POLITECNICO DI TORINO

Master's Degree in Computer Engineering

ARTIFICIAL INTELLIGENCE AND DATA ANALYTICS

Politecnico di Torino

Masters's Degree Thesis

Well-being App: Design and Implementation of an Interactive UI for Lifestyle Monitoring & Admin Management Tool

Supervisor
Prof. Maurizio Morisio

Candidate
Gaetano Roberto

Academic Year 2024-2025

Alla mia famiglia

A mio nonno

Abstract

Maintaining a healthy lifestyle is an increasingly challenging task in modern society. The number of people with sedentary habits and poor nutrition is always higher. To prevent health problems and monitor lifestyle, there are many cutting-edge monitoring devices and applications. The problem is precisely the heterogeneity of the different devices. The project aims to develop an app that allows users to collect, display and interpret health data in the best possible way on a single platform. This allows users to analyze their health habits and conditions and then receive recommendations for improving their well-being. The Frontend of the Well-being App, developed using Flutter, provides an intuitive but detailed user experience. It integrates the Health library to fetch and display health data (collected using wearable devices), utilizing FL_Chart for the graphical representation of the lifestyle user's data. A multilanguage system has been implemented to facilitate the application User Experience as much as possible, ensuring accessibility for a diverse user base. Furthermore, the app uses a timed notification system to encourage users to improve their health and to carry out periodic health assessment tests. Users can also improve their health knowledge through short lessons and a fun quiz system. In addition to the design and creation of the user interface, a management tool has also been implemented where administrators can manage the application parameters, for example how often to send notifications to users. This tool interfaces directly with Firebase and simplifies the management of the app's features. This thesis focuses on the design and implementation of the frontend, detailing the integration of key features and the user experience analysis. Offering a smooth and intuitive interface, the Well-being App aims to become an essential tool for users seeking to monitor, understand and improve their daily habits for a healthier life.

INDEX

CHAPTER 1.

OBJECTIVES AND THEMES: RESEARCH AND DEVELOPMENT

1.1 INTRODUCTION

The paper presented in the following pages is based on the problem of health and the analysis of people's lifestyles. After an in-depth analysis of the problem and the technological context, the proposed solution is discussed, analyzing its design and the implementation in detail. In the end, a case study is discussed to show the operation of the prototype and conclusions are drawn about the topic.

1.2 OBJECTIVE

This thesis was born with the objective of developing a mobile application that supports people to follow a healthy lifestyle. In particular, the aim of the software is to be able to collect and interpret health data from most wearable devices available on the market, to analyze users' lifestyle, give recommendations and provide help to improve it. The final application will be able to:

- read daily data measured by a wearable device,
- to analyze the data by means of graphs showing their evolution over time,
- use a timed notification system in which the user will be encouraged to carry out periodic tests on his state (health or emotional),
- present each screen in more languages to allow an easy user experience,
- have an interface for admin where it is possible to enter and modify application functional parameter,
- to teach users (who wish) some basic knowledge to improve their lifestyle, through a system of small lessons and quizzes,
- provide an assessment based on data collected in the previous week (or other timeslot) highlighting improvements or deteriorations,

- provide recommendations for improving health and lifestyle, generated by artificial intelligence (feature not presented in this thesis because assigned to another colleague).

1.3 RESEARCH AND DEVELOPMENT THEMES

During the initial phase, a detailed study was carried out on the health issue. A key document in this phase was the book “The Path to Longevity: How to reach 100 with the health and stamina of a 40-year-old” of Luigi Fontana [1]. The book presents an evidence-based lifestyle plan designed to help individuals not only extend their life span but also improve their health over a lifetime free of disease and disability. Fontana emphasizes a holistic approach to well-being, integrating key principles in various areas. These principles were the basis of the design of the app, in fact it is possible to see the 4 areas of research outlined in the book divided and each with its own importance and impact in the final assessment of a person’s health status. Once the problem context study was completed, the next step was to analyze and select the technologies to be used to implement the prototype. In particular, to collect useful users’ data, the choice fell on the use of the “Flutter” framework and the “Health” plugin that enables reading and writing health data from/to Apple Health and Google Health Connect [2].

1.4 CONTENT ORGANIZATION

This work is divided into four chapters. The first chapter serves as an introduction to the paper and cites the objectives of its drafting, the research and development topics covered, as well as the organization of the whole work. The second chapter deals with the technological context, the problem faced and a general view of the state of the art. The third chapter illustrates the design and implementation of the prototype, in particular cites functional and non-functional requirements of the project, it graphically shows a view of the logical process of the notification system’s algorithm, architecture and database used, explains the technologies employed and details the implementation illustrating and commenting portions of the code. In the last chapter, the operation of the prototype is

discussed by examining the data expected from the prototype and describing the application scenario through two case studies, one for the generic user and one for the administrator. Finally, in the last paragraph of the chapter, the conclusions reached and possible future developments relating to the prototype are presented.

CHAPTER 2.

CONTEXT, PROBLEM, MODELS AND ENABLING TECHNOLOGIES

2.1 INTRODUCTION

This chapter will describe the technological context and problem addressed, finally presents the state of the art of the prototype.

2.2 THE CONTEXT

In recent years, the concept of well-being has taken on a central role in society, prompting more and more people to seek solutions to improve their physical and mental health. In the last centuries, technology has never ceased to evolve, facilitating its integration into the health sector. This has led to the development of numerous software tools that allow lifestyle monitoring, allowing users to track key parameters such as diet, physical activity, sleep quality and stress management. These are the 4 areas on which the study of lifestyle improvement in the book “The Path to Longevity: How to reach 100 with the health and stamina of a 40-year-old” of Luigi Fontana [1] focuses. The book presents an evidence-based lifestyle plan designed to help individuals not only extend their life span but also improve their health over a lifetime free of disease and disability. Fontana emphasizes a holistic approach to well-being, integrating key principles in various areas:

- Nutrition and diet: supporting a balanced diet rich in plant-based foods, whole grains and healthy fats, minimizing processed foods and excessive sugar intake.
- Physical activity: encourages regular exercise routines that combine aerobic activities with strength training to maintain muscle mass, cardiovascular health and overall vitality.
- Cognitive health: highlighting the importance of mental stimulation through continuous learning, participation in hobbies and social interactions to preserve and improve brain function.

• Social connections: emphasizes the role of meaningful relationships and community involvement in promoting emotional well-being and longevity.

By adopting these interconnected strategies, Fontana suggests that individuals can significantly reduce the risk of chronic diseases, maintain physical and mental vigor, and lead more fulfilling lives well into advanced age. This holistic lifestyle plan aligns with findings from various longevity research studies, including those observing centenarian populations in "Blue Zones" around the world. These communities share common practices such as plant-based diets, regular physical activity, strong social ties, and purposeful living, all contributing to their remarkable health and longevity. [3] Incorporating these principles into everyday life can serve as a practical plan for those who aim to improve their well-being and achieve a longer, healthier life. Indeed, these are precisely the principles that have formed the basis in the design and implementation phase of the mobile application. In this context, there is a need to develop interactive user interfaces and advanced administration tools that improve the user experience and optimize data management. The present thesis is placed in this scenario, proposing the design and implementation of a wellness app with an intuitive interface and an effective management panel, able to support users in monitoring their lifestyle and provide administrators with advanced data management tools.

2.3 THE PROBLEM

One of the main problems on the health issue is the lack of awareness about one's own health. Many people do not worry until health problems emerge, but at that time it is too late. Instead, they could monitor their lifestyle and eliminate harmful habits to prevent many known diseases. The World Health Organization (WHO) defines the prevention as: "approaches and activities aimed at reducing the likelihood that a disease or disorder will affect an individual, interrupting or slowing the progress of the disorder or reducing disability" [4]. These activities do not require special efforts, it would be enough to monitor your own condition with periodic visits to make sure you are well and integrate healthy habits into your lifestyle, instead eliminating harmful ones.

Figure 1: Risk factors for chronic diseases

Figure 1 shows the main risk factors for chronic diseases according to the Prevention Centre. Many studies associate these characteristics in people's lives with the occurrence of certain diseases. For example, from the research conducted by the University of Rochester Medical Center Rochester (New York), the first entries in the top 10 most common health issues concern physical activity, nutrition and obesity [5]. One of the primary objectives of the application is precisely to monitor and improve these aspects in users' lives. The second aspect to consider is the difficulty in tracking and interpreting well-being data. Users often collect data from different sources (smartwatches, fitness apps, diets) but do not have a unified tool to analyze it and draw useful conclusions. Even focusing on smartwatches only, we find a problem of heterogeneity.

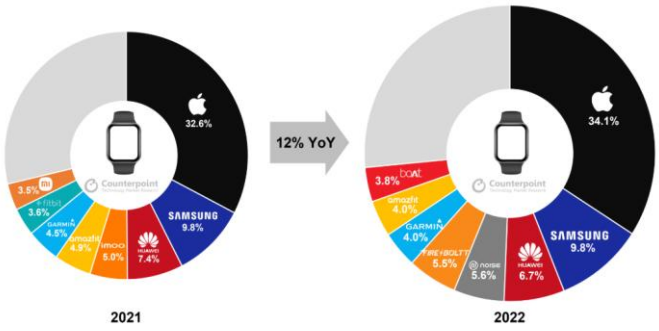


Figure 2: Market analysis Smartwatches Brands 2021 -2022

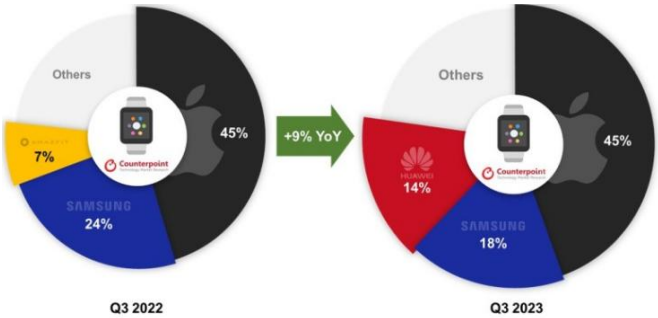


Figure 3: Market analysis Smartwatches Brands 2022 -2023

Looking at Figure 2 and Figure 3 it is possible to notice that the smartwatches market is constantly evolving and consists of many different devices. The problem is that each of them makes different measurements (depending on the sensors installed on the device), analyzes some metrics rather than others and uses its own application to show data over time. The aim of this thesis is to collect data from any device, analyze them and draw conclusions from them. In addition, thanks to the integration of Artificial Intelligence, it will be possible to provide personalized indications based on the data read to improve the lifestyle and well-being of users. These recommendations will therefore not be generic but will take into account the specific needs of each user. Another problem that this thesis aims to solve is the lack of continuity in following a path of constant well-being. Many people start with good intentions to improve their habits of life but end up giving up after a few weeks or months, not seeing tangible results. Therefore, one of the objectives is to introduce a system of gamification and adaptive notifications to maintain this consistency over time.

2.4 OVERVIEW OF THE STATE OF THE ART

In recent years, the rapid progress of technologies has transformed the way people monitor and manage their well-being. Mobile applications and wearables devices have played a crucial role in providing users with real-time information about their life habits, including physical activity, nutrition, sleep and stress levels. Despite the growing popularity of these solutions, several challenges remain, such as difficulties in integrating data from different sources, low user engagement over time and a lack of knowledge of essential health issues. This paragraph analyzes the state of the art by highlighting the strengths and weaknesses of existing wellness applications on the market. The first case considered is Yazio. In particular, it is concerned with helping users to improve their diet and therefore deals with one of the 4 macro areas mentioned above (nutrition). The main feature of this application is the insertion of the meals that the user eats. The software automatically calculates the calories and macronutrients of each meal. It is possible to enter this data manually or by scanning the bar code on the packaging of the various

foods. The calories consumed each day are compared with the user's caloric requirements, calculated based on the data and goals provided by the user during registration. (Figure 5)

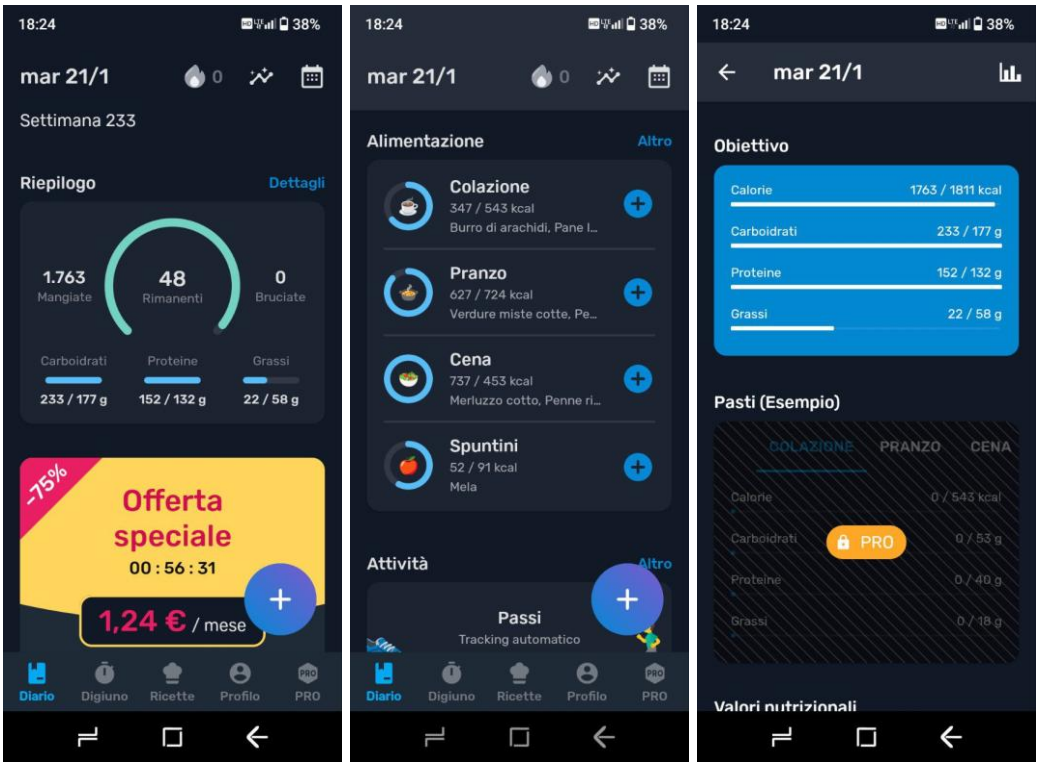


Figure 4: Yazio - Daily Calories and Macronutrients

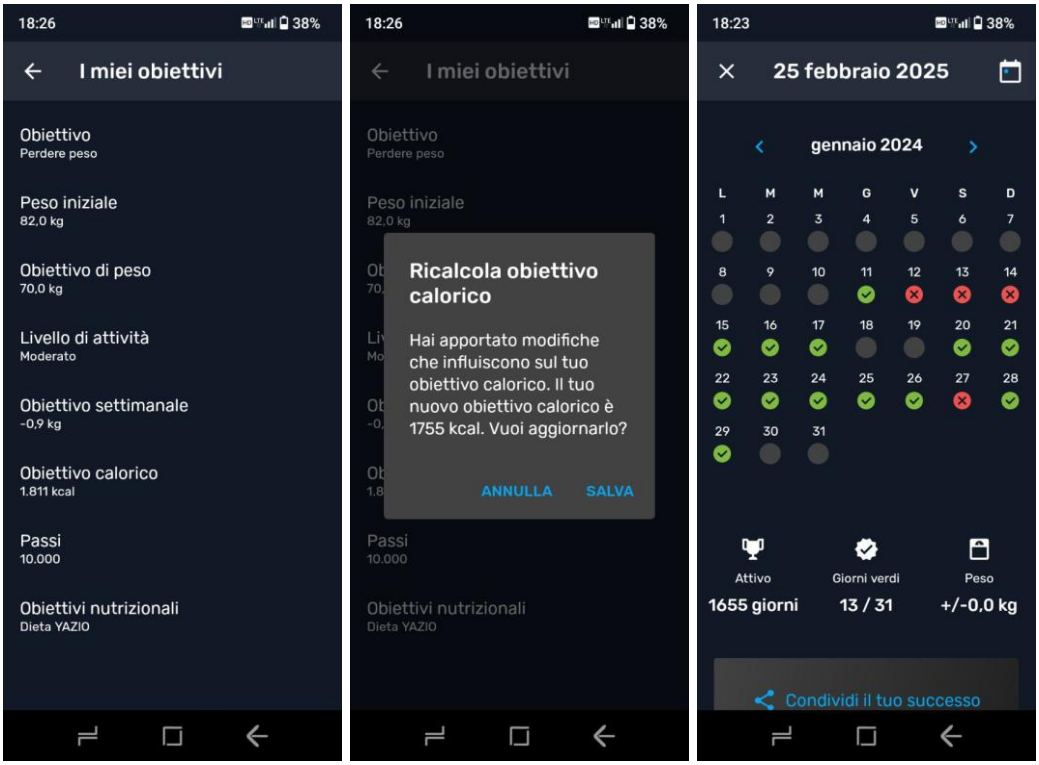


Figure 5: Yazio - Caloric requirements and Goals

Strengths to mention are a user-friendly interface, the ability to sync with wearable devices via Google Fit and other similar applications, the use of notifications and goals to motivate the user to enter meals. While the main weaknesses are the lack of a professional figure (such as nutritionist) in the user's diet, features unlockable only with the premium version and a database not always accurate. Another application that allows users to analyze the data collected from wearable devices is Google Fit. It allows them to examine in detail the data related to 3 of the key areas for good health: Activity, Nutrition and Sleep (Figure 6Figure 6). It also indirectly allows users to investigate the level of stress of users through heart rate and sleeping time measurements. (Figure 7)

