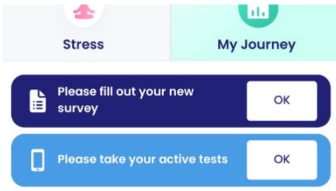

Figure 20 miPROLEPSIS notification area snapshot.

There are **two different notification ribbons** appearing in this area, informing the patient that there is an activity for them to complete:

(1) A questionnaire.

(2) A set of active tests.

Clicking on the button of each notification ribbon, the patient is navigated to the respective application screen, either to the questionnaire (**Figure 20**) or the active tests screen selection.

Depending on the availability of the questionnaires or the active tests, there can be both, one of either type or no notification ribbon at all (**Figure 21**).

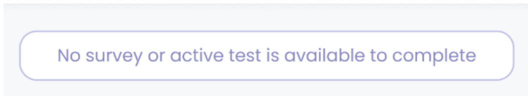

Figure 21 Notification area - Empty state snapshot.

2.2.13.2 miPROLEPSIS push notifications

The miPROLEPSIS application can receive 4 contextually different push notifications, all of which are presented in the following table.

Table 2 miPROLEPSIS push notifications.

Push Notification category	Notification Context
Daily reminder	Acts as a daily reminder for the patient to interact with the miPROLEPSIS application (sent at 12:00 am UTC)
Questionnaire available	Notification which is sent to a user when a questionnaire is available for them to fill in (sent at 10:00 am UTC)
Questionnaire reminder	Notification which is sent to a user when a questionnaire is available for them to fill in and they have not already done so for that day (sent at 10:00 am UTC)
Active tests availability	Notification which is sent to a user when they can take an active test (either photo or video based, sent at 11:00 am UTC)

Daily reminder and questionnaire related push notifications navigate the user to the main dashboard screen of the miPROLEPSIS application.

The active tests related push notification navigates the user to the test selection screen.

2.2.14 The miPROLEPSIS PDPID application "Profile"

By clicking on **Profile** option from the burger menu, the profile setting screen appears to the user (**Figure 22**).

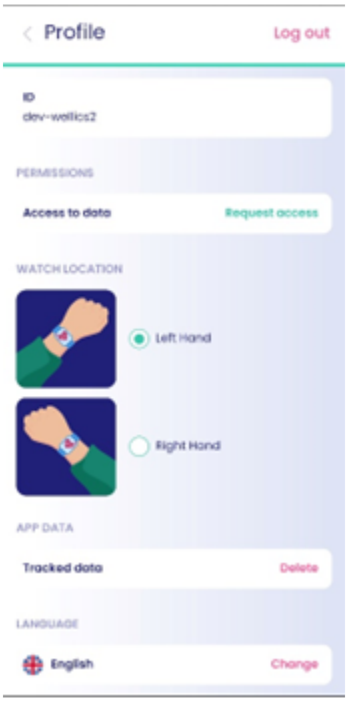

Figure 22 Profile settings screen snapshot.

Through this screen, the user can currently perform the following:

(1) Select the **wearable position** (right hand, left hand) (**Figure 23**).

(2) Select the **language** of the application (**Figure 24**). Currently, the application is available in 4 languages: (1) English (Default); (2) Greek; (3) Dutch; and (4) Portuguese.

Figure 23 Wearable hand selection.

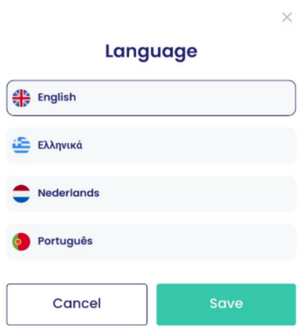

Figure 24 Language selection.

2.3 Technical information

In order to support the scope of the PDPID study application, both for Android and iOS operating systems, two application versions (one for each operating system) were generated. The figure below presents the architecture of these versions:

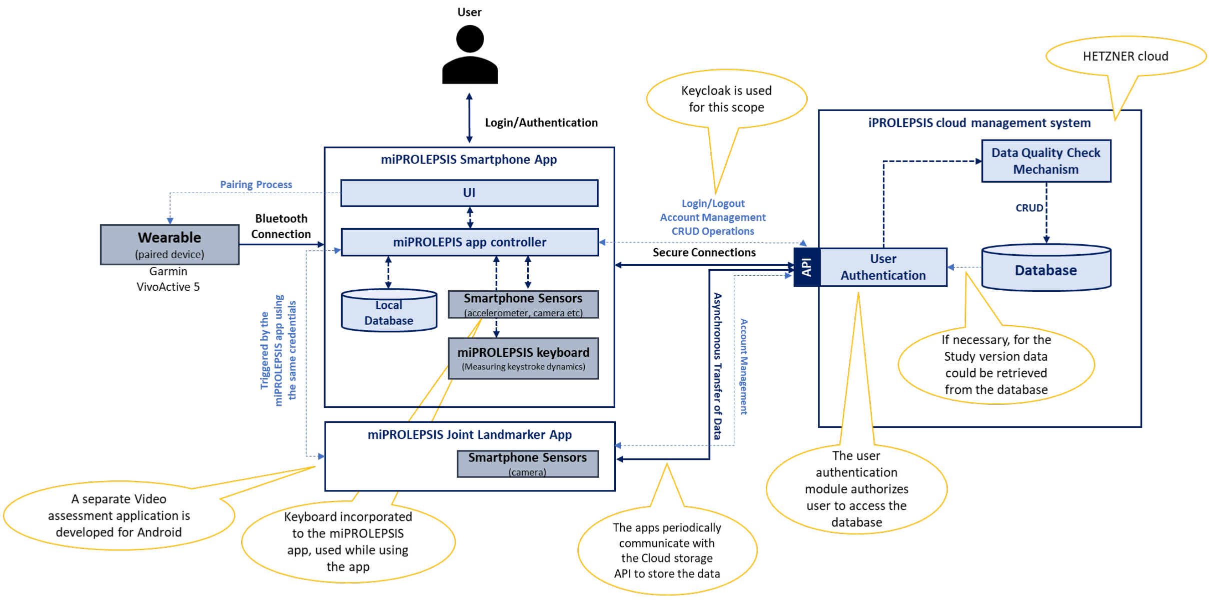

Figure 25 miPROLEPSIS PDPID study application architecture.

Please note that:

(1) The miPROLEPSIS Joint Landmarker app for **video assessment test (VMA)** feature is a **different application** triggered by the PDPID application both for the Android and iOS versions (See Section 4).

(2) The miPROLEPSIS **keyboard** for measuring keystroke dynamics, in part of the PDPID application both for the Android and iOS versions (See Section 3).

The main application features for both versions of the PDPID study architecture are:

- **User Interface (UI)**: UI is the main point of interaction between the user and the rest of the system, through which the user is capable of visually calling the different features and consuming the foreseen functionalities.
- **miPROLEPSIS app controller:** The miPROLEPSIS app controller is the central component of the application. It functions as an orchestrator of all data flows (including external and embedded sensors) within the application. It manages all connections related with this app, it handles the queries of the user (performed through the UI) and supports limited data processing, to align with the foreseen functionalities.
- **Local database:** The local database (in the smartphone) is the place where all information collected through the different data sources as well as the process results are stored before being forwarded to the cloud-based DMS.
- **APIs and other communication interfaces:** These will handle all the connections needed in order for the app to fulfil the foreseen functionalities. These involve, (1) a Bluetooth interface for connecting (and connection management) the wearable devices used in the project[3], (2) an interface for collecting environmental data, (3) an

interface enabling the user to login into the app and (4) an interface for the communication of the app with the cloud-based DMS.

Further details can be found in the deliverable **D4.1 – Initial technical specifications, architecture and product backlog** [2], where the application design and technical specifications are presented in detail.

2.4 FR, NFR and UR checklist

This section presents a checklist of FRs, NFRs and URs for the miPROLEPSIS patient application (as defined in D4.1 [2] and D2.1 [3]), indicating which are covered by the current version of the application along with a plan for the rest.

Table 3 List of FRs for miPROLEPSIS patient application and current status.

ID	Functional Requirement	Description	Implementation Status[2]	Justification
IP-59	Patient App UI and Navigation	The UI must allow the patient to easily access the features of the smartphone application.	Partially Achieved	The implementation covers the current needs of the application. The UIs will evolve further in the next version.
IP-60	Account Management	The account management must handle all related issues of the account using the application.	Partially Achieved	The implementation covers the needs of the PDPID study.
IP-61	Wearable connection establishment and management	The component must be responsible for handling the connection between the wearable and the smartphone miPROLEPSIS patient app.	Achieved	The mechanism for managing the wearable connection is fully implemented.
IP-62	Sync component with Cloud-based DMS	The sync component is responsible for data synchronization functions from the miPROLEPSIS patient app side.	Partially Achieved	The implementation covers the needs of the PDPID study.
IP-63	Scheduler	The Scheduler component should orchestrate all functions that need to be executed on a periodic basis.	Partially Achieved	The implementation covers the needs of the PDPID study.
IP-64	Storage and query handler	The storage and query handler must handle all data functions within the miPROLEPSIS patient app applications.	Partially Achieved	The implementation covers the needs of the PDPID study.
IP-65	Notification center	The Notification center must handle the notifications related with the iPROLEPSIS in the patient's smartphone.	Achieved	A notification center is fully implemented.