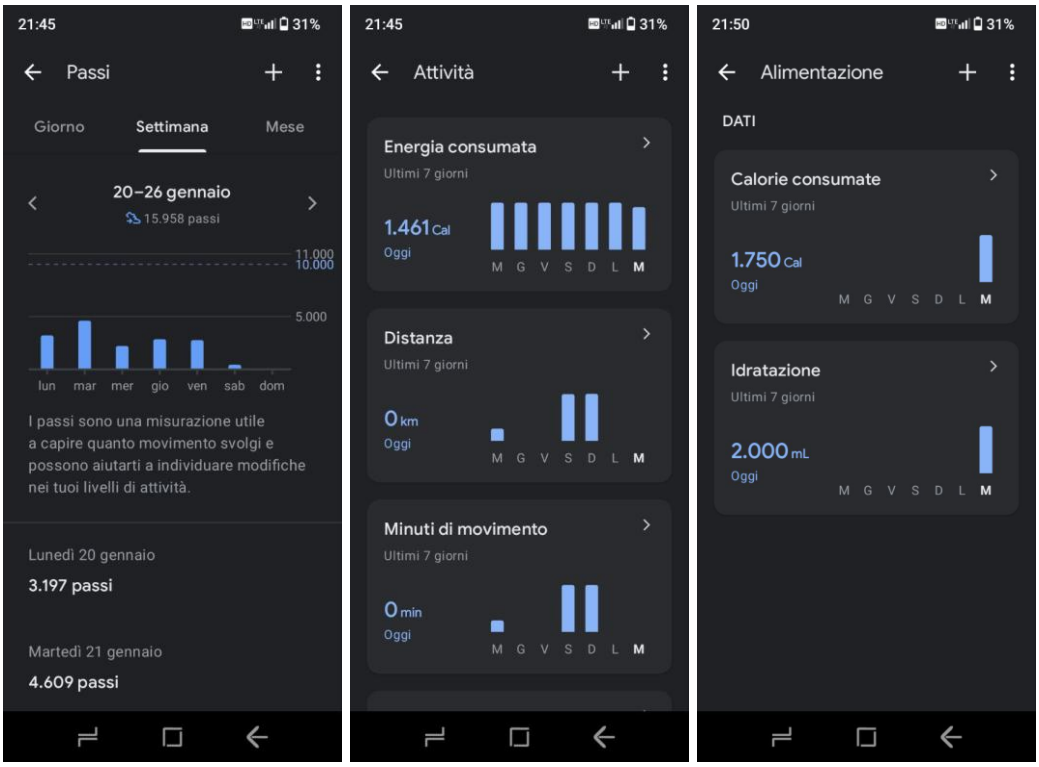


Figure 6: Google Fit - Activity and Nutrition

The strengths are the ability to deepen the different areas of health (each with its own measurements over time), customization for each user thanks to settable goals, but the strong point is definitely the integration with numerous apps including Yazio, FatSecret (similar to Yazio), Fitbit, GloryFit (app to connect smartwatches) and many more. Among the weak points, however, we must mention the very detailed interface but less understandable than that analyzed before and, in addition, there is no real involvement of users, in fact they can only view the data collected by the various connected devices.

Having examined the state of the art, the prototype resulting from this thesis work will try to combine the advantages of the different software already present, trying to fill the gaps revealed by them.

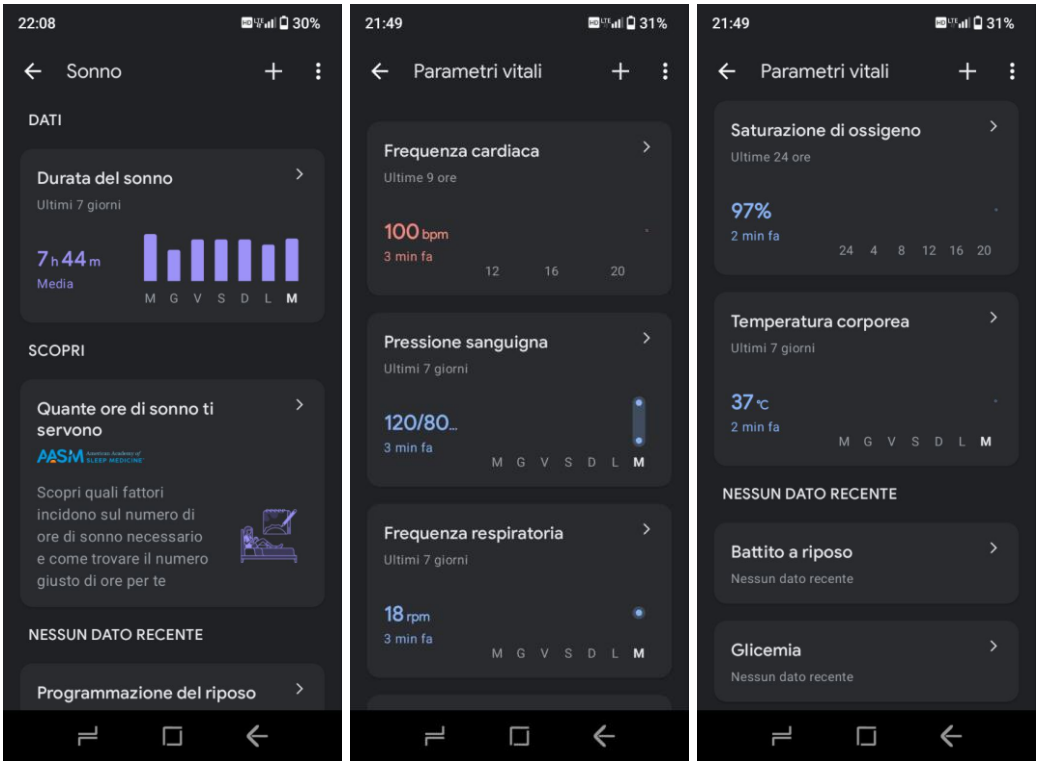


Figure 7: Google Fit - Sleep and Health Measures

CHAPTER 3.

PROTOTYPE DESIGN AND IMPLEMENTATION

3.1 INTRODUCTION

This chapter aims to describe the design and the implementation of the prototype. In particular, will be listed the functional and non-functional requirements of the same, a diagram of the main scheduling process, an architectural view of the prototype and collections created in the database. In addition, enabling technologies and software implementation will be discussed.

3.2 THE REQUIREMENTS

3.2.1 THE AIMS OF THE PROTOTYPE SYSTEM

The prototype is designed to have a different behavior for registered and first-time users. An initial authentication phase has been designed for this purpose, whereby one interface will be shown instead of another. Authentication can be done via Google or using Email and Password. If the user accesses the app for the first time, an onboarding screen appears. In this screen there is an introduction to the purpose of the app and some questions for the user (demographic, techniques and body measurements). After the onboarding phase, the Home screen appears, which is the first screen that also sees the authenticated user who does not access for the first time. On the first launch, the application connects to Google Fit and fetches all the data it needs to work (the data authorized by the user). The user can then move as he wishes: examine the data collected over time relating to Activity, Nutrition, Sleep, Mood and other health measures or, in addition, other sections of the app that allow to chat with the coach (AI that provides assessments and recommendations to the user - not implemented in this prototype), set personal goals to achieve (such as goal steps, maximum daily calories to consume and sleep duration) or learn more about health through the Learn section, provided with lessons and quizzes on different health topics.

3.2.2 USER REQUIREMENTS

Functional requirements

- FR01. Registration - The unregistered user can register.
- FR02. Login - The registered user can log in using email and password or Google account.
- FR03. Logout - The user can log out of the app
- FR04. User profile - The user can view his account information.
- FR05. Password Recovery - The user can reset his password.
- FR06. Modify Account – The user can modify his account data.
- FR07. Delete Account – The user can delete his account.
- FR08. Activity data - The user can see fetched activity data over time.
- FR09. Nutrition Data View - User can see nutrition data over time.
- FR10. Nutrition Data Add - User can add nutrition data.
- FR11. Nutrition Data Delete - User can delete nutrition data.
- FR12. Sleep data - The user can see fetched sleep data over time.
- FR13. Mood data - The user can see mood data over time
- FR14. Mood Data Add - User can add mood data.
- FR15. Time Navigation - User can see data by day, by week or by month.
- FR16. Oxygen data - The user can see fetched oxygen data over time.
- FR17. Heart rate data - The user can see fetched heart rate data over time.
- FR18. Body Balance data View - The user can see body balance data over time.
- FR19. Body Balance data Add - The user can add balance time on left leg, right leg, tandem.
- FR20. Body Strength data View - The user can see body strength data over time.
- FR21. Body Strength data Add - The user can add grip strength, number of push-ups, squats, abdominals.
- FR22. Assessment - The user can see the assessment generated by the system, compared with average values for his age, gender and conditions.

- FR23. Goals - The user can set goals.
- FR24. Lessons View - The user can see different lessons on health.
- FR25. Complete Lesson - The user can complete lessons that he read.
- FR26. Quiz - The user can test his knowledge by taking specific tests for each lesson completed.
- FR27. Data Fetch – The system automatically recovers data from various third-party apps.
- FR28. Multilanguage - The user can easily switch the language that he prefers.
- FR29. Adaptive Notification System – The system sends timed notifications to the user to encourage him to take tests on his body strength, balance and insert food consumed. The system is adaptive because the time that occurs between one notification and the other is suitable for each user.

Non-functional requirements

- NFR01. Usability - The user interface must be intuitive and user-friendly, easy to use for first-time users.
- NFR02. Performance - The app should have response times to user inputs that do not exceed 500ms.
- NFR03. Performance - The app should load each page within 2 seconds.
- NFR04. Security - The system must be protected from unauthorized access: the user must log in to access authorized data.
- NFR05. Compatibility - The app must work properly on devices with different screen sizes (smartphones, tablets).
- NFR06. Extensibility - The system should be easily extensible to run on both Android and iOS devices.
- NFR07. Portability - The system must work on android devices.
- NFR08. Reliability - The system should run properly 99% of the time.

- NFR09. Interoperability - The system must work smoothly with third-party services like Google Fit and Health Connect, ensuring consistency between the data.
- NFR10. Maintainability - The code must be structured in a modular way and documented to facilitate maintenance and future updates.

3.3 MAIN PROCESS DIAGRAM

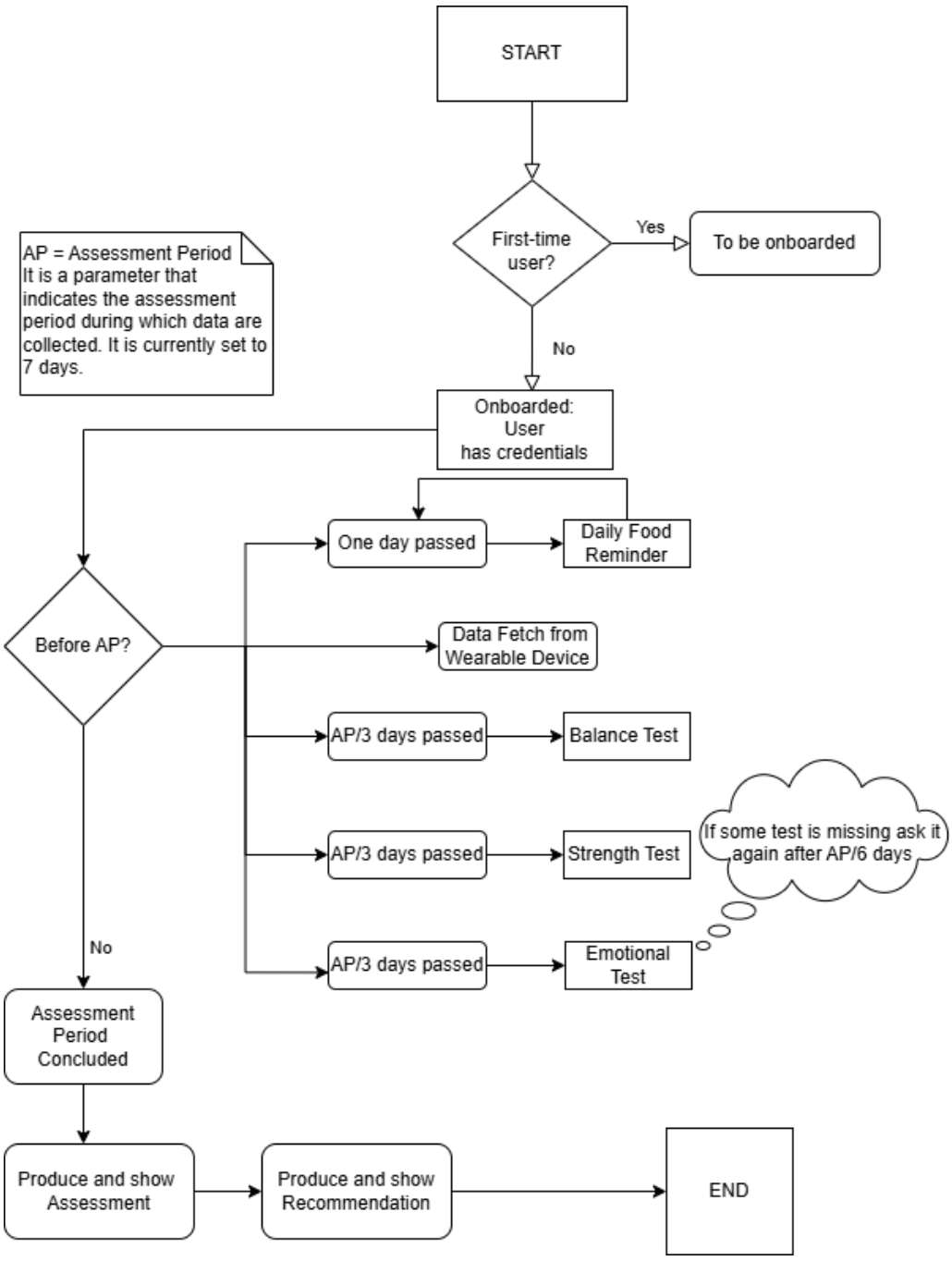


Figure 8: Flow Chart of the Assessment

Figure 8 shows graphically the logical process leading to the generation of the user status assessment. When the app is launched, there is an onboarding phase for first-time users where specific user data such as age or weight are requested so that the system has a general overview of their status. If the user already has credentials to log in (is therefore onboarded) there is an assessment period. This phase has a fixed duration that can be changed by the admin in the dedicated interface and is currently set to 7 days. In this period days notifications are sent depending on the value of assessment period. In particular, the daily Food reminder is sent daily, while data are continuously retrieved from wearable devices through Health Connect. Instead, the functioning of the other 3 notifications is the same. Taking for example an AP value of 7 days, a notification to perform the balance test is sent after 2 days (AP/3) from the first start of the app or from the last assessment. If the user does not enter the test data, then the same notification is sent again after 1 day (AP/6). It only goes forward after having entered the data or having ignored the notification for 3 consecutive times. In the next phases the same thing happens for the strength test and the emotional life test. Once the assessment period is concluded, an evaluation of the user state is made using values obtained from Health Connect for heart rate, blood pressure, steps, sleep, etc. and the values entered by the user related to food, strength, balance and emotional state. The assessment contains values computed for each category that are then compared with estimates values of the average user that vary based on gender and age. The user can then see the assessment and the related recommendations produced by the Coach (LLM).