[2] Considering the final version of the app (iPROLEPSIS digital health ecosystem – D4.8 (M47)) compared to the current application version.

IP-66	Smartphone sensors handler	The smartphone sensor handler must handle the utilization of the smartphone's internal sensors (accelerometers, camera etc.) by the miPROLEPSIS patient app.	Achieved	The mechanism is fully implemented, and the requirement is covered.

Table 4 List of NFRs for miPROLEPSIS patient application and current status.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_UI	The application **must** support different screen sizes.	Partially Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The navigation experience and UI appearance **must** be acceptable (user friendliness).	Partially Achieved	The UIs were developed in close collaboration with the users, who provided their comments and described their needs through a series of workshops (user sessions). The current implementation covers only the needs of the PDPID study.
NFR_REL	The application **must** work smoothly.	Partially Achieved	The current implementation undergoes several user tests, where issues are being reported and included in the application roadmap. Further optimizations are foreseen in new versions.
NFR_PER	The application **must** support continuous monitoring.	Achieved	Continuous monitoring and data collection for the requested parameters is fully supported.
NFR_PER	The application **should** function in an optimized way.	Partially Achieved	A number of optimizations related with the quality and size of data, user experience and battery consumption have been implemented. Further work towards this direction is scheduled for the next PDPID application versions.
NFR_PER	The application **must** be able to perform all activities in an acceptable time frame.	Partially Achieved	The implementation covers the needs of the PDPID study.
NFR_MAIN	The application **must** be able to be updated.	Achieved	A roadmap is already in place for the new versions of the application. Towards this direction automation pipelines for the publishing of new versions in the app stores, as well as proper versioning have been created.
NFR_MAIN	The application **must** be fault tolerant.	Partially Achieved	The implementation covers the needs of the PDPID study.
NFR_POR	The application **must** run in different devices.	Achieved	The application is available in both Android and iOS devices.

NFR_SEC	The application **must** function in a secure way, resilient in different types of attack.	Achieved	SOTA technologies were applied for this scope. In order to proceed with the study, a successful check was performed by the security office of Erasmus University.

Table 5 List of URs for miPROLEPSIS patient application and current status.

UR.mP.#	Description	Implementation Status	Justification
UR.mP.01	A user needs to be able to login to the application	Achieved	This is fully implemented.
UR.mP.02	A user needs to be able to edit his/her account details	Partially Achieved	The implementation covers the needs of the PDPID study.
UR.mP.03	A user needs to be able to delete his/her data	Achieved	A mechanism for this is fully defined and implemented.
UR.mP.04	A user needs to be able to delete his/her account	Achieved	A mechanism for this is fully defined and implemented.
UR.mP.05	A user needs to agree with the terms of the application before entering app	Partially Achieved	Currently the privacy terms cover the use and scope of the PDPID application.
UR.mP.06	Privacy terms need to be available to the users	Partially Achieved	Currently the privacy terms cover the use and scope of the PDPID application.
UR.mP.07	A patient needs to be able to connect his/her wearable	Achieved	This is fully implemented.
UR.mP.08	A patient needs to be able to review the measurements monitored through the smartphone or the wearable in several lifestyle areas, such as nutrition, physical activity, sleep, mood etc	Achieved	This fully supported. The patient has the ability to review the summaries of the data through the application or request access to the raw data stored in the DMS.
UR.mP.09	A patient should be allowed to log and track medication intake, disease flares, and any external factors that may impact their symptoms, such as weather or stress levels.	Partially Achieved	The implementation covers the needs of the PDPID study. This UR will be fully covered This will be covered in miPROLEPSIS MVP (D4.6 – M34).
UR.mP.10	A patient needs to be able to receive notifications	Achieved	A mechanism that supports different types of notifications is fully implemented and deployed.
UR.mP.11	A patient needs to be able to customize the frequency and timing of reminders and alerts to match their comfort level and minimize intrusive thoughts or anxiety.	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.

UR.mP.#	Description	Implementation Status	Justification
UR.mP.12	A patient needs to have an explanation of why the app collects information like keystroke dynamics, screen activity and voice analysis	Achieved	Explanations and guidelines are provided to the patients both within the application as well as before the start of the study.
UR.mP.13	A patient needs to be able to provide feedback to the monitoring healthcare professional	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.
UR.mP.14	A patient needs to be able to answer questionnaires	Achieved	This requirement is fully addressed.
UR.mP.15	A patient needs to be able to receive recommendations	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.
UR.mP.16	A patient needs to be able to retrieve statistics on the wellness activities s/he performed in a selected period	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.
UR.mP.17	A patient needs to be able to set up her/his planner of lifestyle activities (nutrition, physical activity, sleep, and mood etc)	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.
UR.mP.18	A patient needs to be motivated and receive motivational advice	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.
UP.mP.19	A patient needs to be able to provide user generated inputs	Achieved	This requirement is fully addressed (questionnaires, provision of comments through the diary functionality).
UP.mP.20	A patient needs to be able to use the smartphone's embedded sensors	Achieved	This requirement is fully addressed (accelerometers, gyroscope, camera).
UP.mP.21	A patient needs to use the application in both Android and iOS platforms	Achieved	The application is available in both Android and iOS devices
UP.mP.22	A patient needs to use a multilingual application	Achieved	The application is available in 4 languages namely Dutch, English, Greek and Portuguese.
UP.mP.23	A patient needs to have access to a knowledge base	Achieved	A knowledge base describing the use of PDPID study application is available.
UP.mP.24	A patient should be able to have visualisations and reports summarizing trends between symptoms and activities-external factors	Not Achieved	This will be covered in miPROLEPSIS MVP (D4.6 – M34) with the assumption that this UR is still valid.

3 The miPROLEPSIS keyboard

3.1 Scope

The scope of the miPROLEPSIS virtual touchscreen keyboard encompasses the development of an intuitive and responsive keyboard designed to capture detailed timing- and position-related data of keystrokes (also referred to as key taps), the number of deletes (as a proxy of mistakes made), as well as certain key metadata (type of key feedback and a flag denoting a 'long-press') in the background during routine typing for assessing hand function and fine motor skills in PsA patients.

The keyboard stores typing data in sessions. Each session starts when the keyboard pops up and ends when it closes. Data collected for each typing session include:

Table 6 Keyboard data collected for each typing session.

Variable	Description
downtime	Timestamps corresponding to the moment keys were tapped down
uptime	Timestamps corresponding to the moment keys were released
pressure	Normalized (0-1) maximum pressure applied for each key tap
longPress	Flag denoting whether a key was long pressed
distance	Relative distance between keys tapped
downtimeDelete	Timestamps corresponding to the moment delete key was tapped down
uptimeDelete	Timestamps corresponding to the moment delete key was released
downtimeSpace	Timestamps corresponding to the moment space key was tapped down
uptimeSpace	Timestamps corresponding to the moment space key was released
numDels	Number of times the backspace/delete key was tapped
isSoundOn	Flag denoting whether sound feedback is on
isVibrationOn	Flag denoting whether vibration feedback is on
vibrationDuration	In case of vibration, its duration
keyboardScale	Scale of the keyboard (portrait or landscape)
startTime	The time the typing session started
endTime	The time the typing session ended
currentLanguage	Keyboard language

3.2 Using the miPROLEPSIS keyboard

This section presents a brief manual for using the miPROLEPSIS keyboard. Each supported function is detailed in a stepwise approach, combined with screenshots to help visualize the navigation flow. The miPROLEPSIS keyboard is designed to provide an efficient and seamless typing experience while capturing valuable typing data and metadata. Instructions are included on how to set up the keyboard, utilize the keyboard's features and customize its settings.

3.2.1 General

The miPROLEPSIS keyboard is available for Android, while an iOS version of the keyboard is planned. The two versions are different, however sharing the same requirements. For the Android version of the app, the AUTH-SPBTU custom keyboard, namely Research Keyboard, is bundled with the miPROLEPSIS application to capture timing- and position-related data of keystrokes (also referred to as key taps), as well as certain typing metadata in the background during routine typing for assessing hand function and fine motor skills in PsA patients. The keyboard is based on the latest release of the Android Open-Source Project (AOSP) Keyboard (LatinIME2). It supports several languages and typing facilitators, such as suggestions and auto-correction. The keyboard comes with a step-by-step setup process (Section 3.2.3), and its dedicated settings are located in the settings (Section 3.2.4) of the Android Operating System, usually under 'System > Languages & input'. This keyboard replaces the native keyboard of the Android smartphones. The iOS version of the miPROLEPSIS keyboard will be based on the SensorKit's SRKeyboardMetrics3 class. In this case, the device's native keyboard is used, while various metrics about the length of words, the time between keystrokes, and errors a user makes while typing on their iPhone are captured in the background.

3.2.2 Minimum requirements

Smartphone:

- **iOS:** version 14 and higher
- **Android:** version 11 (API level 30) or higher

An **account** (username and password) should be available, enabling the user to login into the application. (See Section 2.2.3).

3.2.3 Setting up the keyboard

Before using the miPROLEPSIS application for the first time after installation, the Research Keyboard is not enabled. Upon signing into the miPROLEPSIS application (Section 2.2.5), the user is prompted to enable the Research Keyboard via a pop-up dialogue. Step-by-step instructions of the keyboard setup follow along with number-matched screenshots (**Figure 26**):

Step 1: The first screen (**Welcome to Keyboard**) displays a visual representation (GIF) of the keyboard. Tap **Get started** to begin the setup process.

Step 2: This screen (**Setting up Keyboard**) provides information on enabling the keyboard. Tap **Enable in Settings** to be directed to your smartphone's keyboard settings. Note: Keyboard settings may vary by device.

Step 3: Enable the Research Keyboard by **tapping the corresponding switch** in the keyboard list. By default, the Research Keyboard includes **System Languages** as input methods. For enabling **additional languages**, refer to **Figure 27** and the corresponding instructions (Section 3.2.4).

Step 4: When enabling the keyboard, an **Attention** pop-up dialogue will appear for security reasons. Tap **OK** to proceed.

Step 5: A second pop-up dialogue will appear for security reasons. Tap **OK** to continue.

Step 6: The next screen (**Setting up Keyboard**) requires the user to switch input methods to the Research Keyboard. Tap **Switch input methods** to proceed.

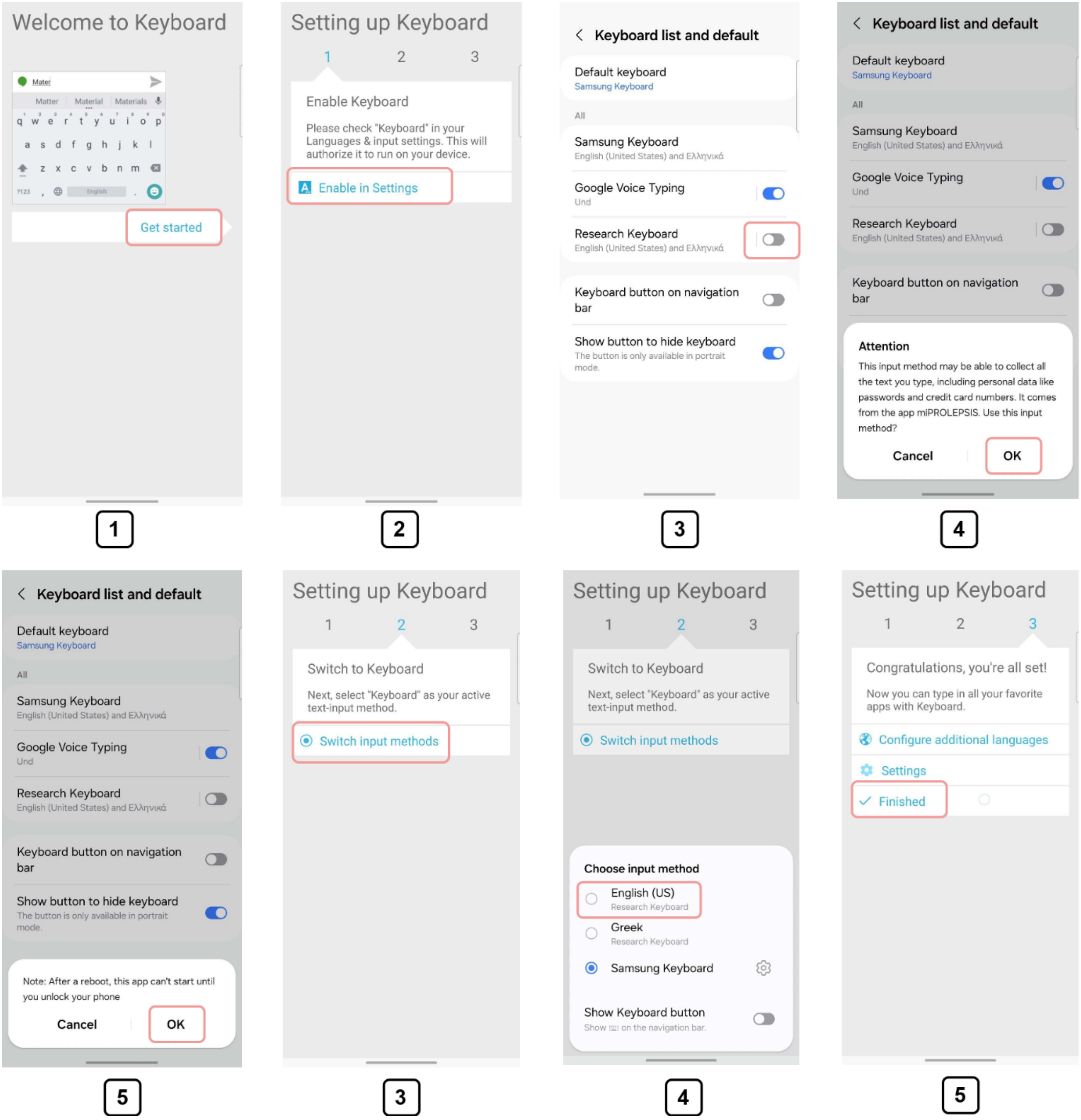

Figure 26 Research keyboard setup instructions.

Step 7: A pop-up dialogue appears (**Choose input method**), prompting the user to select the Research Keyboard as input method. **Note**: This step may differ slightly between devices. If multiple system languages are available (e.g., English (US) and Greek), select one of the **Research Keyboard options** to continue.

Step 8: This screen (**Setting up Keyboard**) confirms the successful setup of the keyboard. To customize the keyboard, tap **Configure additional languages** or **Settings** (this can be done anytime, see Section 3.2.4). Tap **Finished** to complete the setup process.

3.2.4 Modifying keyboards settings

Dedicated settings for the Research Keyboard are located in the settings of the Android Operating System, usually under **Settings > System > Languages & Input > Research Keyboard** (this path may vary by device). To use additional languages or customize the keyboard, follow these instructions, with reference to the screenshots in **Figure 27**.