3.4 GENERAL ARCHITECTURAL VIEW

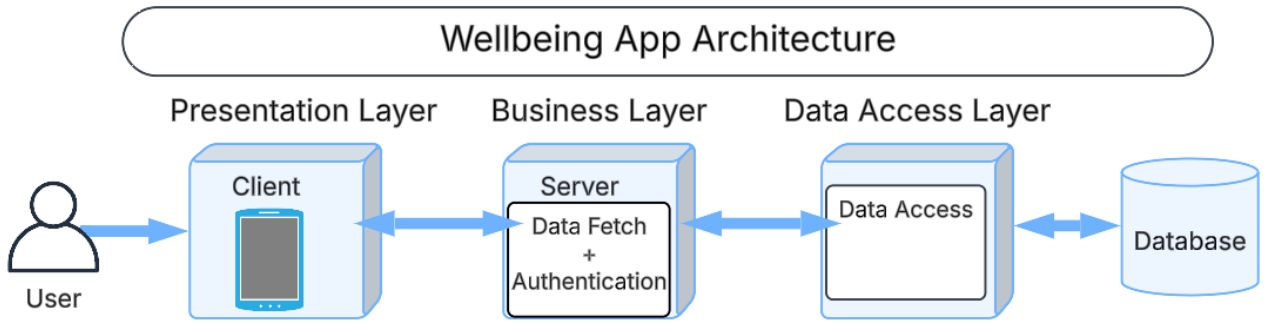


Figure 9: Wellbeing App Architectural View

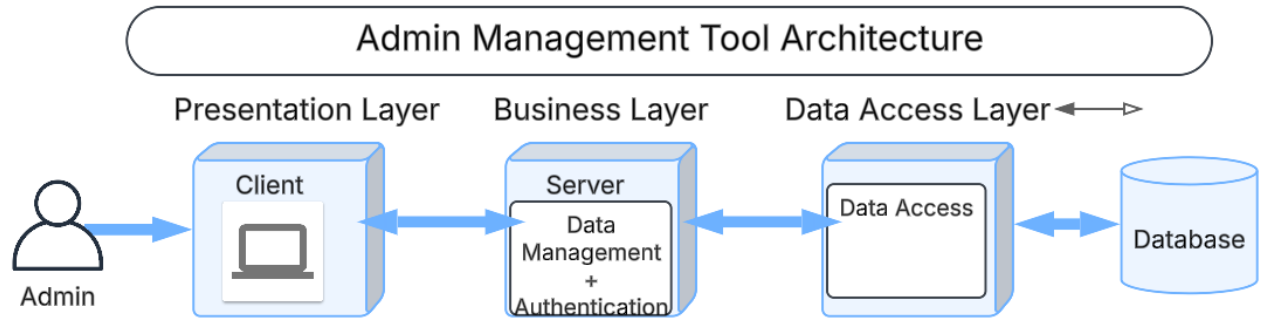


Figure 10: Admin Management Tool Architectural View

The architecture presented are both three-tier architectures [6]. The application layers are divided into three logical and physical tiers:

- Presentation Tier: it shows the interface to the user where he reacts with the application. This tier is usually built using JavaScript, HTML and CSS and can be run on a generic web browser if the GUI is a web application (Admin Management Tool Client) or using different languages such as Flutter, Kotlin and React Native if it is a mobile application (Wellbeing App).
- Business Tier: it contains the business logic of the system and is indeed the heart of the application. It uses business logic to query and manipulate data in the Data Access tier. It also communicates with the presentation tier by receiving requests from it and forwarding the corresponding responses.
- Data Access Tier: this level is responsible for data management and storage. Includes databases or other data management systems that store information used by the application.

The three-tier separation offers several benefits, including increased application scalability, maintainability and security. For example, a failure in one of the levels is less likely to affect the others, improving the overall reliability of the system. [7]

3.5 DATABASE DESIGN

In the design phase of the solution, the database chosen to be used is Cloud Firestore.

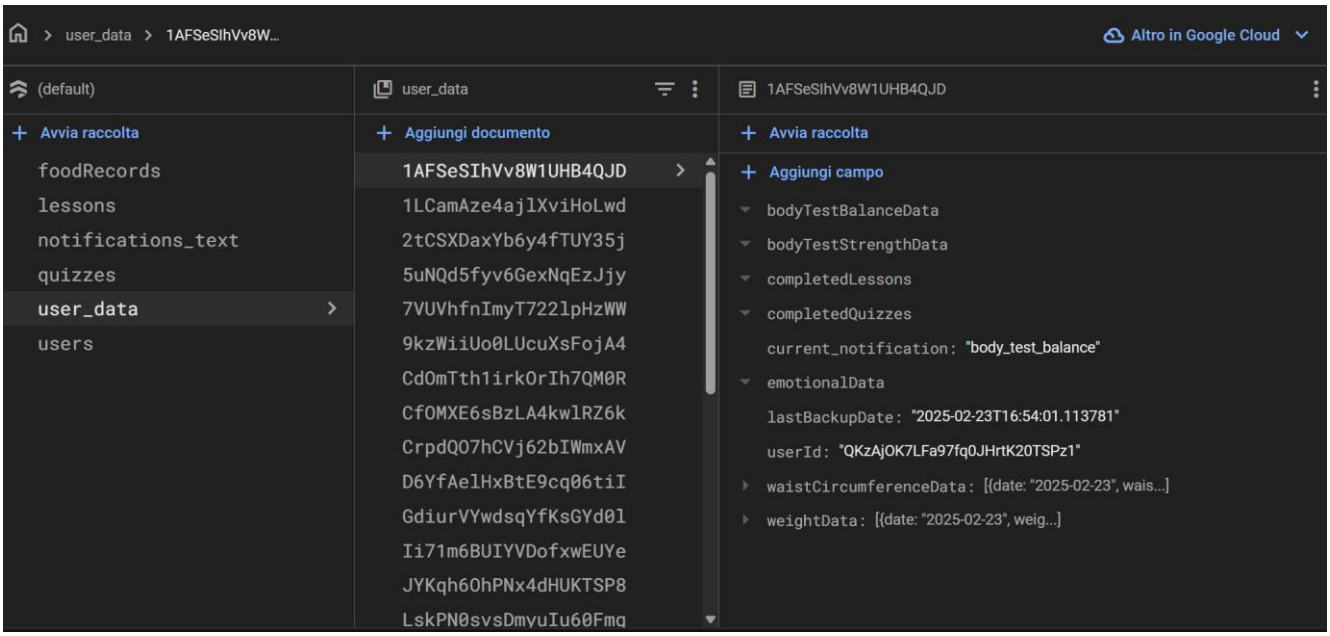


Figure 11: Firestore Database

In Figure 11 all collections of the database are shown. They are:

- users: this collection contains general information about registered users such as User Id, goals (daily steps for example), wearable device used and personal information. Each document corresponds to a user.
- user_data: this collection contains information related to the application such as user metrics tracked (weight, waist circumference, balance data, strength data), lessons and quizzes completed and finally data for the functioning of timed notifications.
- foodRecords: is used to track users' food inserted by users. In detail each document has registration date, user Id, food amount and type.
- lessons: this collection has a document for each lesson. Each document has an id, the corresponding quiz id, two titles and two pills (micro-lessons): one for

English lessons and one for Italian lessons. The structure is the same, but thanks to this division it easily to modify the lessons' text in only one language using the Admin Management Tool.

- quizzes: this collection follows the style of the lessons collection, with question and answers fields in both English and Italian.
- notifications_text: this is a support collection with 3 documents. All the data in this collection are editable in the Admin Management Tool. One of the three documents is for storing parameters values of the application (coach name and assessment period duration). The other two, instead, store the texts shown in notifications and in the onboarding phase of the application. The two documents contain the same keys, but the values change: one contains the texts in Italian and the other one the texts in English.

3.6 ENABLING TECHNOLOGIES

3.6.1 WELL-BEING APP TECHNOLOGIES

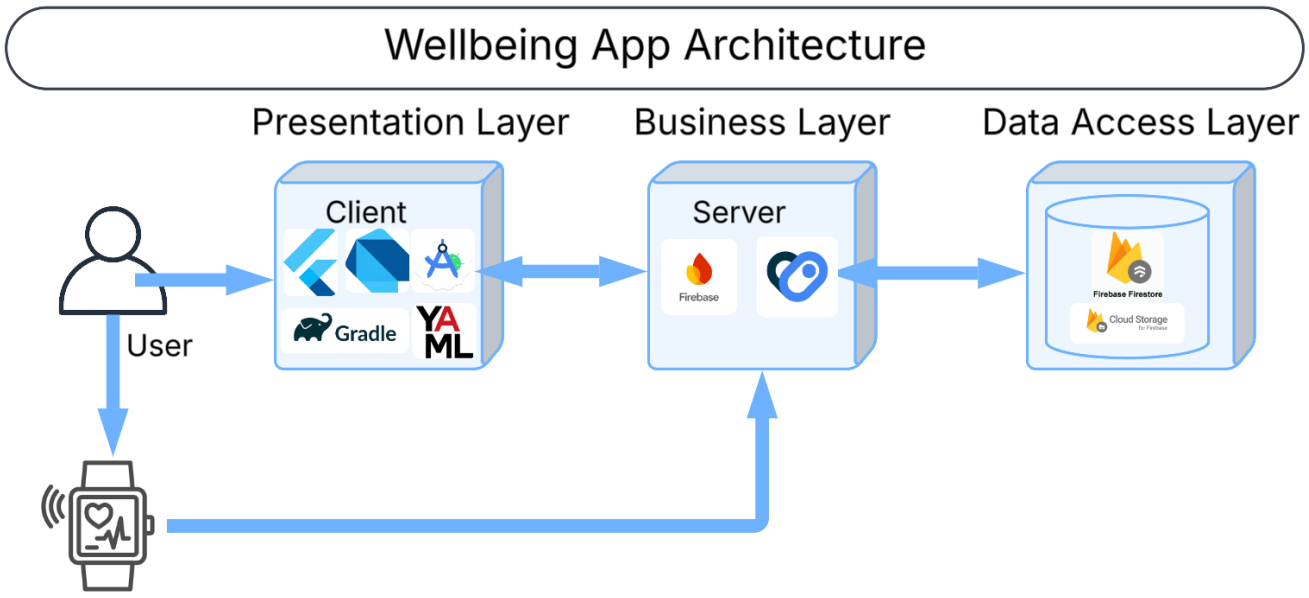


Figure 12: Wellbeing App Technology Mapping

The technologies used for the Wellbeing app are:

- **Flutter**

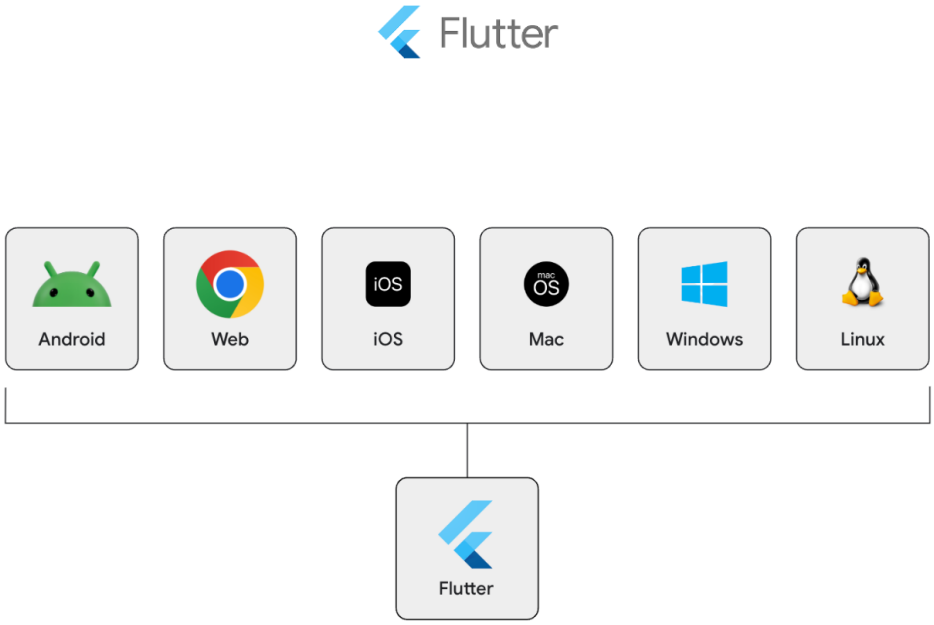


Figure 13: Flutter - Multi-platform framework

Flutter is an open-source framework developed by Google for building natively compiled applications for mobile (iOS and Android), web, and desktop from a single codebase [8]. Flutter offers several advantages beyond cross-platform development. Flutter's hot reload feature provides a reactive framework that permits developers to build UI, add features, and fix bugs quickly and easily. Hot reload works by injecting the updated source code file into the running Dart virtual machine (VM). After the VM updates the classes with the functions modified, the Flutter framework automatically rebuilds the widget tree, showing the effects of the changes [9]. Furthermore, Flutter provides a series of pre-made widgets allowing the creation of personalized and interactive widgets. Another of the advantages of Flutter is its adaptive design: being designed to work on different platforms makes it easy to implement adaptive and responsive interfaces (Figure 13).

- **Dart**

Dart is an object-oriented, class-based programming language developed by Google. It is the main language used to build Flutter applications. It is designed to be strongly typed and has features that make it quite modern as a language like null safety and pattern matching. Also, use hot reload to see the effects of code changes instantly [10]. Dart was created with the intention of solving JavaScript problems and maintaining its performance and functionality. The Dart software development kit (SDK) comes with a stand-alone virtual machine, which allows Dart code to be executed in a command line interface environment. In addition, the Dart code can be filled in advance (ahead of time) or at the moment (just in time), but in the first case the AOT compilation is useful to have highly efficient native machine code [11]. Dart is mainly used in the development of mobile applications with Flutter, but it is also applicable to server-side development, web applications and command line tools.

- **Android Studio**

Android Studio is the official IDE (Integrated Development Environment) for the development of Android applications. It offers a set of tools that facilitate the creation of mobile applications. One of these is Jetpack Compose. This tool is very useful during the design phase because it allows developers to create dynamic layouts and see a preview of them, so that it is possible adapt animations and layouts for any screen. However, this feature was not used in the design of the prototype under study. In addition, in the development phase, the IDE assists the developer with an intelligent code editor that provides code completion for different programming languages. Another advantage is the Android Studio's build

system, powered by Gradle. This allows the customization of the builds and generates variants that start from a single project and adapt to different android devices. In addition, the android emulator enables the testing of applications on different android devices. By following this method, it is possible to determine whether the layouts are suitable for various devices (phones, tablets, ChromeOS, TV) [12].

- **Gradle**

Gradle is an open-source build and automation tool, implemented in Java. It is used to build apps in different ecosystems: JVM-based (such as Java, Kotlin and Groovy), Android, Native (such as C, C++ and Swift) or other (Python, Go, etc.). Gradle builds are divided into 3 phases [13]. Each phase executes a part of code defined in the build files. The first phase is initialization. At this stage the projects and sub-projects to be included in the build are identified, these are specified in the setting file. In the configuration phase, instead, for each project activities are recorded and the build file is run. The output of this phase is the "Task Execution Graph", which indicates the required steps of compilation by the user in a precise order. The last stage is the execution phase, where the application is created. In detail, the Task Execution Graph is taken as input and performs the tasks defined in the order specified in the graph (Figure 14). The build files can be written in Kotlin or Groovy, in this project it was used the second one.

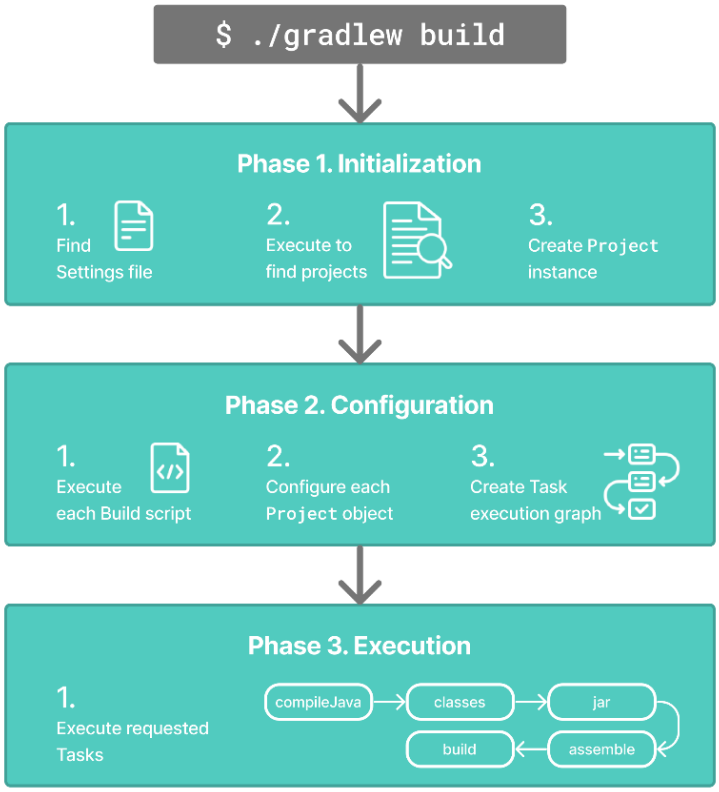


Figure 14: Gradle - Build Lifecycle

- **YAML**

YAML (YAML Ain't Markup Language) is a human-readable data serialization language commonly used to write configuration files [14]. It denotes several advantages, but the main one is its feature of being simple and readable. It is composed of a clean syntax based on key-value pairs. This property allows complex data structures to be expressed in a comprehensible way, similar to human language. This file is generated automatically when creating a new Flutter project. It defines project metadata, dependencies required by the project, assets and fonts used.