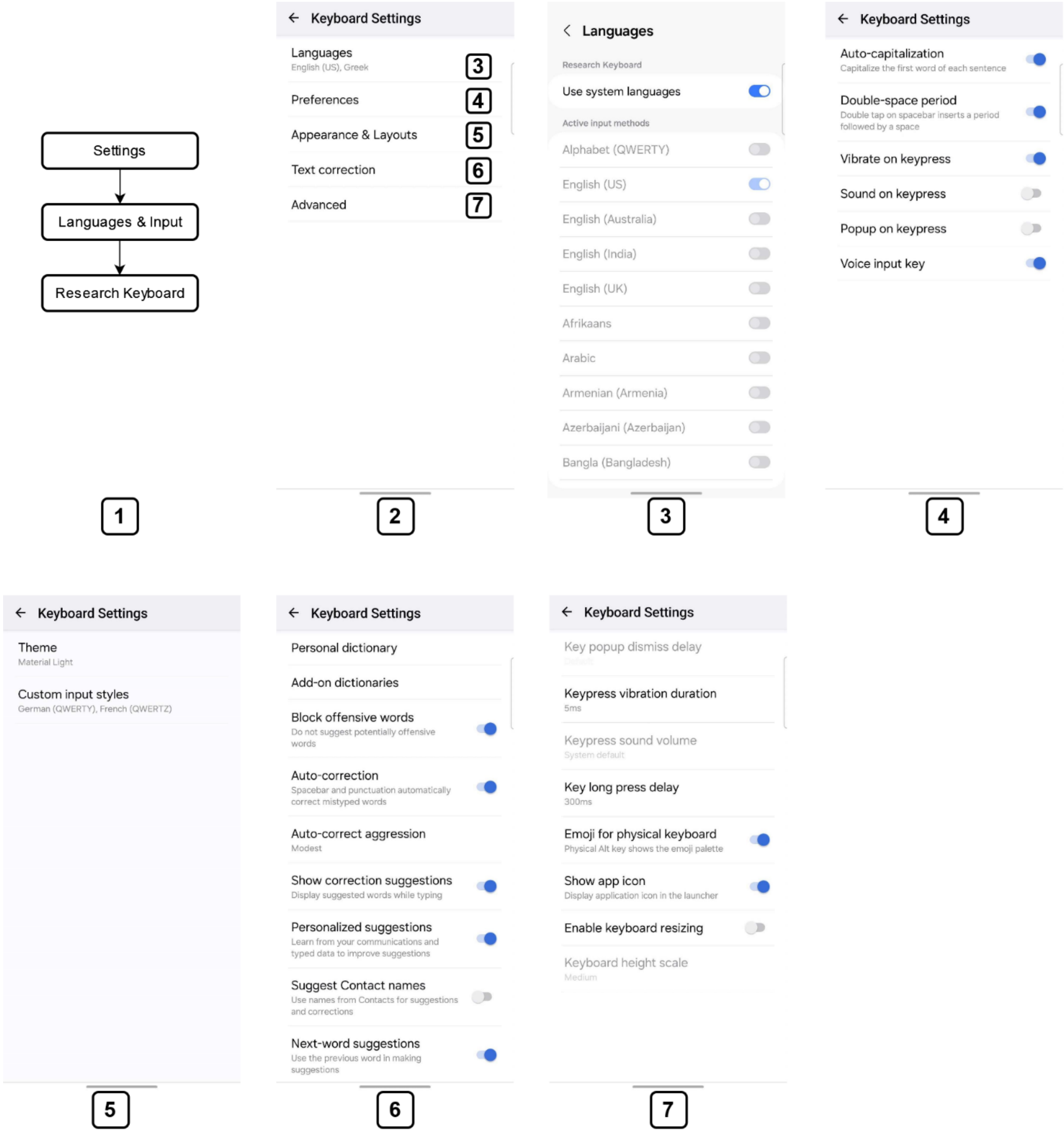

Figure 27 Research Keyboard settings (Settings > System > Languages & Input > Research Keyboard).

Step 1: Open your device's **Settings**, navigate to **Language & Input** settings, and tap Research Keyboard settings. **Note:** This path may vary by device.

Step 2: The main menu (**Keyboard Settings**) provides options for **Language**, **Preferences, Appearance & Layouts**, **Text correction**, and **Advanced settings**.

Step 3: Tap **Languages** in the main menu to access language settings. By **default**, **Use system languages** is selected. Tap the **Use system languages** switch to modify your language preferences, such as adding an additional language not included in the system defaults.

Step 4: Tap **Preferences** in the main menu to customize typing preferences, such as auto-capitalization and vibration settings.

Step 5: Tap **Appearance & Layouts** in the main menu to modify themes and input styles.

Step 6: Tap **Text correction** in the main menu to configure text correction settings, including suggestions and auto-correction options.

Step 7: Tap **Advanced** in the main menu to configure additional settings, such as vibration duration and long-press delay.

3.2.5 Using the keyboard

Refer to **Figure 28** for an example of the Research Keyboard in use within a messaging app.

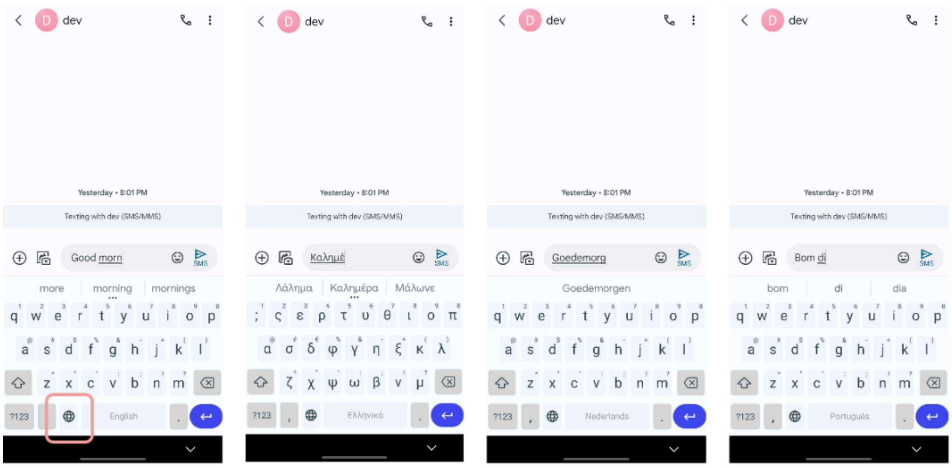

Figure 28 Screenshots of the Research Keyboard while using a messaging app in four languages: English, Greek, Dutch, and Portuguese. To switch between active languages, tap the globe key.

3.3 Technical information

The custom keyboard is based on the latest release of the Android Open-Source Project (AOSP) keyboard (Latin IME). The AOSP keyboard is licensed under Apache License 2.0. The keyboard stores typing data in sessions. Each session starts when the keyboard pops up and ends when it closes. This data is saved as a JSON string in the internal SQLite database of the keyboard, in a single table. The keyboard is developed for smartphones with Android Operating System version 11 (API level 30) or later.

The native keyboard of the iPhone devices will be used for data collection. Data collection will be based on the SensorKit SRKeyboardMetrics class, which provides the configuration of a device's keyboard and its usage patterns.

3.4 FR and NFR checklist

Table 7 List of FRs for miPROLEPSIS keyboard and current status.

ID	Functional Requirement	Description	Implementation Status	Justification
IP-174	Keyboard Layout	The keyboard must support QWERTY layout	Partially Achieved/ Achieved	The implementation covers the needs of the PDPID study.
IP-175	Keyboard Data Collection	The keyboard must support keystroke-related data and typing metadata capturing	Achieved	The implementation covers the needs of the PDPID study.

ID	Functional Requirement	Description	Implementation Status	Justification
IP-176	Keyboard Data storage	The keyboard must store collected data locally on the device	Achieved	The implementation covers the needs of the PDPID study.
IP-177	Keyboard Additional Layout	The keyboard should support other layouts (e.g., AZERTY, QWERTZ)	Achieved	The implementation covers the needs of the PDPID study.
IP-178	Keyboard Text Correction	The keyboard should provide text correction options (e.g., auto-correction, suggestions)	Achieved	The implementation covers the needs of the PDPID study.
IP-179	Keyboard User Preferences	The keyboard should provide typing preferences settings (e.g., auto-capitalization and vibration settings)	Achieved	The implementation covers the needs of the PDPID study.
IP-180	Keyboard extra UI options	The keyboard could provide appearance (e.g., themes) settings	Achieved	The implementation covers the needs of the PDPID study.

Table 8 List of NFRs for miPROLEPSIS keyboard and current status.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_UI	The keyboard **should** provide the same typing experience as the one offered by the native keyboard	Partially Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The keyboard **must** allow users to select the preferred language of the keyboard	Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The keyboard **should** provide vibration and sound feedback options to be assisted when typing	Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The keyboard **should** provide text suggestions to assist users when typing	Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The keyboard **must** provide immersive user experience (user-friendly interface with easy navigation)	Achieved	The implementation covers the needs of the PDPID study.
NFR_COMP	The keyboard **must** support a wide range of OS versions	Partially Achieved	The implementation covers the needs of the PDPID study.
NFR_UI	The keyboard **must** support multiple languages and regional settings	Achieved	The implementation covers the needs of the PDPID study.
NFR_COMP	Integration with other applications and services **must** be seamless	Achieved	The implementation covers the needs of the PDPID study.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_SEC	The keyboard **must** provide transparent privacy policy regarding data usage	Achieved	The implementation covers the needs of the PDPID study. Only timing, delete, and space data of the data are captured, not the typed characters/ text.
NFR_REL	The keyboard **must** ensure data accuracy and consistency and reliable data storage	Achieved	The implementation covers the needs of the PDPID study. A backup mechanism for the local keyboard database is planned.

4 The miPROLEPSIS Joint Landmarker app for video assessment tests

4.1 Scope

The miPROLEPSIS Joint Landmarker app is an application that accompanies the miPROLEPSIS PDPID study app and it cannot function as standalone. This means that the app is launched by the miPROLEPSIS PDPID study app whenever is needed to perform the video assessment tests and when it finishes it returns the control back to the miPROLEPSIS PDPID study app. The functionalities included in this version of the application are aligned with the project's DoA and were developed in collaboration with iPROLEPSIS medical partners. A simple user-friendly UI is implemented to allow the users to perform the tests by providing them with textual and auditory feedback.

4.2 Using the module

4.2.1 General

The miPROLEPSIS Joint Landmarker app is available for both Android and iOS systems. The two versions provide exactly the same functionalities and features. The applications are available in four (4) languages, namely Dutch, English, Greek and Portuguese. For the development of the application features, native approaches were employed, meaning that Kotlin and XCode programming languages were used for Android and iOS versions, respectively.

4.2.2 Minimum requirements

The current version of the miPROLEPSIS Joint Landmarker app has the following minimum requirements:

Smartphone:

- **iOS:** version 15.0 and higher
- **Android:** minimum SDK version 24, target SDK version 33

4.2.3 Installing the application

Step 1: The application is currently available for distribution on Firebase, but it will also be available both on Google Play and App Store.

Step 2: Upon download the installation is performed automatically (similarly to all other smartphone applications).

4.2.4 Executing the video movement tests

As stated before, the miPROLEPSIS Joint Landmarker app needs the miPROLEPSIS PDPID study app to execute. Through the miPROLEPSIS PDPID study app, the user can select to perform the movement tests. The 6 movement tests need to be performed consecutively for the entire test to be considered completed. When a movement test is to be performed, the user sees a screen of the miPROLEPSIS PDPID study app with instructions in text format, as well as in video and audio format, as shown in **Figure 29**.

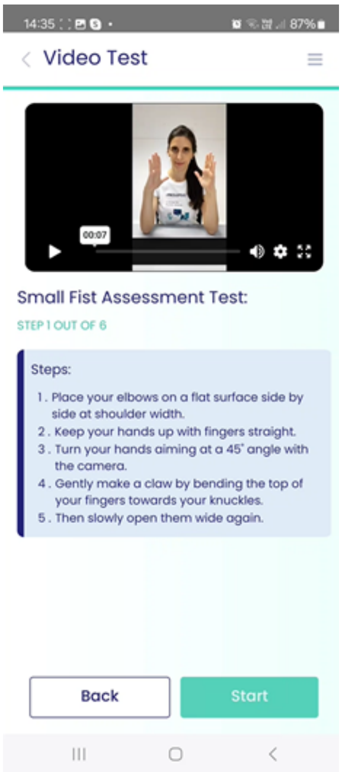

Figure 29 Video, audio and text instructions on how to perform each movement test.

When the user presses the "Start" button, the miPROLEPSIS PDPID study app calls the miPROLEPSIS Joint Landmarker app to record and process the video sequence of the movement test. When the miPROLEPSIS Joint Landmarker app launches, it displays several messages to the user through its UI to assist them (**Figure 30**). For instance, in the hand movement tests, the app asks the user to place both hands in front of the screen and when it detects the hands, a countdown of 5 seconds starts to allow the user to prepare for the test (Figure 7). When the countdown ends, a message "Recording started" appears, as well as an auditory sound is produced to alert the user.

During the execution of the movement tests, the miPROLEPSIS Joint Landmarker app visualizes the skeletal joints computed in each image frame. The positions of these skeletal joints are then gathered and sent to the miPROLEPSIS DMS. In addition, by visualizing the skeletal joints, the user can judge whether they are satisfied with their motions and the results.

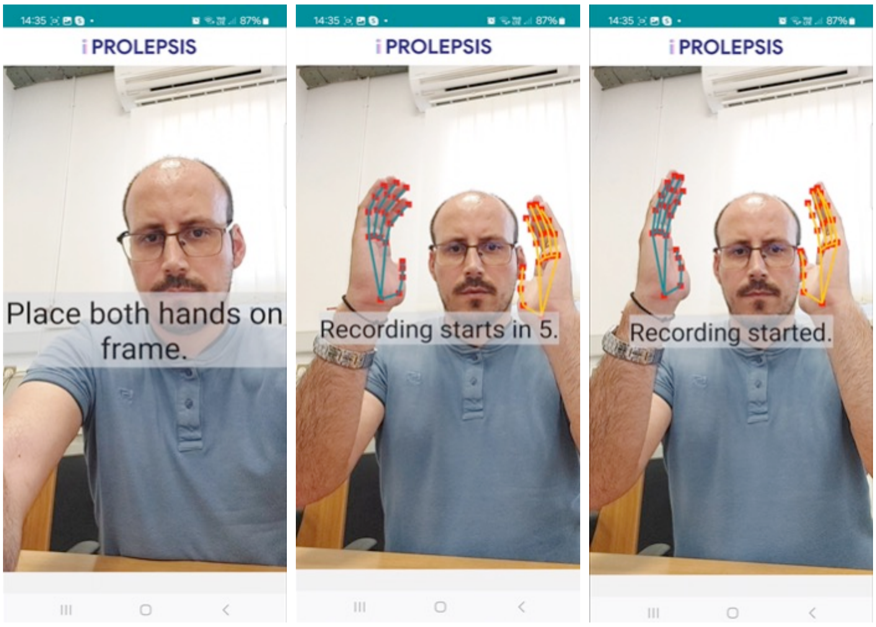

Figure 30 UI with feedback to prepare the user for the movement test.

The user is given a certain amount of time to execute each movement test. The amount of time given was defined to be adequate so that the user could perform the test without pressure. When the time finishes, a dialog box appears to notify the user whether the test was successful or not (**Figure 31**). The success of the movement test depends on whether the skeletal joints were accurately computed by the miPROLEPSIS Joint Landmarker app. As a result, missing joints are taken into consideration, with the possibility to lead to the failure of the test if they exceed a certain threshold. In the case the movement test is successful, the dialog box prompts the user to either repeat the test if they want or proceed to the next test. On the other hand, the dialog box prompts the user to repeat the test if it was unsuccessful.

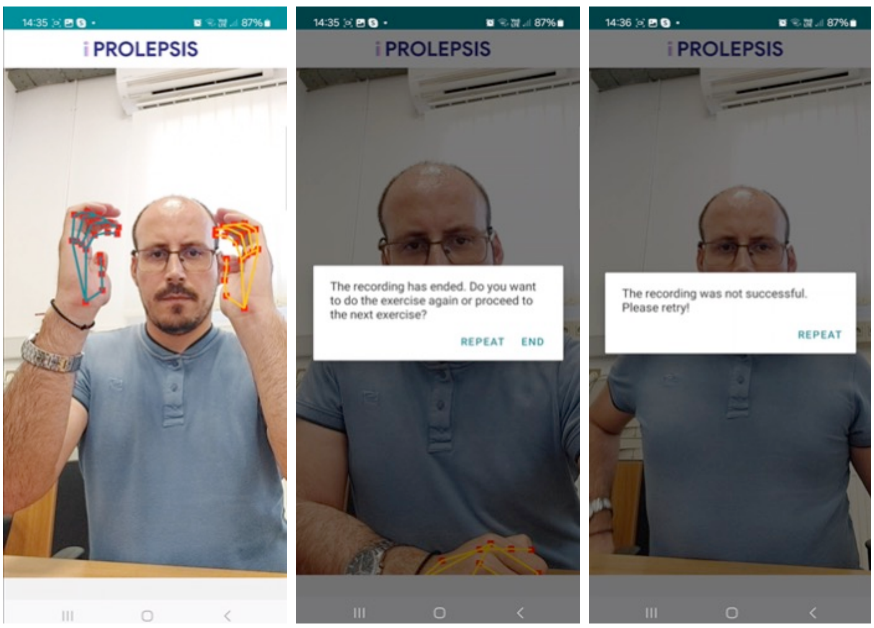

Figure 31 User executing the movement test and dialog boxes appearing at the end to notify the user about the result of the recording.

4.3 FR and NFR checklist

This section presents a checklist of the Functional Requirements (FRs) and Non-functional Requirements (NFRs) for the miPROLEPSIS Joint Landmarker app, indicating their implementation status in the current version of the application.

Table 9 List of FRs for miPROLEPSIS Joint Landmarker app and current status.

ID	Functional Requirement	Description	Implementation Status[3]	Justification
IP-118	Hand skeletal extraction algorithm from videos	An algorithm that processes video sequences and extracts skeletal data from hands must be developed	Achieved	The hand skeletal extraction algorithm is fully developed and integrated in the application.
IP-119	Body skeletal extraction algorithm from videos	An algorithm that processes video sequences and extracts skeletal data from the body must be developed	Achieved	The body skeletal extraction algorithm is fully developed and integrated in the application.
IP-132	UI for video movement tests	A user-friendly UI must be developed for the user to perform movement tests	Achieved	A simple UI that offers feedback and guidance is fully implemented.

Table 10 List of NFRs for miPROLEPSIS Joint Landmarker app and current status.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_UI	The UI **should** assist users during the exercise execution	Achieved	The app provides textual and auditory feedback, indicating to the user when the exercise starts and ends.
NFR_UI	The UI **should** allow the user to perform the exercise again if they are not satisfied	Achieved	A dialog box appears after the end of the exercise, allowing the user to perform the exercise again if they are not satisfied.
NFR_MAIN	The application **must** be fault tolerant.	Achieved	The app automatically identifies whether the hands/body of the users are visible during the exercises and prompts the users to perform the exercises again if needed.
NFR_COMP	Integration with other applications and services **must** be seamless	Achieved	The implementation covers the needs of the PDPID study.
NFR_SEC	The app **must** provide transparent privacy policy regarding data usage	Achieved	The implementation covers the needs of the PDPID study. Only 3D skeletal joints are captured, not visual features of the users.