- **Firebase**

Firebase is a platform designed by Google to provide a set of cloud-based services for developing mobile and web applications [15]. The Firebase Console offers services such as authentication, hosting, data analytics and real-time database. In detail, it provides two types of databases: Realtime Database and Cloud Firestore, both for storing and updating data in real time in all clients connected. The latter is the type chosen for the realization of the application. In addition, authentication through the service offered by Firebase has been implemented. In particular, authentication is possible via Email and Password or by using a Google account.

- **Health Connect**

Health Connect is a platform powered by Google that allows the integration of health and fitness data across different apps. It collects health data collected via wearable devices and combines them with data collected via smartphone apps and sensors. It also offers the possibility to view these data in detail via Google Fit [16]. It provides a set of APIs to collect, store and display health data. This solution was chosen because it allows reading and writing of many health metrics in a single place. In fact, by collecting and combining data from multiple sources, it provides a complete picture of a user's well-being and allows the visualization of data of all 4 spheres of well-being: nutrition, sleep, physical activity and stress.

- **Cloud Firestore**

Cloud Firestore is a Firebase's service and is a NoSQL document database offered by Firebase and Google Cloud, designed for mobile, web and server application development. The choice of this type of database was made to obtain a flexible, scalable and reliable product. Its main feature is real-time synchronization between client and database, so as to have updated data on any connected device. [17] In particular, data is organized into documents and collections. Each collection has a name and may contain zero or more documents. A document consists of a JSON data structure where each entry is a key-value pair. [18]

- **Cloud Storage**

Cloud Storage for Firebase is a fast, secure Google Cloud infrastructure that allows storing and managing photos, videos, documents or other application-generated content [19]. In the mobile application, this Firebase service allows memorizing periodic backup of data fetched from Health Connect, which is useful to create personalized recommendations for the user.

3.6.2 ADMIN MANAGEMENT TOOL TECHNOLOGIES

The technologies used for the Admin Management Tool are:

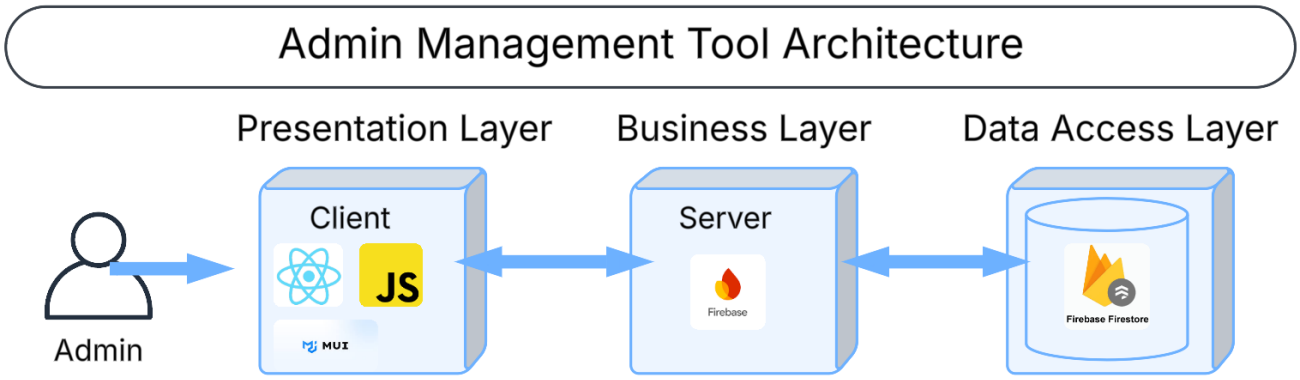


Figure 15: Admin Management Tool Technology Mapping

- **React and MUI**

React is an open-source JavaScript library useful for building user interfaces. These interfaces are built defining components, that are pieces of the interface that correspond to JavaScript functions [20]. React offers high performance via a virtual representation of the DOM: this ensure that only necessary components are re-rendered, minimizing the changes to the actual DOM. In addition to the division into components, React facilitates programming with the introduction of hooks. By using these functions, it is possible to monitor the state of React and the component lifecycle. The most important ones are: useState (to keep track of application states), useEffect (to manage side effects in components) and useContext (which allows components to access context values directly, without having to pass them through nested components). For the development of the admin management Tool interface, both React and Material UI were used. Material UI (MUI) is a React component library that provides a set of UI pre-defined customizable components such as buttons, tables, text fields and icons.

- **JavaScript**

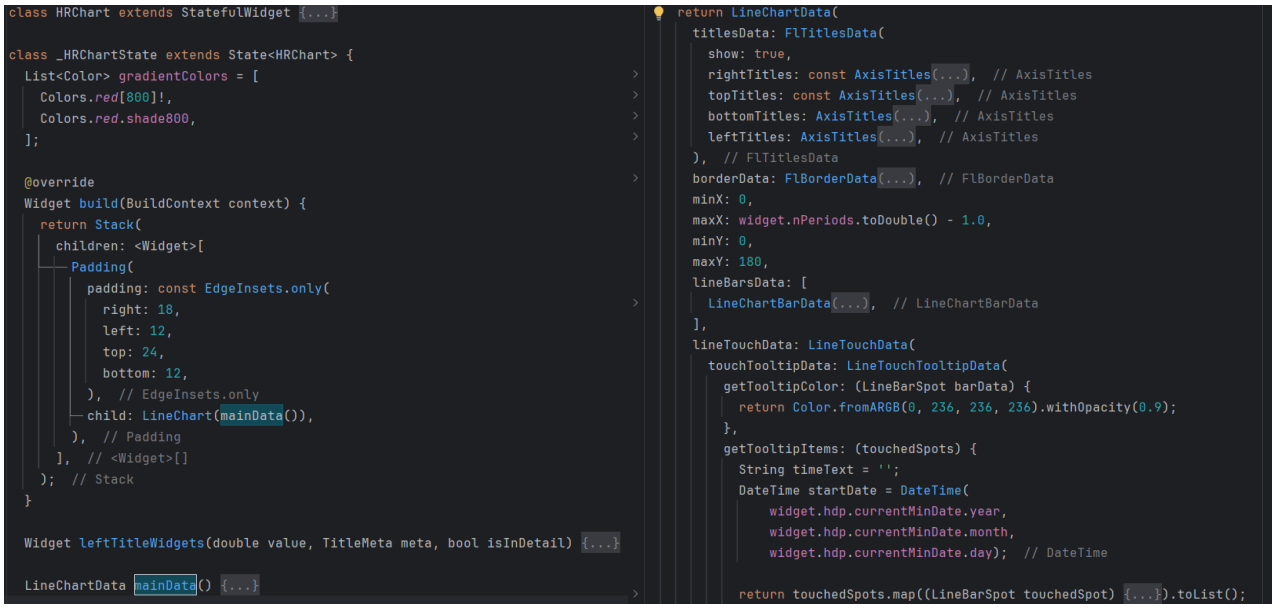
JavaScript is a high-level programming language used for the development of both clients and servers. It was initially introduced to make the web pages interactive and now 99% of websites utilize it to handle the behavior of the webpage's clients [21]. Its main features make the code: imperative and structured (supports C syntax structure), weakly typed, dynamic (variables are not bound to a specific data type), object-oriented, functional (functions are first-class), delegative (supports traits and inheritance).

- **GitLab**

GitLab is a web-based tool that provides a Git repository manager [22]. It permits us to manage the code thanks to its features of branching, merging and access control. Other features include automate testing, building and deployment of applications. They also provide support to the developer teams through a system of project planning, milestones and problem tracking. It was useful in the production phase of both the application and the admin Management Tool to handle the cooperation of the different members who participated in the development of the code.

3.7 IMPLEMENTATION

Well-Being App: Chart implementation



```
class HRChart extends StatefulWidget {...}

class _HRChartState extends State<HRChart> {
  List<Color> gradientColors = [
    Colors.red[800]!,
    Colors.red.shade800,
  ];

  @override
  Widget build(BuildContext context) {
    return Stack(
      children: <Widget>[
        Padding(
          padding: const EdgeInsets.only(
            right: 18,
            left: 12,
            top: 24,
            bottom: 12,
          ),  // EdgeInsets.only
          child: LineChart(mainData()),
        ),  // Padding
      ],  // <Widget>[]
    );  // Stack
  }

  Widget leftTitleWidgets(double value, TitleMeta meta, bool isInDetail) {...}

  LineChartData mainData() {...}
```

```
return LineChartData(
  titlesData: FlTitlesData(
    show: true,
    rightTitles: const AxisTitles(...),  // AxisTitles
    topTitles: const AxisTitles(...),  // AxisTitles
    bottomTitles: AxisTitles(...),  // AxisTitles
    leftTitles: AxisTitles(...),  // AxisTitles
  ),  // FlTitlesData
  borderData: FlBorderData(...),  // FlBorderData
  minX: 0,
  maxX: widget.nPeriods.toDouble() - 1.0,
  minY: 0,
  maxY: 180,
  lineBarsData: [
    LineChartBarData(...),  // LineChartBarData
  ],
  lineTouchData: LineTouchData(
    touchTooltipData: LineTouchTooltipData(
      getTooltipColor: (LineBarSpot barData) {
        return Color.fromARGB(0, 236, 236, 236).withOpacity(0.9);
      },
      getTooltipItems: (touchedSpots) {
        String timeText = '';
        DateTime startDate = DateTime(
            widget.hdp.currentMinDate.year,
            widget.hdp.currentMinDate.month,
            widget.hdp.currentMinDate.day);  // DateTime

        return touchedSpots.map((LineBarSpot touchedSpot) {...}).toList();
```

Figure 16: Well-Being App Heart Rate Chart Class

In Figure 16 is presented the definition of the HRChart widget. The Flutter code uses the fl_chart library to represent data fetched from Google Health regarding heart rate in a specific time period (day, week or month). All charts developed in the application follow this structure. Going into the detail of this graph it is evident that on the x-axis there is the time interval while on the y-axis there are the heart rate ranges (Widgets in Figure 17). The HRChart class extends StatefulWidget, allowing it to maintain mutable state that can change over time. It receives 3 parameters:

- the HomeDataProvider hdp, from which the heart rate data points are taken;
- nPeriods that is an integer that serves to set the graph display on day, week or month;
- isInDetail: a boolean that determines whether the chart is on the Home screen or in detail after clicking on it.

In the figure it is shown that the build method creates the chart structure. The most important pieces are mainData() which configures the data and how it should appear on the chart and LineChartData which specifies the theme of the chart.

```dart
Widget leftTitleWidgets(double value, TitleMeta meta, bool isInDetail) {
  TextStyle style = TextStyle(
    fontWeight: FontWeight.bold,
    fontSize: isInDetail ? 13 : 10,
  );
  String text;
  switch (value.toInt()) {
    case 60:
      text = '60bpm';
      break;
    case 120:
      text = '120bpm';
      break;
    case 180:
      text = '180bpm';
      break;
    default:
      return Container();
  }

  return Text(text,softWrap: false, style: style, textAlign: TextAlign.left);
}
```

```dart
return touchedSpots.map((LineBarSpot touchedSpot) {
  int index = touchedSpot.x.toInt();
  switch (widget.hdp.currentTopBarSelect) {
    case "day":
      switch (index) {
        case 0:
          timeText = "0-6";
          break;
        case 1:
          timeText = "6-12";
          break;
        case 2:
          timeText = "12-18";
          break;
        default:
          timeText = "18-24";
          break;
      }
      break;
    case "week":
      timeText = DateFormat('EEEE')
          .format(startDate.add(Duration(days: index)));
      break;
    case "month":
      timeText = DateFormat('EEEE d')
          .format(startDate.add(Duration(days: index)));
      break;
  }
```

Figure 17: Well-Being App Heart Rate Chart Widgets

Health: Authorization and Fetch Data

```dart
/// Install Google Health Connect on this phone.
Future<void> installHealthConnect() async =>
    await Health().installHealthConnect();
/// Authorize, i.e. get permissions to access relevant health data.
Future<void> authorize() async {
  Health().configure();
  bool? hasPermissions = false;
  final healthConnectStatus = await Health().getHealthConnectSdkStatus();
  if (
      healthConnectStatus == HealthConnectSdkStatus.sdkUnavailable ||
      healthConnectStatus == HealthConnectSdkStatus.sdkUnavailableProviderUpdateRequired) {
    await installHealthConnect();
    return; // Stop further data requests if Health Connect is not available.
  }
  await Permission.activityRecognition.request();
  // Check if we have health permissions
   hasPermissions =    await Health().hasPermissions(types);
  bool success = await Health().requestAuthorization(types);
  if (!hasPermissions!) {
    try {
      if (!success) {
        debugPrint("Authorization failed. Unable to fetch data.");
        return; // Stop further data requests if authorization fails.
      }
    } catch (error) {
      debugPrint("Exception in authorization: $error");
    }
  }else{
    debugPrint("Already has permissions,authorization");
  }
}
```

Figure 18: Health Authorization Code

The code snippets in Figure 18 explain how the health package works and in particular how the permissions required by the user for the correct operation of the application work.

The installHealthConnect() function initializes the installation process of Health Connect, which is essential for collecting user data and providing a picture of their state. Instead, the authorize() function handles the process of authorizing Health Connect data. First initialize the health plugin using Health.configure(), then check if the SDK is available or requires updates, in case it is not available exit from the function. Then check if the application already has the necessary permissions, otherwise it requires them for all types defined in the types data structure, which collects the types of data that are shown in the application (heart rate, oxygen, steps, etc.). If an error occurs, it produces an error message.

```
Future<List<DefaultDataPoint>> fetchNutritionDataPoints(DateTime startDate, DateTime endDate) async {
  try {

    List<HealthDataPoint> healthDataPoints = await Health().getHealthDataFromTypes(
        types: [HealthDataType.NUTRITION ],
        startTime: startDate,
        endTime: endDate,
    );

    List<DefaultDataPoint> defaultDataPoints = [];
    healthDataPoints.forEach((element) {
      defaultDataPoints.add(DefaultDataPoint.fromHealthDataPoint(element));
    });
    return defaultDataPoints;
  } catch (e) {
    print("Error fetching health data: $e");
    return [];
  }
}
```

Figure 19: Health Fetch Data

In Figure 19 the fetchNutritionDataPoints function shows how data is fetched from Health Connect. This function fetches the nutrition data, but the same structure was adopted for other types of data. The function inputs two parameters that determine the time period in which data will be taken. Using Health().getHealthDataFromTypes, data of the specified type in types are retrieved, over the time period from startTime to endTime. Data is then converted from HealthDataPoint to DefaultDataPoint using the fromHealthDataPoint constructor. Finally, the catch statement shows any errors that occurred during the fetch.