[3] Considering the final version of the app (iPROLEPSIS digital health ecosystem – D4.8 (M47)) compared to the current application version.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_REL	The app **must** ensure data consistency and reliable data storage	Achieved	The implementation covers the needs of the PDPID study.

5 The iPROLEPSIS cloud-based data management system

5.1 Implementation details for miPROLEPSIS PDPID study application

The cloud-based Data Management System (DMS) is implemented as a set of services that facilitate all data operations of the miPROLEPSIS applications. The development of the system follows an Agile methodology, with frequent intermediate versions regularly released throughout the project's duration. To support the PDPID study application, the DMS includes the following sub-components, which are bundled and deployed as individual Docker Containers or as services on Linux OS virtual machines:

- User Authentication mechanism.
- HTTPS RESTful API, encompassing also the data quality checks mechanism.
- Data Storage solution, including a NoSQL storage cluster and an Object storage space.
- User Adherence Monitoring Service.
- Data Analytics Layer.

Figure 32 showcases the current internal architecture. The DMS components are divided into three logical groups according to their purpose:

The center of the Data Management System is the Data Storage Solution which was set up to host all incoming data collected from the users of the miPROLEPSIS application. The design of the storage solution required consideration of several important factors. Firstly, the challenge of the data volume generated: There are around 600 expected study participants for the 12 months duration of the PDPID study, with constant high-frequency data points collection, e.g. phone accelerometer data being collected at 20Hz and patient biometric data, like heart rate. It is estimated that 9GB of data will be collected from each user every month. Another consideration is related to the heterogeneity of the collected data. In addition to data collected from the smartphone and smartwatch sensors, users will also be uploading photos, and questionnaire responses, as well as typing sessions via the use of the dedicated miPROLEPSIS Keyboard. Finally, being the center of the iPROLEPSIS ecosystem of apps, the storage solution had high requirements for performance and reliability. The design must ensure that all data will be stored securely and can be retrieved easily and efficiently.

To satisfy all the requirements and above considerations, two storage spaces were set up for the DMS, a MongoDB cluster with three replica nodes, and a MinIO object storage. MongoDB is an open-source NoSQL database, optimized for big data storage and querying. The flexible schema is useful for accommodating the different types of data collected. By setting up a replica set cluster of multiple nodes, redundancy, high availability, and scalability is achieved. For the unique case of the image data, as well as for the files of the accelerometer and gyroscope data, the use of the MinIO object storage solution offers high performance, scalability, and security. A reference to the location of these files and other metadata are kept in the MongoDB cluster as an index.

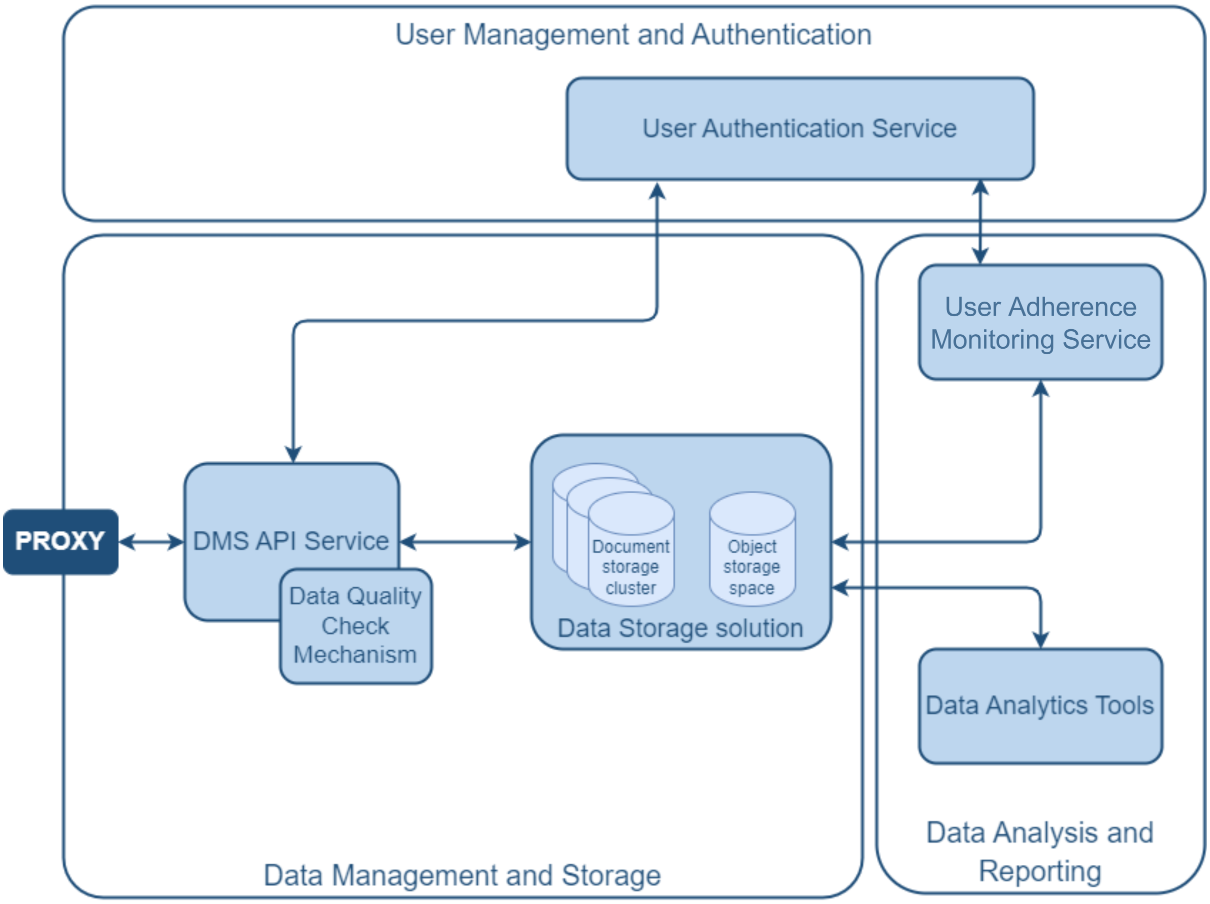

Figure 32 Cloud-based DMS internal architecture diagram.

The **DMS API Service** has been implemented as a secure and standardized way for the iPROLEPSIS apps to interact with the DMS storage spaces. This service is a REST API that acts as an abstraction layer between the client applications and the databases, allowing for controlled access to the data stored within. Input data and requests are validated according to the specified rules, integrating the data quality check mechanisms, and permissions are checked for each request with tokens acquired from the User Authentication Service. Additional endpoints for querying the data were also defined according to the client apps' needs. The documentation for the API can be found in the “docs” page which is continuously updated with each version of the API released. It is developed using Python and the FastAPI framework.

The API Service requests are routed through a **Reverse Proxy** service (NGNIX). It enhances the security of the DMS, managing the SSL certificates, but also acts as a load balancer, optimizing resource utilization by distributing traffic to the multiple deployed instances of the API, which improves performance, uptime, and scalability.

The **User Authentication Service** is an instance of the Keycloak open-source Identity and Access Management (IAS) solution set up on servers located in the IT Center of the Aristotle University of Thessaloniki (AUTH). This service implements the OAuth 2.0 protocol to provide secure Single Sign-On (SSO) access management for the cloud-based DMS and all other applications of the iPROLEPSIS ecosystem. The server has been set up and is managed by INTRA, in collaboration with AUTH as the infrastructure provider. Patients log into the miPROLEPSIS app and access the DMS for syncing and accessing their data by authenticating through this service.

Although this service was initially planned to be hosted on the INTRA-acquired Hetzner cloud, it was concluded to be hosted on AUTH servers due to data protection requirements. The service must include a database of user credentials but also collected user emails to allow iPROLEPSIS study participants the option of regaining access to their accounts and resetting their passwords without clinician intervention. Due to the sensitivity of the email addresses as personal data under GDPR, hosting on Hetzner servers was not an option.

The **Data Analytics Layer** can be described as a set of tools and services to support WP3 activities of developing ML models utilizing the data stored in the DMS. The initial design of the Data Analytics Layer includes a service (JupyterHub) where ML developers can log in and set up their isolated environment for development of their analysis, performing training on the online cloud resources using a Web-based interface. In addition, interfaces will be provided for querying and downloading data for offline (possibly more computationally heavy) workloads. Finally, a feature store can be provided for storage of results/outputs of algorithms, but the requirements of this are not yet defined, so no more details can be provided at the time of writing.