Admin Management Tool: Interface Implementation

```
function AppParams(props) {
    const [selectedSection, setSelectedSection] = useState('App Parameters');
    const sideBarSections = [
        { text: 'App Parameters', icon: <TuneIcon /> },
        { text: 'onBoarding', icon: <PlaylistAddCheckIcon /> },
        { text: 'Daily Reminder Food', icon: <FastfoodIcon /> },
        { text: 'Body Test Balance', icon: <Balance/> },
        { text: 'Body Test Strength', icon: <FitnessCenterIcon /> },
        { text: 'Emotional Life Test', icon: <FavoriteIcon /> },
        { text: 'Assessment', icon: <AssessmentIcon /> },
        { text: 'Add Lesson/Quiz', icon: <AddCircleIcon /> },
        { text: 'Lessons', icon: <MenuBookIcon /> },
        { text: 'Quizzes', icon: <ChecklistIcon /> },
    ];
    const [language, setLanguage] = useState("English");

    var componentToRender;
    switch (selectedSection) {
        case 'App Parameters':
            componentToRender = <AppParameters/>;
            break;
        case 'onBoarding': ···
        case 'Daily Reminder Food': ···
        case 'Body Test Balance': ···
        case 'Body Test Strength': ···
        case 'Emotional Life Test': ···
        case 'Assessment': ···
        case 'Add Lesson/Quiz': ···
        case 'Lessons': ···
        case 'Quizzes': ···
    }
```

Figure 20: Admin Management Tool - Interface Sidebar Sections

Figures 20 and 21 show the main code of the management tool interface for admin. It was implemented using React and Material-UI. The AppParams component uses hooks to manage states and render the interface based on one of the selected sections in the Drawer, which is a navigation sidebar. Each section contains a series of forms for data insertion where it is possible to edit the application through API calls to Firebase. It possible to notice a menu that allows admins to change the language set to English or Italian, this updates the interface showing the related application texts. Additionally, there is a logout button, since to access this interface users need to log in to prove that they are administrators.

```
function AppParams(props) {
    return (
        <Box sx={{ display: 'flex', height: '100vh' }}>
            <Drawer…
            </Drawer>
            <Box component="main" sx={{ height: "100%", width: "100%" }}>
                <Box sx={{ display: 'flex', justifyContent: 'space-between', alignItems: 'center', mb: 2 }}>
                    <Typography component="div" sx={{ flex: 1, textAlign: 'center', width: '70%' }} variant="h4" gutterBottom>
                        Change Well Being App Parameters:
                    </Typography>
                    {selectedSection !== 'Add Lesson/Quiz' && selectedSection !== 'App Parameters'? <Select
                        label="Language"
                        name="language"
                        value={language || ''}
                        onChange={(e) => { setLanguage(e.target.value) }}
                        sx={{ width: '12%' }}
                    >
                        <MenuItem value={"Italian"}>
                            <ListItemIcon>
                                <img src={italianFlag} alt="Italian" width="48" height="27" />
                            </ListItemIcon>
                        </MenuItem>
                        <MenuItem value={"English"}>
                            <ListItemIcon>
                                <img src={englishFlag} alt="English" width="48" height="27" />
                            </ListItemIcon>
                        </MenuItem>
                    </Select> : ''}
                    <Box sx={{ display: 'flex', justifyContent: 'flex-end', alignItems: 'center', width: '18%' }}>
                        <Button
                            onClick={props.handleSignOut}
                            variant="contained"
                            color="primary"
                        >
                            Logout<LogoutIcon sx={{ ml: 1 }}/>
                        </Button>
                    </Box>
                </Box>
                {componentToRender}
            </Box>
        </Box>
    );
```

Figure 21: Admin Management Tool - Interface Code

CHAPTER 4.

HOW THE PROTOTYPE WORKS

4.1 DATA

Some of the data in the database used during the presentation of the application scenario are presented below.

Figure 22: DB - user_data and users

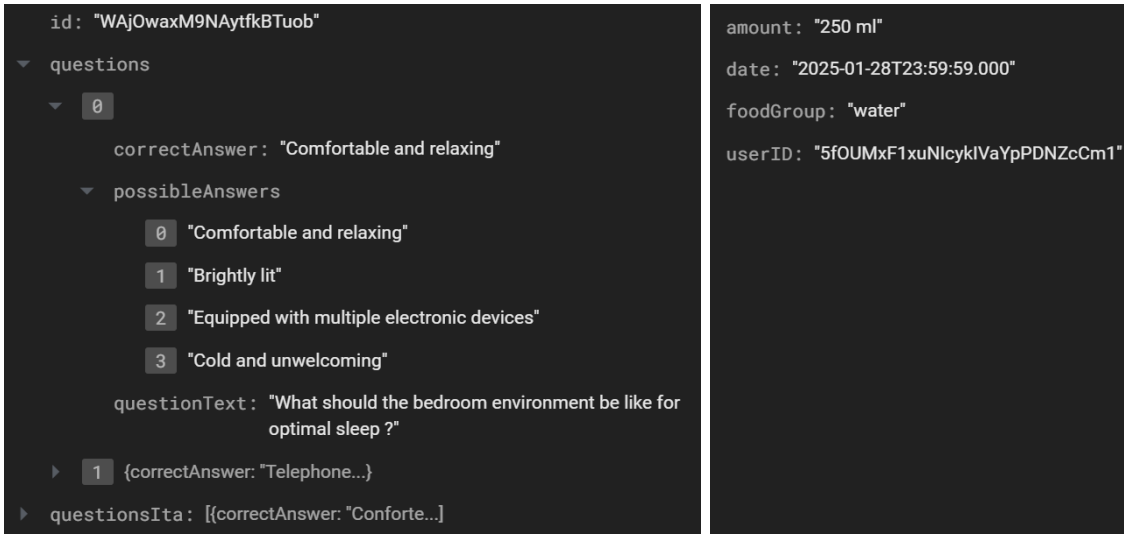


Figure 23: DB - quizzes and foodRecords

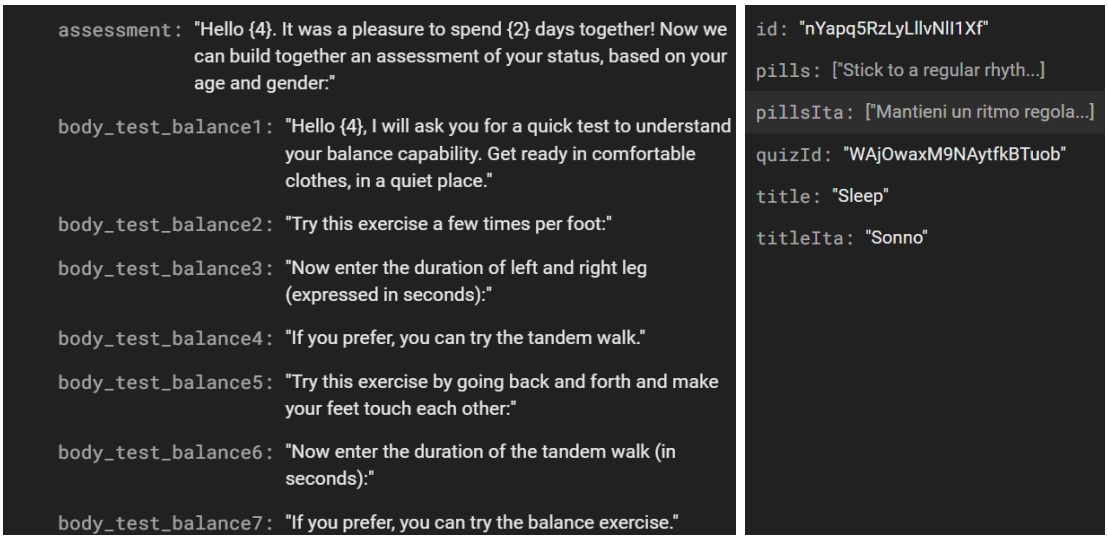


Figure 24: DB - notifications_text and lessons

4.2 APPLICATION SCENARIO: USER

The first application scenario discussed will be that of a general user downloading the Well-Being App for the first time. A series of steps will follow, explaining what the user will see and how he will interact with the GUI.

Onboarding

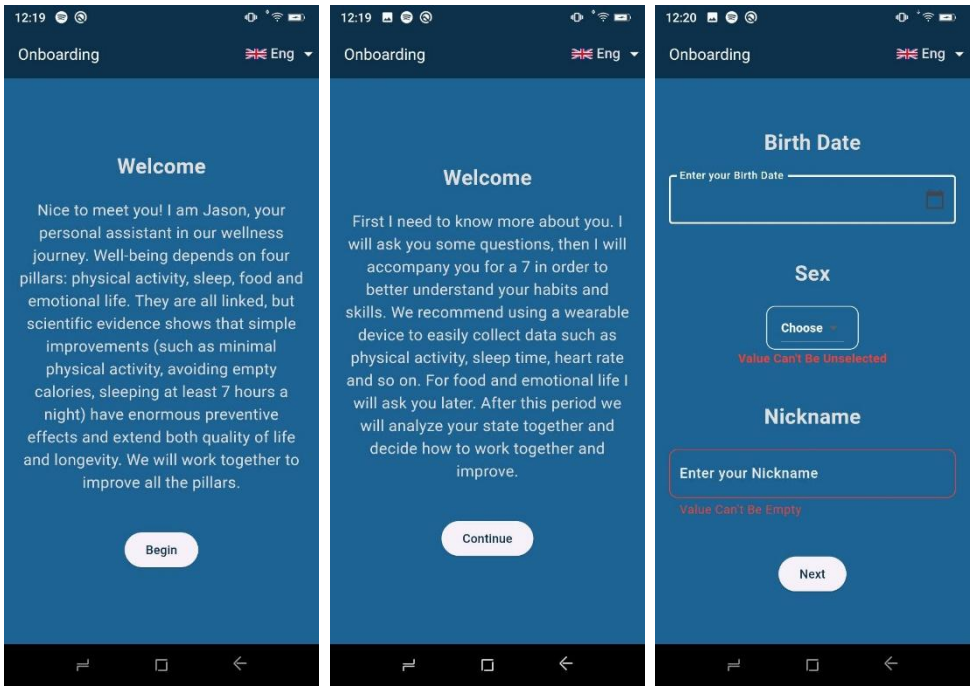


Figure 25: onBoarding 1

In the onboarding phase, the user is asked for personal information to understand his habits and his starting state. Figures 25 and 26 show the screens that the user sees and the forms (with field validation active) to enter date of birth, gender, nickname, wearable

device used, height, weight and abdominal circumference. The user can also decide which language is preferred in order to set the app to that language. This setting is always modifiable in the future. In addition, the user is explained how it will be defined the assessment of his status.

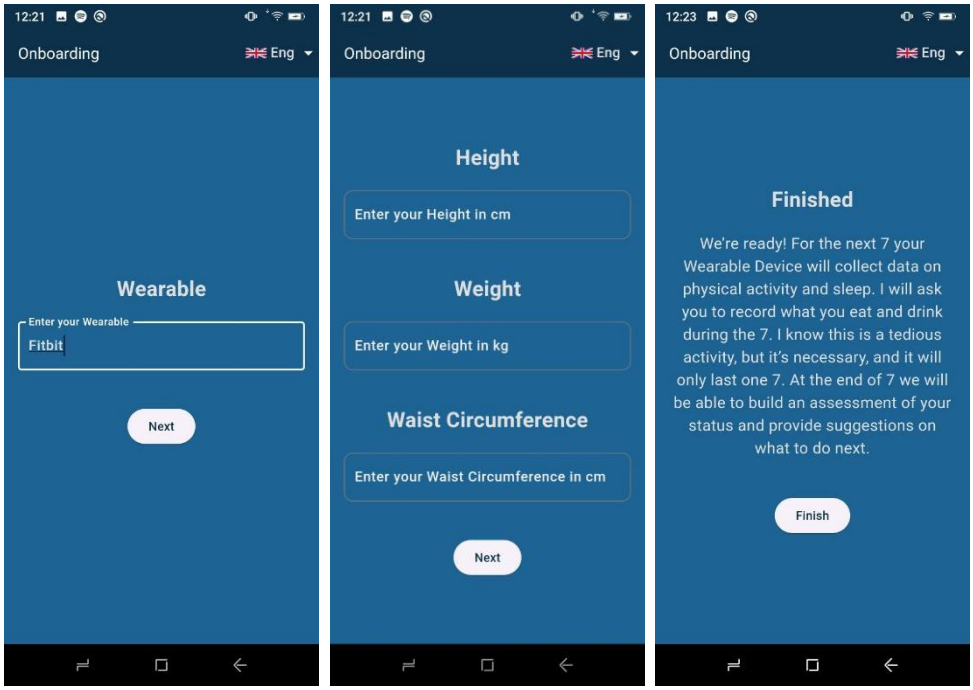


Figure 26: onBoarding 2

After that, the authorization phase to collect data from Health Connect takes place.

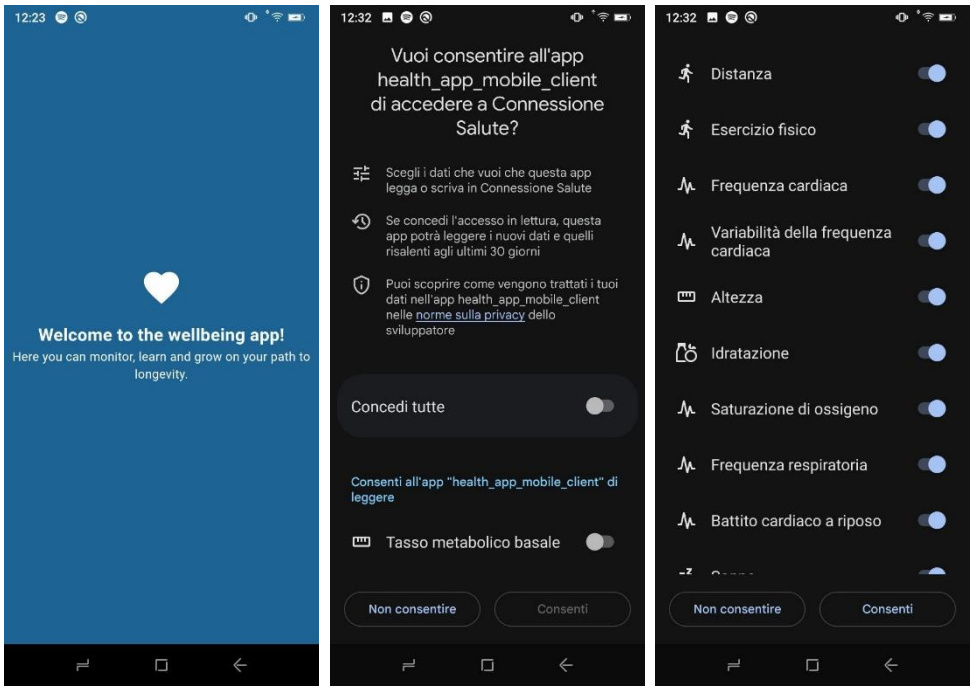


Figure 27: Health Connect Authorization

After the user has authorized the app to read data, it starts fetching the data and the application shows the home. This is also the starting point for users who are already registered and log in, skipping the onboarding phase (scenario 2).

Home

Figure 28: Home (day, week, month)

The Home is presented as in Figure 28. There are 4 main graphs: Steps, Food, Sleep and Mood. In addition, thanks to the navigation bar at the top, it is possible to view the data by day, week or month. The graph steps in the "day" view allows users to see the number of steps divided by 4 time slots, each 6 hours. In addition, the graph shows the total number of steps taken during the day and highlights the bar if the user has exceeded his goal (this applies to the bars of all graphs and for each time selection). The "Food" chart displays the foods that have been consumed in the day added by the user with their respective quantities. The "Sleep" graph shows the total sleep minutes and divides them according to the type of sleep: Awake, Light Sleep, Deep Sleep and REM. Finally, the "Mood" chart shows 2 bars: the green bar shows the average of positive affects while the red bar shows the average of negative affects. The values in question refer to the measures of mood following the study I-PANAS-SF by Thompson [23]. Weekly and monthly

views, on the other hand, group data by week and month respectively and show a daily average of each measurement. In the case of the "Food" chart, the unit of measurement are the calories, while in the case of sleep and steps the total values of each day are taken and represented. Instead, for the mood an average of the values entered during the week or month is shown.