The **User Adherence Monitoring Service** is a microservice that runs daily to ensure patient data collection for the PDPID study. The service performs checks for the previous 2 days to monitor that the latest data has been synced from the patients' phones and smartwatches to the DMS. The following data types are monitored:

- Heart rate readings
- Typing sessions
- Sleep sessions

Each time the service is run an inactivity report is produced for the 48-hour period checked (2 days). The example below showcases the report structure for a single patient:

PatientId	Heart Rate	Typing Session	Sleep Session	Start Date	End Date	Date Checked
patient00001	Yes	No	Yes	5/29/2024 9:00	5/31/2024 9:00	5/31/2024 13:00

- **PatientId:** This column lists the unique identifier assigned to each patient.
- **Heart Rate:** Indicates whether the heart data (beats per minute) collected from the patient's smartwatch has been uploaded in the last 48 hours. "Yes" means the data has been uploaded, while "No" means it hasn't.
- **Typing Session:** Indicates whether the typing session data for the patient has been uploaded in the last 48 hours. "Yes" means the data has been uploaded, while "No" means it hasn't.
- **Sleep Session:** Indicates whether the sleep session data for the patient has been uploaded in the last 48 hours. "Yes" means the data has been uploaded, while "No" means it hasn't.
- **Start Date:** The beginning datetime of the 48-hour period being checked.
- **End Date:** The ending datetime of the 48-hour period being checked.
- **Date Checked:** The datetime of when the report was generated.

Developed with Java Spring, the service utilizes Spring Security and Spring Data MongoDB for security and efficient data querying. The service communicates with the User

Authentication Service (Keycloak) to retrieve the usernames/patient ids, which then allows the service to query and access patient data from the DMS.

5.2 FR and NFR checklist

This section presents a checklist of FRs, NFRs and URs for the cloud-based Data Management system (DMS) (as defined in D4.1 [2] and D2.1 [3]), indicating which are implemented by the current version of the DMS along with a plan for the rest.

Table 11 List of FRs for miPROLEPSIS DMS and current status.

ID	Functional Requirement	Description	Implementation Status	Justification
IP-46	Storage of Collected Patient Data	The cloud-based DMS must be able to provide storage for dynamic heterogeneous data originating from multiple sources (different components of the system).	Achieved	-
IP-47	Storage of Retrospective Research Data	The cloud-based DMS must be able to provide storage of retrospective research datasets.	Not Achieved	Not to be implemented in the context of the DMS.
IP-48	Unified Patient Profile	The cloud-based DMS could be able to unify all collected patient data constituting a comprehensive profile.	Not Achieved	This is scheduled later on in the project.
IP-49	REST API	The cloud-based DMS must provide a REST API to perform CRUD operations on stored data	Achieved	-
IP-50	Data Quality Checks	The cloud-based DMS must perform automated data quality checks on incoming data.	Achieved	-
IP-51	Analytics Layer	The cloud-based DMS must include a data analytics layer to support the research needs of WP3.	Partially Achieved	The implementation covers the needs of the PDPID study.
IP-52	User Management Maintainer panel	The cloud-based DMS should include a maintainer panel for management of users of the iPROLEPSIS ecosystem.	Achieved	-
IP-53	SSO Authentication	The cloud-based DMS must provide SSO across all applications of the iPROLEPSIS ecosystem for secure data exchange.	Achieved	-
IP-54	User Roles	The cloud-based DMS must support the creation of different system roles (e.g. patient, HCP) with different permissions	Achieved	-

Table 12 List of NFRs for miPROLEPSIS DMS and current status.

Category Code	Non-Functional Requirement	Implementation Status	Justification
NFR_SEC	The Authentication Mechanism SSO of the cloud-based data management system must follow standard security protocols (e.g., OAuth, SAML).	Achieved	The implementation covers the needs of the PDPID study.
NFR_SEC	The cloud-based DMS must adhere to GDPR.	Achieved	-
NFR_PER	The cloud-based DMS REST API should have low latency and respond to queries within a reasonable time frame.	Achieved	-
NFR_SEC	Access to data stored in the cloud-based DMS must be restricted based on user roles and permissions.	Achieved	-
NFR_SEC	Communications between the cloud-based DMS and other components must be performed securely over HTTPS.	Achieved	-
NFR_COMP	The cloud-based data management system's data must be compatible with the OMOP CDM where possible, to allow interoperability with other healthcare systems.	Not Achieved	This is scheduled later on in the project.
NFR_PORT	The cloud-based DMS could be implemented using open-source technologies to avoid the risk of vendor lock-ins.	Achieved	-
NFR_REL	The cloud-based DMS should have high availability.	Achieved	-

6 System orchestration, monitoring tools and integration

6.1 Implementation details and technologies

The iPROLEPSIS project employs different technologies and methodologies to ensure the efficient development, integration, and deployment of its digital health ecosystem. The implementation centers around a robust Continuous Integration and Continuous Delivery (CI/CD) pipeline, which enhances collaboration, automates processes, and ensures high-quality software delivery.

At the core of our implementation is an orchestration system that seamlessly manages the entire development and deployment lifecycle. This system provides an environment where developers can collaboratively manage source code, track issues, and conduct code reviews through an integrated version control system. The orchestration system automatically retrieves code from the repository, orchestrates the deployment process, ensuring that each partner can effectively manage their repositories while maintaining efficient collaboration and streamlined development. The repository organization allows each partner to effectively manage their code repositories, ensuring efficient collaboration and streamlined development.

In addition to this, an automation server will orchestrate the CI/CD processes, automating various stages of development, from code building and testing to deployment. By using defined pipelines, the CI/CD processes will be treated as part of the versioned application deployment, ensuring consistency, traceability and automatic deployment. Containerization is used to package software components, guaranteeing that applications run consistently across different environments by isolating dependencies within containers. A private registry stores and manages these container images, securing and scanning images for vulnerabilities to ensure only safe and compliant images are deployed. Tied to the automation server is a management interface for the container environment that offers visualization and administrative tools to handle various resources such as containers, images, and networks. A reverse proxy is used to efficiently manage incoming requests and distribute them to the appropriate services, enhancing performance and security, with this being reinforced with a system that restricts access to the infrastructure through account-based and certificate-based authentication. For unified and secure access to CI/CD services and dashboards, an identity and access management solution is employed.

The project's infrastructure is hosted on scalable virtual machines provided by a cloud service, running a Linux operating system. These VMs are configured to meet the project's requirements and can be scaled up or down as needed. Security features such as built-in firewalls, DDoS protection, and periodic backups ensure the integrity and availability of our infrastructure.

In terms of CI/CD workflows, the project utilizes two primary approaches to accommodate the diverse needs of its partners. The "source code approach" involves direct code commits to the version control system, triggering the automation server to perform automated builds, tests, and deployments. This workflow includes static code analysis, container image creation, and rigorous testing before deployment. Alternatively, the "container image approach" allows consortium partners to build and push container images directly to the private registry, with the automation server managing the subsequent automation processes, ensuring a seamless and secure integration without unnecessarily exposing source code.

6.2 CI/CD tools

Continuous Integration and Continuous Delivery/Deployment (CI/CD) is a collection of software development practices that enables technical teams to build, test and release software in a more automated and reliable way. Some of the benefits that have made CI/CD so popular, are the shorter development cycles which lead to more frequent releases, the reduced risk of defects due to the automation introduced to the different steps of the process and the early testing, and the easier management of rollbacks in case something goes wrong. Consequently, all these result to higher quality of delivered software. **Figure 33** shows how the CI/CD lifecycle is commonly presented, which emphasizes the fact that the various steps are closely interconnected and the transition from one to another is continuous.

Figure 33 CI/CD lifecycle.

Continuous Integration ensures that code changes are frequently merged into a shared repository, where automated tests are run to detect integration issues early. Continuous Delivery/Deployment automates the release process, allowing for frequent and reliable software releases. This methodology streamlines development, enhances collaboration, and ensures high-quality software delivery.

The objective of the following sub-sections is to introduce the CI/CD tools that are deployed in this framework and are utilized by all partners in their workflows. The goal of CI/CD is to streamline and standardize the delivery of the various versions of the components developed during the project, ensuring that the integrated iPROLEPSIS digital health ecosystem is effectively assembled and operational by the project's conclusion.

6.2.1 CI/CD tools

The CI/CD methodology leverages a suite of powerful tools to facilitate seamless integration and deployment. All tools are well established and widely used within the software development community. Also, most tools are open source and supported by large communities, which mitigates the risk of vendor lock-in. The main tools are:

GitHub is used as the version control system (VCS) for the IPROLEPSIS CI/CD platform. It provides a distributed Git-based version control along with an extended set of features for the development procedure, including issue tracking, code reviews, wiki, etc. Additionally, it supports extensive branching capabilities. For the iPROLEPSIS project, an organization named iprolepsis-project-eu has been created, with teams for each project partner to manage their source code, repositories, and versioning.