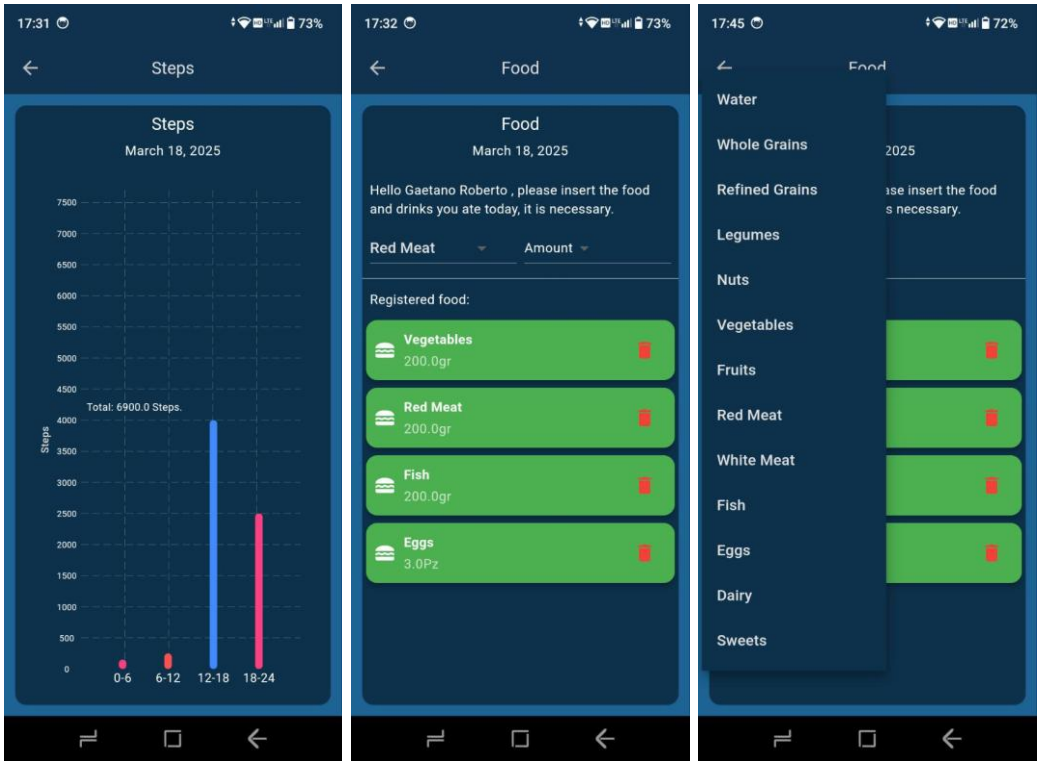


Figure 29: Steps Chart - Food List

By tapping on a single chart, users can go to the detailed view of the same, as shown in Figures 29 and 30. In the case of Steps and Sleep, users have a wider display of measurements, instead in the case of Food and Mood they have the possibility to enter the relative values. The Form Food allows users to add the food consumed by choosing from two dropdowns the type of food and the corresponding quantity (Figure 29). Instead, the Mood chart allows users to enter data related to the mood by assessing the user's status following the PANAS standard. In particular, it is necessary to evaluate if the user feels active, determined, attentive, inspired and alert, regarding the positive affects value, and how much he is afraid, nervous, upset, hostile and ashamed for the negative affects value. For each of these characteristics it is possible to express a value from 1 to 5 based

on how much one perceives that condition. In addition, at the end of the form there is a detailed display of the Mood chart (Figure 30).

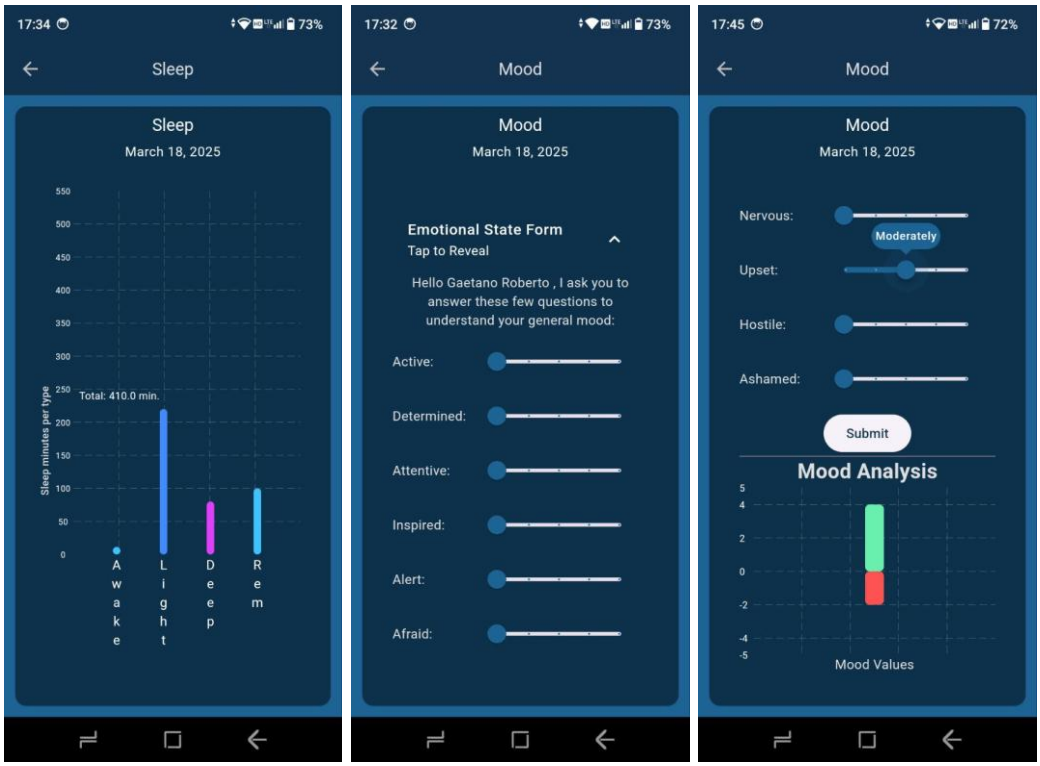


Figure 30: Sleep Chart - Mood Form - Mood Chart

Lessons

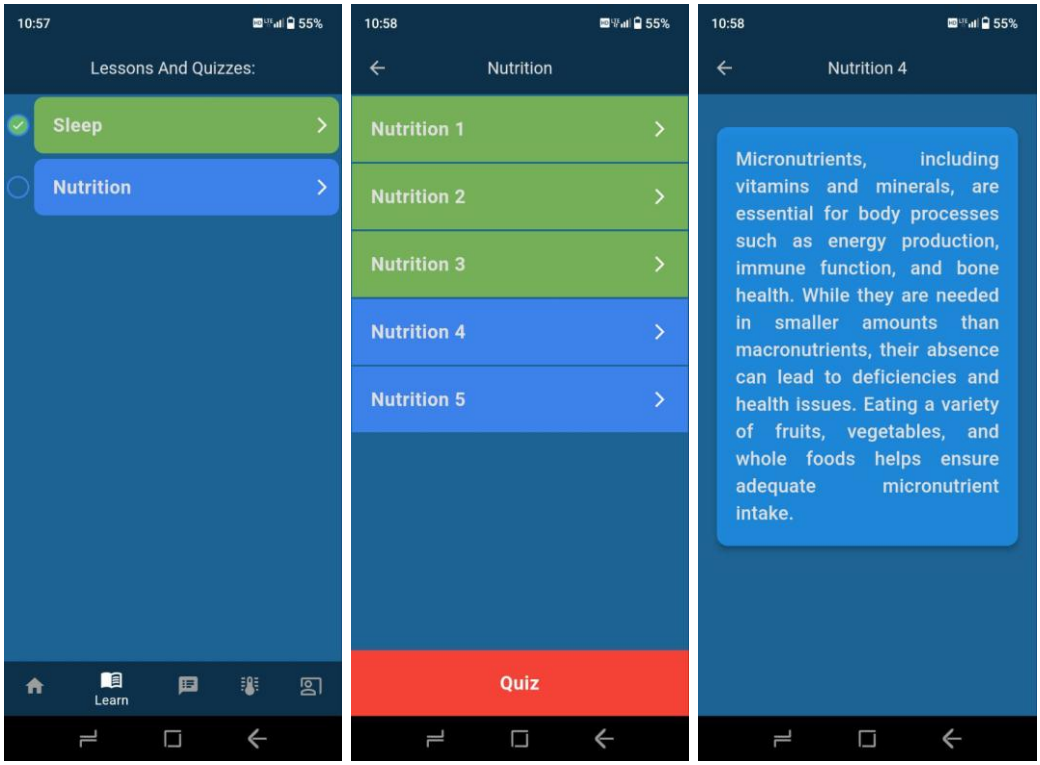


Figure 31: Lessons

From the Home users can move to other screens through the bottom tab menu. The next tab to the Home is "Learn" (Figure 31). Here the user can learn how to improve his health and habits through lessons and test his knowledge. In the first screen a list of topics appears, these are green if the user has already completed the lesson or blue if there are still some knowledge pills to read. Clicking on a topic brings up the knowledge pills which are micro lections. These also appear green if completed or blue otherwise. Clicking on one of these appears the respective text and the pill is set completed. In the screen with the list of pills it is also possible to test the knowledge acquired with a quiz on the whole lesson.

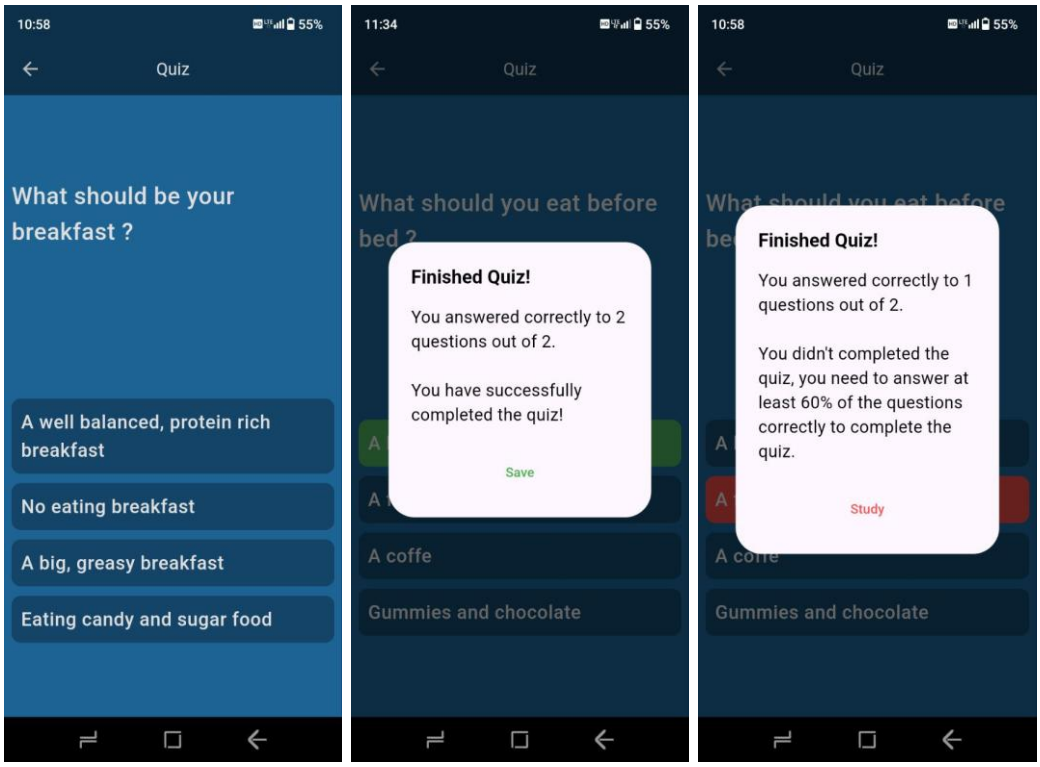


Figure 32: Quizzes

Each quiz contains a series of multiple-choice questions. To complete the quiz, the user must answer at least 60% of the questions correctly (Figure 32).

Assessment

The "Recommend" screen evaluates the metrics entered by the user. The metrics taken into account are those ranging from the current day to "assessment_period" days before. At present, assessment_period is set to 7 days, so the data considered are those of 7 days earlier today. The data covers 4 spheres: Balance, Strength, Mood and Food. For each

category, the average data entered by the user is compared with the standard data of an average user of the same age and gender (Figure 33). Going into detail, the age groups considered are Children (6-12 years), Teenagers (13-19 years), Young adults (20-39 years), Middle-aged adults (40-59 years) and Older adults (60+ years). The metrics considered for each age group and gender are seconds in balance per leg, tandem walk, number of squats, abs and push-ups, strength resulting from the grip test, daily calories taken, values of positive and negative affects.

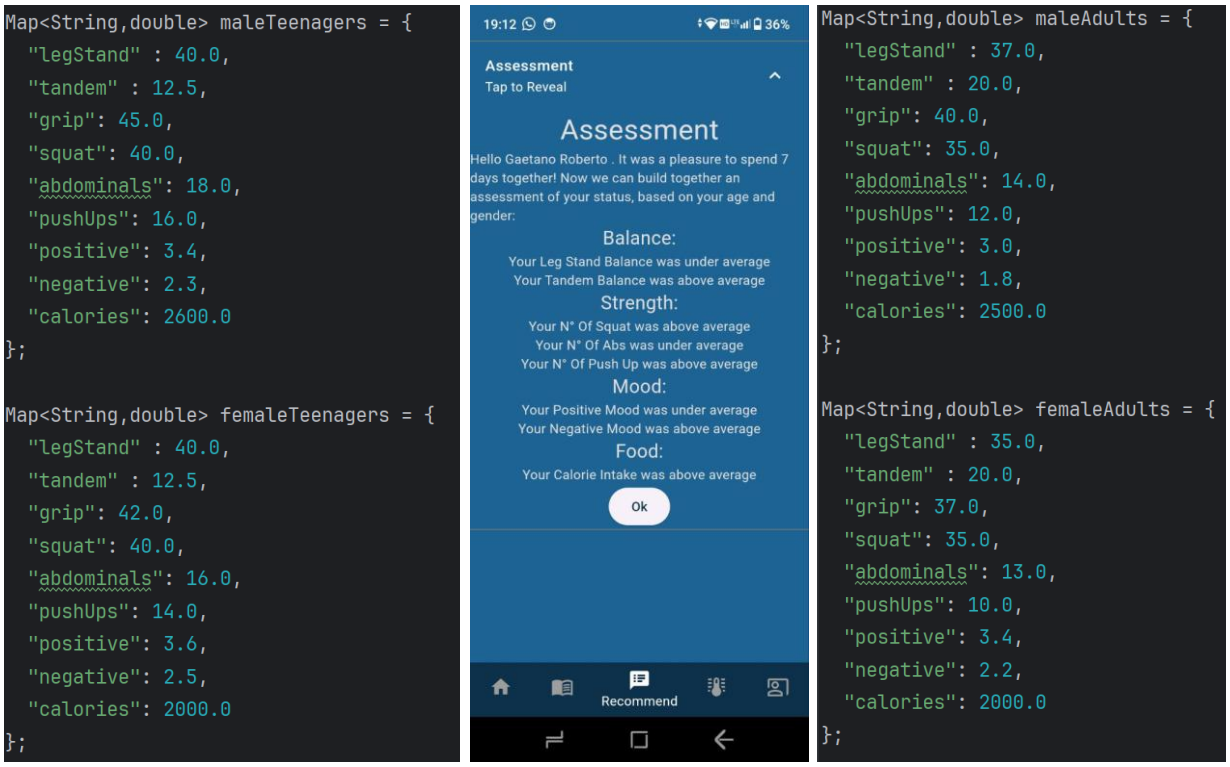


Figure 33: Assessment and standard metrics

For each of these metrics are compared the data of the user and in case they are higher appears a message that explains to the user that his characteristic is above the average for his age and sex, lower otherwise.

Health Measures

The fourth tab is "Health Measures", from here it is possible to see the data fetched by Health Connect of oxygen saturation, respiratory rate, resting heart rate, heart rate variability in daily, weekly, monthly or detailed view. It is also possible to display and enter information regarding the balance and strength of the user.

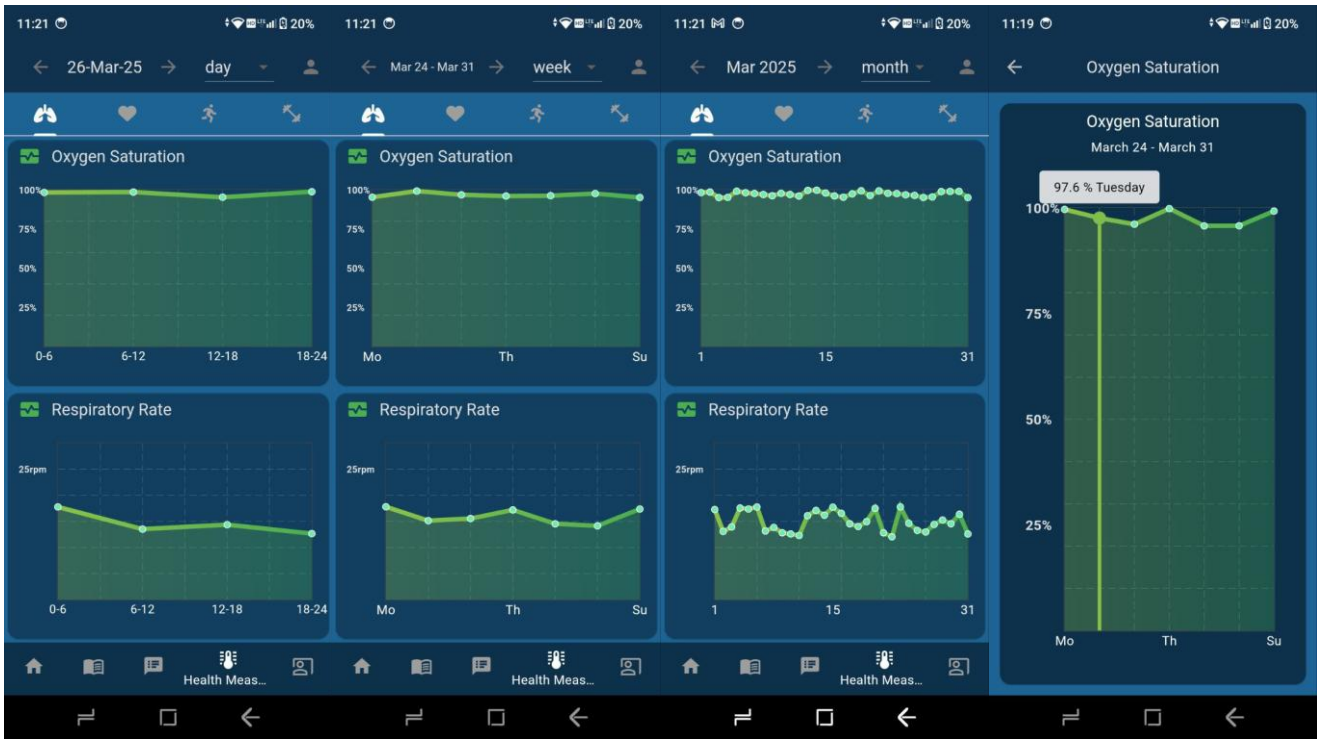


Figure 34: Oxygen (day, week, month, detailed)

The first measure of health is oxygen. Oxygen saturation is the measure of how much oxygen is present in a person's red blood cells. For healthy adults, normal values are between 96% and 100% [24]. The respiratory rate, on the other hand, is defined by the number of breaths per minute. For an adult at rest normal values are 12 – 18 bpm. [25]. As shown in Figures 34 and 35, users can change the display of the graphs using the navigation menu at the top. Thanks to this they can view the data of a specific day, week or month. In the daily display of oxygen saturation and respiratory rate are shown the values measured by the synchronized wearable device dividing the values by 4-time bands: 0 - 6, 6 - 12, 12 -18 and 18 - 24. In the monthly and weekly view, instead, values are taken over a day and an average is made, this is the value of that day of the week or month. In the plots there is one point for each day of the week for weekly chart and one point for each day of the month for the monthly chart. This applies to both oxygen and heart rate measurements. As for the heart rate, however, two measures are considered: resting heart rate and heart rate variability (Figure 35). Resting heart rate is calculated by counting the beats per minute at rest. Standard values for a healthy adult man are 60 to 100 bpm [26]. The heart rate variability is the variation in time between heartbeats and is expressed in milliseconds. To calculate these values, the RMSSD method (root mean

square of successive differences) was used and the values considered normal are in the range 40 - 100 ms depending on age, gender, health condition and physical activity [27]. The monthly, weekly and detailed charts work in the same way, what changes is the daily chart. Instead of showing the values by time bands, the average values are shown, being more sporadic measurements than those for oxygen.

Figure 35: Heart Rate (day, week, month, detailed)

Instead, in figures 36 and 37 it is possible to notice how the information about balance and strength of the user is displayed. The charts have the same structure as the previous ones, the difference is in the daily chart. The general view shows the data entered on that day. Instead, by clicking on a specific daily chart, users can enter the data related to the chart. Another way to enter this data is by clicking the notification that appears on the screen during the assessment period. The data that can be entered for balance are weight, waist circumference, seconds in balance on one leg (right and left) and tandem walk. Instead, for strength data, they can enter the result of the grip test (if the user has the dynamometer), number of squats, abs and push-ups that the user is able to perform. In all the graphs presented users can touch a point in detail to know as much information as possible about that day/time slot. As shown in Figure 35, by touching the fifth point of the graph, information appears on Friday for the week in question (March 10 - March 16),

showing a resting heart rate of 72.4 bpm. The values of balance and strength combined with those of mood and food allow the system to evaluate the user's state, giving an overview on all fields.

Figure 36: Balance Data (day, week, month, detailed)

These measurements are the basis of the analysis that the application performs, based on the principles and studies of L. Fontana in the book The Path of Longevity [1]. In detail, he explains that low muscle mass and strength are a risk both to bone health and to be able to perform basic daily tasks such as carrying shopping bags, climbing stairs or lifting heavy objects. In the book he explains which exercises to perform to improve muscle mass without the need for special tools, using only his own body. Likewise, explains Fontana, posture, balance and flexibility of the body are also essential factors to live in health and improve one's well-being. The risks of neglecting these aspects include restricted breathing, increasing risk of bone fractures, chronic neck, back, and knee pain. Therefore, he explains how to adopt a correct posture, and which exercises can improve balance. The tests carried out and the data acquired allow the user to understand whether his condition is good or needs correction.

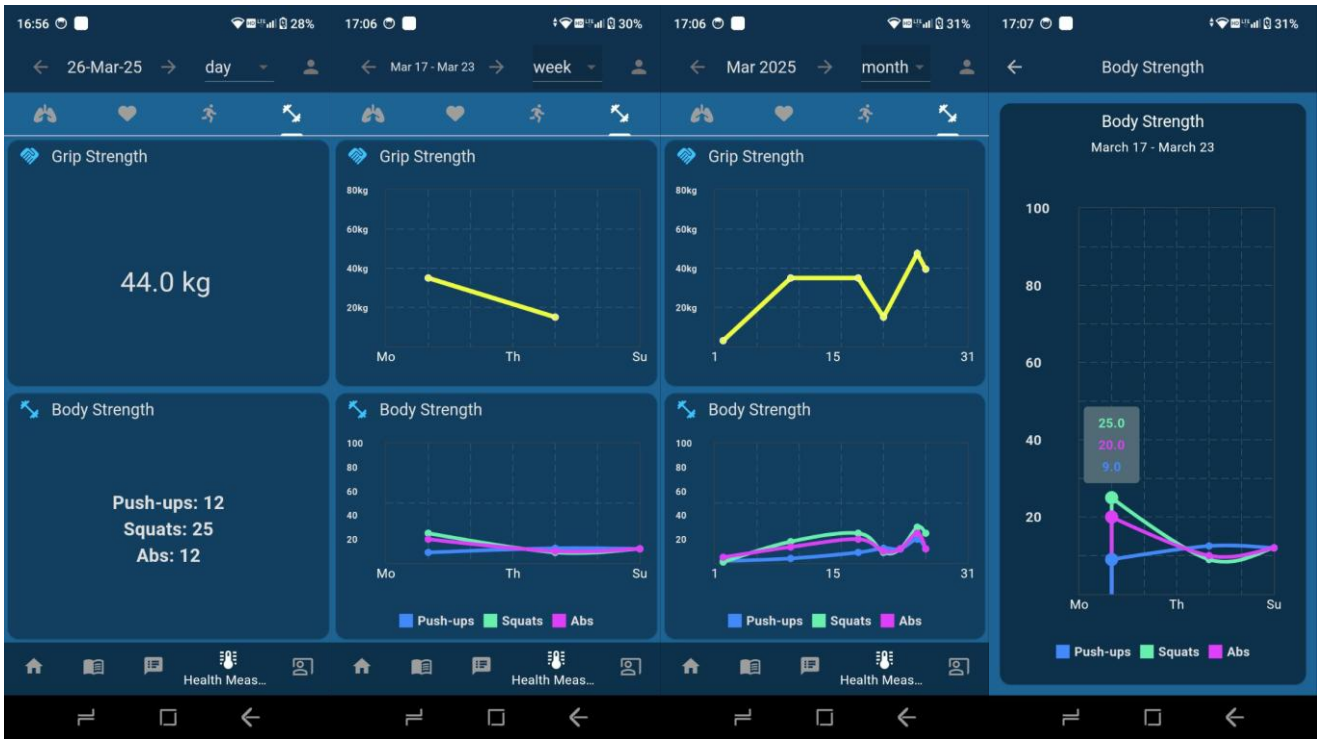


Figure 37: Strength Data (day, week, month, detailed)

Personal Information

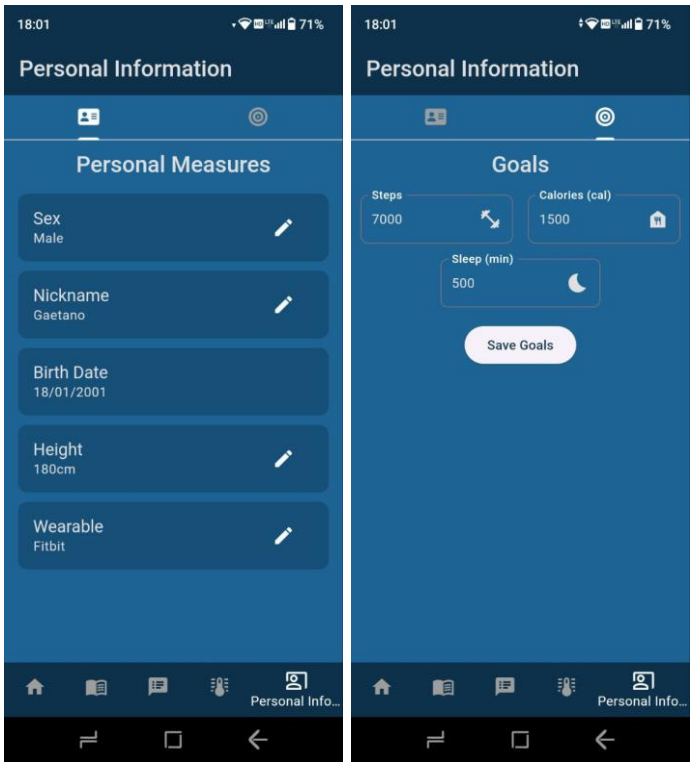


Figure 38: Personal Information – Goals

In the "Personal Information" screen there are two subsections: personal measures and goals. In the first it is possible to edit user's personal information such as gender, nickname, date of birth, height and wearable device used. This information is essential to

make the most of the application, as it compares users' data with that of the average user based on age, gender and other factors in order to assess their health and well-being. In the goals section, instead, users can set goals to be achieved for daily steps, daily calories and minimum sleep time that they want to sleep. These objectives will also change the display of the graphs in the Home, because the bars of the graphs have as max-y set the defined goal and if in a day users exceed the established goal then the bar is highlighted with yellow color and with a thick dotted border (Figure 28).

Notifications

As explained in section 3.3 MAIN PROCESS DIAGRAM), a time-based notification system has been implemented. Figure 39 shows some of the screens that open when a notification is clicked. There are five possible notifications: Daily Food Reminder, Body Test Balance, Body Test Strength, Emotional Life Test and Assessment. The first four have a reminder function to remind the user to enter data on food, balance, strength and mood, while the last notification notifies the user that the assessment period has ended and that it is possible to see the assessment. Clicking on the Daily Food Reminder notification displays a screen with the same form as Figure 29, where users can enter the foods and their quantities eaten that day. This notification is set to repeat every day. After entering the data and clicking on the "Submit" button, the app starts from the home page. This happens for all notifications that require data entry. Body Test Balance and Body Test Strength notifications show the screens of the Figure 39. In both are shown a small introduction for each exercise editable through the Management Tool for admin, a video tutorial that shows how to perform the exercises and data forms to enter data on the exercise just performed. In the Body Test Balance, users can insert the seconds in balance on the right and the left leg, or alternatively, the duration of the tandem walk in seconds. Instead, the strength exercises start with a short video showing how to do the warm-up, then the screens require to perform squats, abs and push-ups and enter the number of those performed. Finally, optionally, they can enter the grip value expressed in Kg. This is optional because it requires the use of a dynamometer. The next notification is Emotional Life Test, this allows to understand the general mood of the user. After clicking

the notification appears the same form presented in Home in Figure 30, with the values of positive and negative affects. The final notification is Assessment, this shows the screen with the evaluation of the individual metrics entered and fetched during the assessment period (Figure 33).

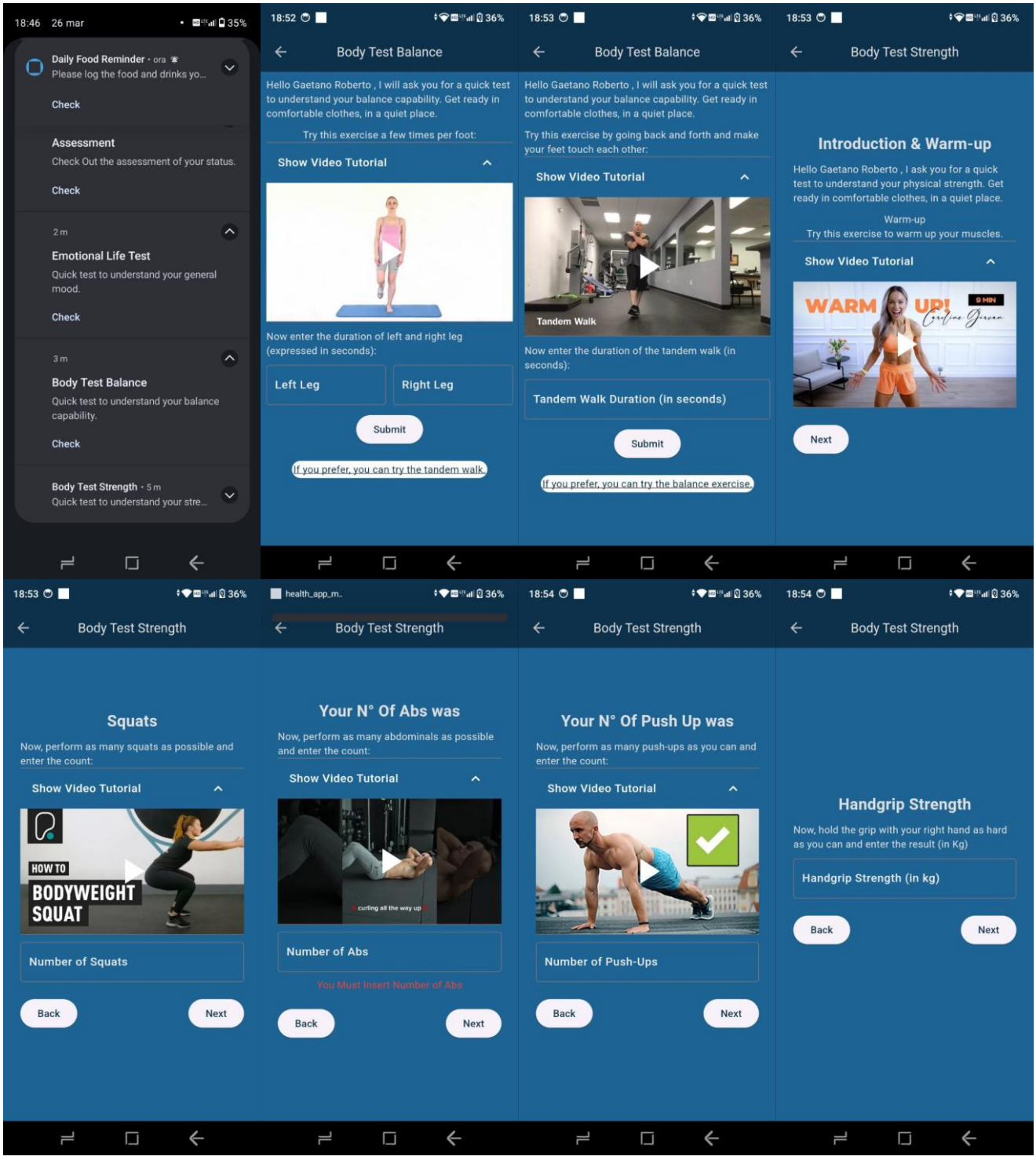


Figure 39: Notifications Screens

Profile

From the Home and other screens, users can access the "Profile" screen by clicking on the man icon in the top right. On this screen they can change their username, set a profile picture, set the language of the application, view login methods, log out or delete the account. The active login methods are currently 2: email and password or Google account. In the case of email and password it is also possible to set a new password in case of loss of the old one. As shown in Figure 40 users have to click "Forgotten Password" and the screen for password recovery appears. From here they have to enter the email for the recovery process, will be sent an email that allows them to set the new password.