Jenkins is an automation server that serves as the main facilitator of the CI/CD process. This typically includes developers cloning a local copy of the remote source code repository, performing updates, and pushing changes back to the centralized repository. Jenkins is then automatically notified of incoming changes and initiates a series of actions such as pulling the latest code, building the software, running unit and integration tests, and deploying the software component for testing. Jenkins uses Pipelines to create an ordered sequence of tasks related to building, testing, packaging, deploying, and storing software. Pipelines can be defined either through the web UI or via a text file called Jenkinsfile, which is a best practice for treating the CI/CD process as part of the application, versioned like the source code.

Docker is used for the packaging of the developed software services and components into containers using de facto standards.

Harbor Registry is used to accommodate the protection of proprietary code packaged in Docker container image format, which may include sensitive data. A private container registry has been provisioned using the Harbor registry open-source solution. Developers can use Harbor registry to manage, store and distribute the produced container images.

Portainer is a free software that has as main function to manage Docker container hosts by offering a convenient UI tool for visualization. It can manage different Docker environments and offers administrators various health monitoring and management options for all Docker resources (containers, images, volumes, networks, etc.) of the developed and deployed components/services.

NGINX is used as a reverse proxy for efficiently managing requests towards the deployed services (CI/CD and/or project components). The services under the CI/CD infrastructure remain covered and are not publicly available.

PfSense is used for restricting access to the infrastructure. Each partner will be given an account and a certificate.

Keycloak is used for providing authenticated access to all CI/CD services dashboards in a unified way.

Finally, cloud-hosted Linux virtual machines (VMs) are used to host the Dockerised software applications, and the CI/CD tools presented above. The multiple VMs have been purchased and configured according to the project's needs from **Hetzner Cloud**, a public cloud provider with three data center parks in Europe. The Ubuntu Linux open-source operating system is used by all virtual machines, and their hardware configurations for CPU, memory, and disk storage depend on the purposes for which they are used. Note that these VMs can be up-scaled or down-scaled depending on the project needs and are billed monthly according to the acquired resources and usage. Hetzner provides built-in firewalls and periodic backups of the VMs, and all servers are protected by the built-in DDoS protection automated system and other security features. Access to the servers for management purposes is secured using SSH public key authentication, which ensures strong, encrypted verification and communication.

6.2.2 CI/CD workflows

Based on the currently identified needs of the iPROLEPSIS partners there are two different types of CI/CD workflows that can be used.

The **"source code approach"** is addressed to the partners that have no restrictions to use the source code management and versioning tool used in the project (GitHub). **Figure 34** depicts a high-level view of that workflow which includes the following steps:

1. The developer commits code to the project's source code repository on GitHub.
2. A webhook between GitHub and Jenkins is triggered and initiates Jenkins's build/test automation processes.
3. The code can be tested without being executed using SonarQube for static analysis.
4. A container image is created and can pass additional tests. These may include scanning the image for vulnerabilities and running unit, integration or other tests.
5. If all checks are successful, the created image is pushed to the Harbor registry.
6. The container image is pulled from the registry onto the deployment server and the respective container is instantiated and starts running.

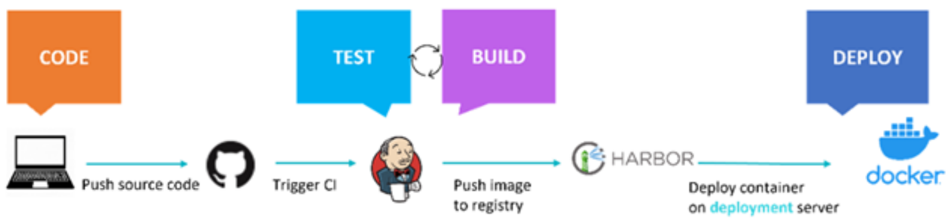

Figure 34 CI/CD workflow using the source code approach.

In cases where a partner cannot share the source code of a specific component, the **"container image approach"** is utilized. **Figure 35** presents that workflow which includes the following steps:

1. The developer builds locally (on his machine) a new container image and pushes/uploads it to the Harbor registry.

2. A webhook between Harbor and Jenkins is triggered and initiates Jenkins's automation processes. These may include scanning the container image for vulnerabilities and running unit, integration or other tests.
3. The container image is pulled from the registry on the deployment server and the respective container is instantiated and starts running.

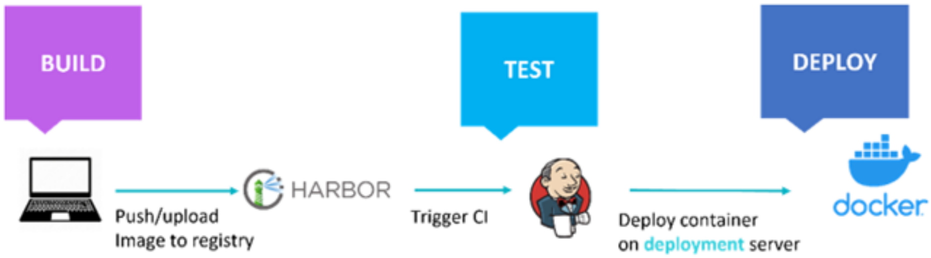

Figure 35 CI/CD workflow using the container image approach

7 Conclusions and next steps

The current report presents the project developments for covering the needs of the iPROLEPSIS PDPID study, including (1) the miPROLEPSIS PDPID smartphone application; (2) the miPROLEPSIS keyboard; (3) the miPROLEPSIS joint landmarker application for video assessment tests; (4) the cloud-based Data Management System; and (5) CI/CD tools and workflows. The deliverable is complementary to the actual prototypes prepared for this scope. The outcomes presented in this document (along with the prototypes) are the result of an intensive development and implementation process, that covered (fully or partially) the majority (>85%) of the currently defined [2] URs, NFRs and FRs.

The developments are expected to continue for the next period towards the following directions:

1. Generation of new miPROLEPSIS PDPID application versions, including new features, optimizations and bug fixes, supporting the execution of the iPROLEPSIS PDPID study. This activity will be treated as an independent and standalone branch, enabling the effective support of the study, facilitating the collection of feedback and its incorporation into the new version, maximizing in this way users' participation and overall experience.

2. Implementation of a new application (based on the current one), that includes a subset of the existing functionalities (all but the wearable support), as well as new ones (currently under discussion) for supporting the execution of the HPOS study. (M21)

3. The extension of the current application version, integrating WP4 developments scheduled for the upcoming period, towards the production of other official applications versions [1], namely: (1) Alpha Version (M23); (2) MVP Version (M34) and (3) Integrated version (M47).

For the latter leading up to each release, the process begins with the development cycle where all components are developed in all the technical tasks. Each phase includes performing unit tests, functional tests, and bilateral integrations tests. All periods will be divided into multiple sprints by the technical teams. Following this, a stable version of each component is expected to be present in the project registry. The next step is the execution of end-to-end functional and integration tests of the platform, leading to the release of version.

Table 13 Project developments and integration phases

Version	Deliverables & Milestones	Period	Phase
Study Version (PDPID study)	D4.2 (due M12)	**M09 - M18**	**1st Dev Cycle**
		M15	Components stable v1 release
		M14 - M17	End-to-end platform integration
	MS3 (due M12)	M18	iPROLEPSIS PDPID application version Release
Alpha version HCP app, Patient app, Serious Games Suite (individual games)	D4.3 & D4.4 (due M23)	**M16 - M21**	**2nd Dev Cycle**
		M21	Components stable v2 release
	MS5 (due M27)	M22-23	End-to-end platform integration
		M23	iPROLEPSIS alpha version Release
MVP version HCP app & Patient app (integrated components), Serious Games Suite (integrated suite with personalisation features)	D4.6 & D4.7 (due M34)	**M24 - M32**	**3rd Dev Cycle**
		M32	Components stable v3 release
		M33-M34	End-to-end platform integration
	MS7 (due M34)	M34	iPROLEPSIS MVP version Release
iPROLEPSIS ecosystem	D4.7 & D4.8 (due M47)	**M35-M47**	**Platform Validation & Finalization**
	MS10 (due M48)	M47	iPROLEPSIS digital ecosystem Release

Following the plan of **Table 13**, the Study Version and consequently the first development cycle is considered completed, while the second one has started. To this end, the results of this phase are going to be reported in deliverables D4.3 and D4.4 (M23), which is considered the major milestone for the next period.

References

[1] Horizon Europe 101095697 – iPROLEPSIS: Description of Action (Date: 24/11/2022)

[2] Horizon Europe 101095697 – iPROLEPSIS: D4.1 - Initial technical specifications, architecture and product backlog (Date: 31/07/2023)

[3] Horizon Europe 101095697 – iPROLEPSIS: D2.1 - Initial report on user research and co-WP2 creation process (Date: 11/09/2023)