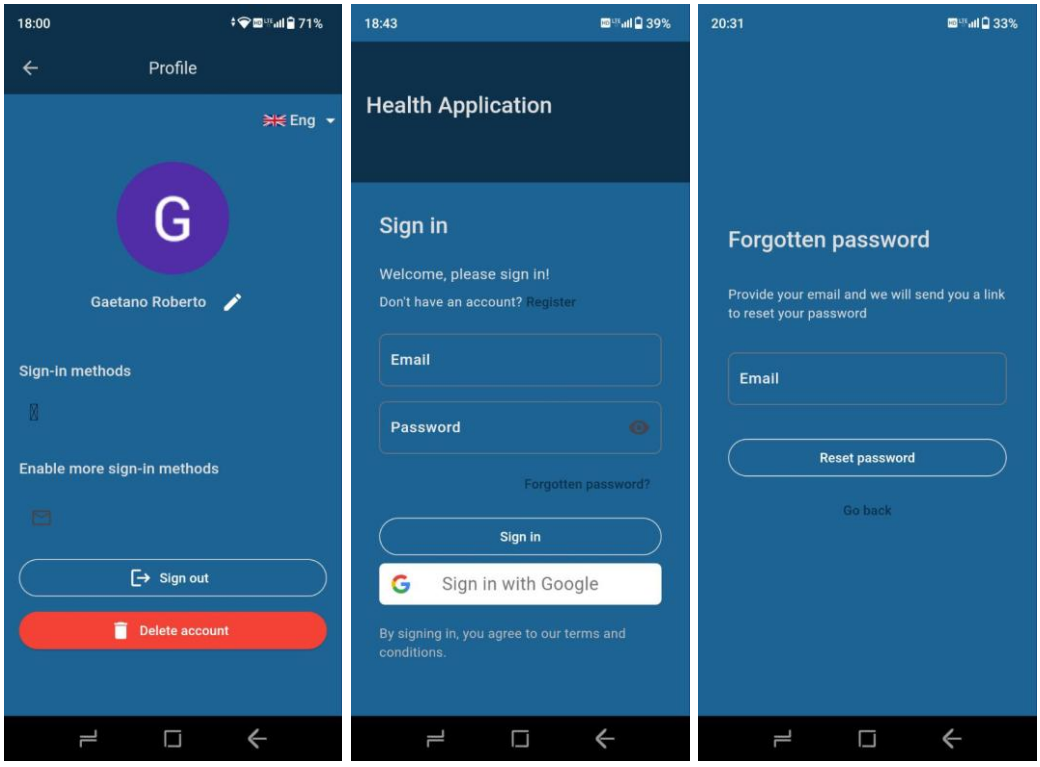


Figure 40: Profile – Sign in

4.3 APPLICATION SCENARIO: ADMIN

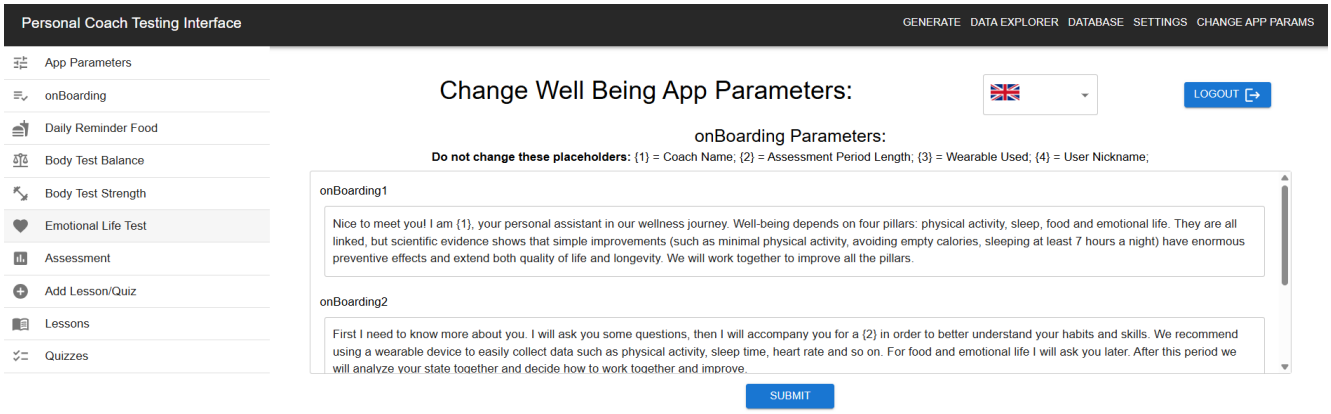


Figure 41: Management Tool Interface

The third and final scenario shows the admin management tool interface (Figure 41). A screen prior to this allows admins to log in by proving to be an admin via email and password. After logging in they can decide to change the application parameters, or the English and Italian texts used in the app. The currently existing parameters are 2: the name of the coach (the artificial intelligence that "reads" the user's data, analyzes them and produces recommendations) and the assessment period (which indicates the evaluation period of the user's status). In addition, on any screen there is a dropdown that allows them to switch texts from one language to another and a logout button. As shown in the figure, it is possible to use placeholders that act as parameters in texts: for example, by inserting in a text {1} the name of the coach chosen at that moment will appear in the app. There are four placeholders: coach name, duration of the assessment period, wearable device used and user nickname. The last two cannot be changed by the admin but are fetched based on the information entered by each user. On the left of the screen, instead, there is a navigation menu that allows admins to modify the interface based on what they want to change. In particular, admins can edit app parameters, onboarding screen texts, lessons and quizzes, text within notifications and related screens of Daily Reminder Food, Body Test Balance, Body Test Strength, Emotional Life Test and Assessment or they can insert new lessons with related quizzes in Italian and English. Editing a text in these forms and pressing the submit button produces an edit in the corresponding collection of the Cloud Firestore database.

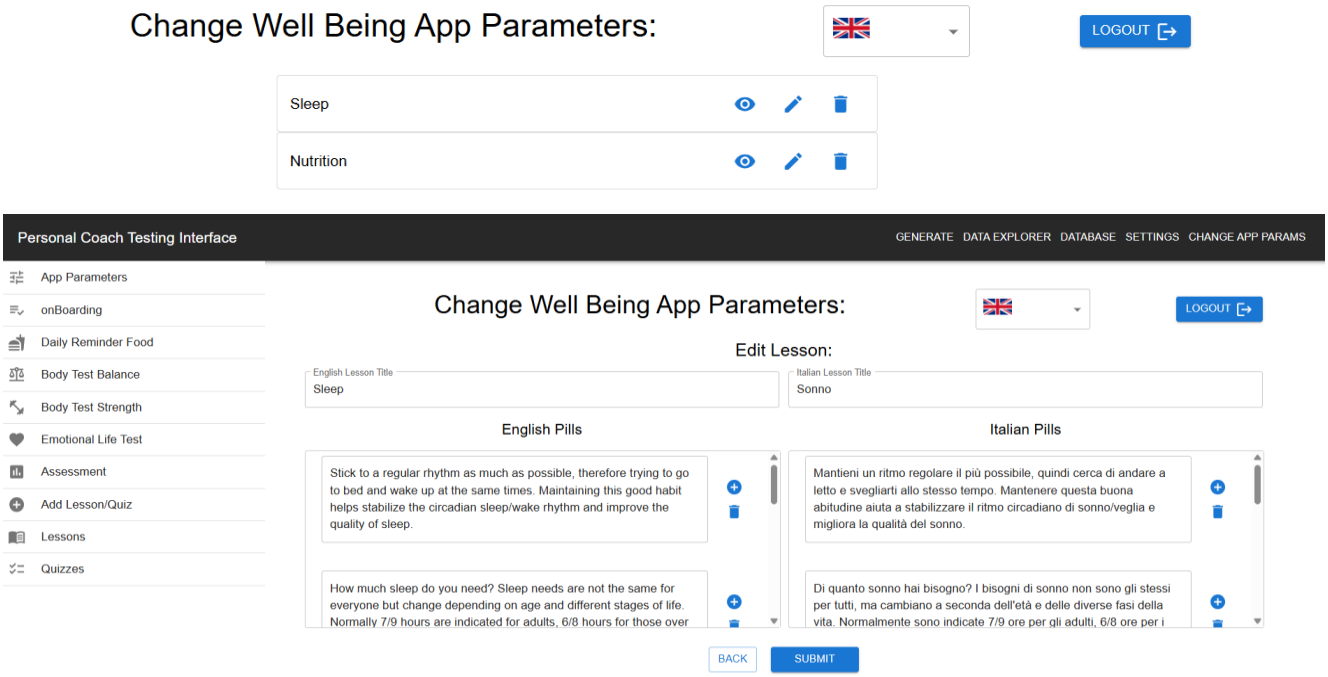


Figure 42: List Lessons - Edit Lesson

Clicking on the Lessons button displays a list of the existing lessons. For each of these it is possible to see the knowledge pills in detail, modify the lesson or cancel it. When a lesson is canceled, the corresponding quizzes are also deleted, and the cancellation takes place for both languages. By going into more detail in editing a lesson, it is possible to change the title in English, the title in Italian, the knowledge pills in English and the knowledge pills in Italian. The same applies to changing quiz questions (Figure 43). The structure of the quizzes requires that there is at least one question, each question has 4 multiple answers and only one correct answer. In both lessons and quizzes screens is present a validation of the fields, with consistency checks (a quiz must have a correct answer and no field can be empty).

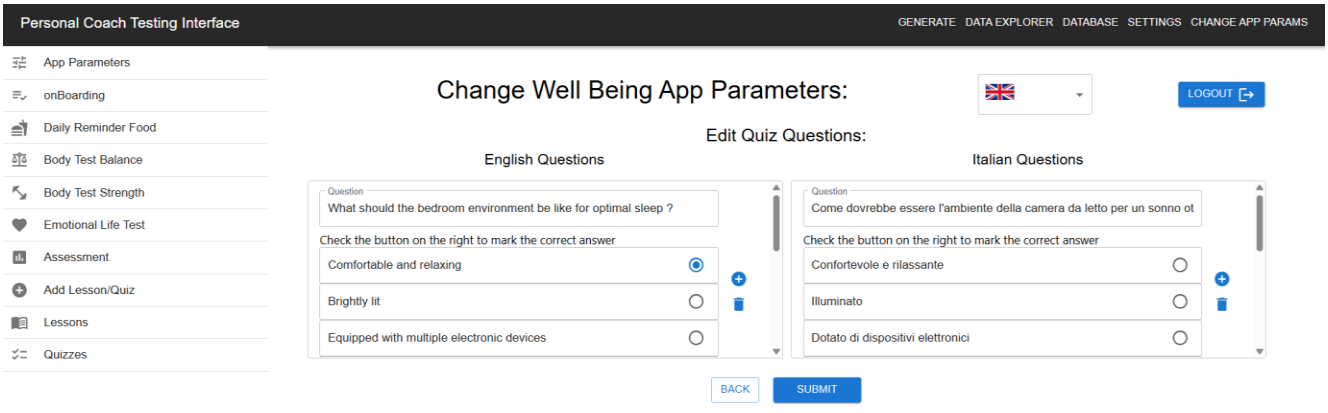


Figure 43: Management Tool – Quizzes

4.4 CONCLUSIONS AND FUTURE DEVELOPMENTS

The Well-being application is a mobile application that manages to provide a detailed plan on the state of health and well-being of users. In addition, it succeeds in its intention to educate users with advice and lessons on user health. The software reflects the initial specifications, succeeding in bridging the lack of health awareness and creating healthy habits with the aim of improving health. Moreover, its design provides all the information needed to compile a health report in a single application. From a functional point of view, further changes could be made that would enrich the application, such as the inclusion of more lessons and quizzes, translation into more languages and training of the LLM model to provide targeted recommendations to the user. A further future development could be to extend the application to support not only Android devices but also iOS, as its design foresaw this. Finally, the implementation of a system that periodically sends reports to the user's doctor would bring many benefits and could reduce the development of latent diseases.

Bibliography

[1] L. Fontana, The Path to Longevity: How to reach 100 with the health and Stamina of 40-year-old.

[2] "Health | Flutter package," [Online]. Available: https://pub.dev/packages/health.

[3] "health.com," [Online]. Available: https://www.health.com/10-foods-centenarians-eat-11682351.

[4] "Prevention," [Online]. Available: https://preventioncentre.org.au/about-prevention/what-is-prevention/.

[5] "University of Rochester Medical Center Rochester, NY," [Online]. Available: https://www.urmc.rochester.edu/senior-health/common-issues/top-ten.

[6] "Three Tier Architecture - IBM," [Online]. Available: https://www.ibm.com/it-it/topics/three-tier-architecture..

[7] "IBM Benefits," [Online]. Available: https://www.ibm.com/think/topics/three-tier-architecture.

[8] "Flutter," [Online]. Available: https://flutter.dev/.

[9] "Flutter," [Online]. Available: https://docs.flutter.dev/tools/hot-reload?.

[10] "Dart," [Online]. Available: https://dart.dev/.

[11] "Dart - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Dart_(programming_language).

[12] "Android Studio," [Online]. Available: https://developer.android.com/studio?hl=en.

[13] "Gradle Build - Android," [Online]. Available: https://developer.android.com/build/gradle-build-overview?hl=it.

[14] "YAML - IBM," [Online]. Available: https://www.ibm.com/it-it/topics/yaml.

[15] "Firebase," [Online]. Available: https://firebase.google.com/.

[16] "Health Connect - Google Documentation," [Online]. Available: https://support.google.com/fit/answer/12830119?hl=it.

[17] "Firestore," [Online]. Available: https://cloud.google.com/firestore/docs/overview?hl=it.

[18] "Data Model - Firestore - Google Cloud," [Online]. Available: https://cloud.google.com/firestore/docs/data-model.

[19] "Cloud Storage for Firebase," [Online]. Available: https://firebase.google.com/docs/storage?hl=en.

[20] "React," [Online]. Available: https://react.dev/.

[21] "Wikipedia - JavaScript," [Online]. Available: https://en.wikipedia.org/wiki/JavaScript#.

[22] "GitLab," [Online]. Available: https://about.gitlab.com/.

[23] Karim, Weisz and Rehman, "International positive and negative affect schedule short-form (I-PANAS-SF)," 2011.

[24] "Wikipedia - Oxygen Saturation," [Online]. Available: https://en.wikipedia.org/wiki/Oxygen_saturation_(medicine).

[25] "Wikipedia - Respiratory Rate," [Online]. Available: https://en.wikipedia.org/wiki/Respiratory_rate.

[26] "Harvard - Heart Rate," [Online]. Available: https://www.health.harvard.edu/heart-health/what-your-heart-rate-is-telling-you.

[27] "National Library of Medicine - Heart Rate Variability," [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC5624990/.

[28] "Cardio Mood," [Online]. Available: https://cardiomood.com/2023/06/17/what-is-a-good-hrv-by-age/.

Index of Figures

Ringraziamenti

Desidero esprimere la mia più profonda gratitudine a tutte le persone che hanno contribuito alla realizzazione di questa tesi e che mi hanno supportato durante il mio percorso accademico.

Ringrazio il mio relatore, il prof. Maurizio Morisio, per la sua disponibilità e per il suo supporto durante lo sviluppo dell'applicazione e la stesura della tesi.

Un ringraziamento speciale va alla mia famiglia — Angelapia, Mimmo e Grazia — per il loro sostegno, la pazienza, la comprensione e l'affetto dimostrati nei momenti più difficili. Senza il loro costante incoraggiamento e i loro sacrifici, non avrei mai raggiunto questo traguardo.

Ringrazio i miei nonni e i miei zii per tutto il sostegno. In particolare, voglio ringraziare mio nonno Gaetano, che ci ha creduto prima che ci credessi io.

Voglio ringraziare i miei amici di SGAP, che mi hanno permesso di vivere la vita universitaria con la massima leggerezza. Anche se ognuno ha preso la propria strada, so che insieme resteremo sempre una famiglia: un posto sicuro da chiamare casa, dove condividere le gioie ma anche i momenti difficili.

Un pensiero speciale va anche ai miei amici e colleghi universitari, che hanno reso questo viaggio più piacevole e ricco di momenti indimenticabili, condividendo con me gioie e difficoltà. In particolare, ringrazio:

- *Domenico TruvellaGPT, compagno di avventure universitarie fin dal primo giorno. Abbiamo seguito i corsi insieme e affrontato progetti assurdi fino a tarda notte, condividendo sfide e soddisfazioni. Abbiamo lavorato fianco a fianco anche per il progetto di tesi e ora, dopo tutto questo percorso, arriviamo insieme al traguardo della laurea. Non potevo desiderare un compagno di viaggio migliore.*
- *Ale Costa la Queen, per essere stato un coinquilino non proprio impeccabile, ma un amico fantastico. Sei riuscito a gestire me e Domenico insieme, e non è cosa da poco. Grazie per le risate, le serate spensierate e i momenti divertenti.*
- *Davide DaveBreaks, per aver reso il CSS il mio peggior nemico. Grazie a te gli incubi mi perseguitano, ricordandomi di aggiustare il padding di 0.24 mm. Scherzi a parte, grazie soprattutto per avermi portato quel giorno in palestra, se sono quello che sono oggi è anche grazie a te.*
- *Giorgione e Giuseppe Loacker, per la loro compagnia in aula studio e durante le lezioni, rendendo il tutto meno pesante e più sopportabile.*

Infine, voglio ringraziare tutte le persone che, direttamente o indirettamente, hanno contribuito alla realizzazione di questo progetto e alla mia crescita personale.

Grazie di cuore a tutti.
Gaetano.