Data Security Approach in Remote Healthcare Monitoring

Dragoș NICOLAU[1], Ovidiu BICA[1], Lidia BĂJENARU[1,2]
[1]National Institute for Research and Development in Informatics - ICI Bucharest
[2]University Politehnica of Bucharest
dragos.nicolau@ici.ro, ovidiu.bica@ici.ro, lidia.bajenaru@ici.ro

Abstract: In the context of an increased need to develop home care systems such as health monitoring systems, this paper presents an innovative system, namely Ro-SmartAgeing system. It represents a concrete, integrative, telemedicine solution, based on the collection, remote storage, and real-time processing of medical data. The Ro-SmartAgeing system reveals the potential of the emerging technologies, the Internet of Things, cloud computing, Big Data, and predictive analytics to induce a positive transformation of healthcare services offered to older people. This paper also focuses on the presentation of two fundamental security mechanisms, which are necessary in telemedicine, and which were employed in order to implement a secure flow of medical data within the Ro-SmartAgeing Integrated Information System.
Keywords: Security of Internet of Things (IoT)-based systems, older adult, health monitoring systems, Flask, OAuth 2.0, RO-SmartAgeing system, JSON format.

INTRODUCTION

As more and more older adults prefer to enjoy age-related care at home, proper remote monitoring of health parameters becomes an important support for both patients and their caregivers (Ianculescu, 2022).

The development of home care systems, such as e-Health monitoring systems, represents both a certainty and an increased necessity especially in the context of the COVID-19 pandemic. In addition to lessening the overall cost of healthcare systems, these technology solutions can potentially relieve the care burden for families and caregivers. They will allow older adult patients to stay at home as safely as possible.

The applications monitoring the current health state of the older adults, alongside cutting-edge ICTs (Information and Communication Technologies) pieces such as artificial intelligence and robotics have become a priority in the field for some time.

Many of them have addressed important aspects related to improving the quality of life and well-being of seniors by providing non-intrusive monitoring and support, enhancing professional healthcare, helping caregivers and providing intelligent care to senior people (Kubitschke, 2016).

The study of the literature has been helpful in selecting those applications that have integrated cutting-edge technologies, committed to non-intrusive monitoring, as it was demonstrated by the results of some European projects and AAL programs.

Within the BeyondSilos project, appropriate integrated care services (ICT) have been developed, namely the use of telemedicine, these services being based on care pathways that exceed the boundaries that usually separate health and social care (Meyer & Müller, 2014).The CareWell project focused on providing care and support for older adults who have complex health problems, are at high risk of hospitalization or home care, and require a range of high-level interventions due to their frailty and several chronic diseases. (Mateo-Abad, 2020)

The main goal of the GrowMeUp project (Martins et al., 2015) was to increase the number of years an older person can live independently and actively. It provides a robotic system with accessible services that is able to learn the needs and habits of the older adults over time and increase its capabilities in line with the older adults's decline in capabilities.

An EU project (AAL program), the vINCI project „Clinically-validated INtegrated Support for Assistive Care and Lifestyle Improvement: the Human Link - vINCI" developed an integrated IoT (Internet of Things) framework to provide non-intrusive monitoring and assistance to older people (Spinsante et al., 2023; Băjenaru et al., 2020). vINCI's objective is to improve the active aging of the older adults and to provide professional healthcare (Băjenaru et al., 2022). The result of the project is a vINCI app that can be downloaded on the phone or tablet by the older adults and an integrated technology platform that can be consulted by both the user and the caregiver to access the recorded information. This information can prevent or detect early symptoms typical of age-related deficiencies (Dobre et al., 2023).

In this perspective, this paper presents some aspects of the Non-invasive monitoring and health assessment of the older adults in a smart environment (RO-SmartAgeing) system. The RO-SmartAgeing system is an example of an IT system that has the potential to improve home healthcare outcomes for an older adult by providing personalized monitoring of health, environmental and lifestyle parameters in a smart environment. Last but not least, it can trigger emergency alerts and suggest preventive measures.

The project Ro-SmartAgeing demonstrates the potential of emerging technologies (such as the Internet of Things, cloud computing, Big Data, and predictive analytics) for the positive transformation of the way healthcare services are provided to older people and staff/health care providers (Alexandru et al., 2022).

The solutions proposed in this project include the use of wearable physiological sensors that collect at regular intervals the medical parameters of the monitored patient and transmit them to the cloud platform to be stored. The motion sensors intended to be used aim to monitor both the daily activities of the older adults, and their location and positioning in order to quickly or preventively identify potentially dangerous situations for physical integrity. Ambient sensors have the role of monitoring the environment surrounding the older adults in order to support healthy living conditions.

The personalized intelligent environment for an older adult developed in this project integrates physiological, motion, and environmental sensors, interconnected digital devices, and technology capable of providing services and support for an older adult's well-being, continuous and remote monitoring of his activities daily, health, and safety. The main purpose of the smart environment is to enable the older adults to continue their life with dignity in a familiar space while receiving the best health supervision, treatment, and physical support, but also data protection and security (Cîrnu, 2018; Vevera, 2018).

The Ro-SmartAgeing system represents a concrete, integrative, telemedicine solution, based on the collection, remote storage and real-time processing of medical data. This paper highlights the security mechanisms, necessary in telemedicine, that have been utilized to implement a secure flow of medical data within the Ro-SmartAgeing Integrated Informatics System (Alexandru et al., 2022). The general features of the non-invasive monitoring and health assessment smart environment, i.e. the Ro-SmartAgeing System, an innovative solution designed for improving home healthcare results (mainly for older adults) are presented.

After briefly presenting the utility and the functioning of the proposed System as a whole, this paper focuses on the functional traits of the above-mentioned security mechanisms, along with the context in which they act in order to provide a safe transfer of sensitive data to and from the storage server. Also, the paper presents the proposed sensors that collect at regular intervals the medical parameters of the monitored patient and transmit them to the cloud platform to be stored.

OVERVIEW OF THE RO-SMART AGEING SYSTEM

The functioning of the Ro-SmartAgeing system implies the existence of three data carriage paths: the sending of acquired raw medical data to the Withings server (accomplished by the medical devices), the retrieval of that medical data from the Withings server and the usage of data within a web application.

Acquisition of raw medical data. Sending it to the Withings server. It is primarily achieved by the means of an IoT, Withings-type medical sensor equipment. These devices collect medical data and instantly transmit it to a specialized Android application, running on a smartphone located in the vicinity; promptly, the application packs medical data as a Https request that is sent in real time to the Withings server. The functional sequence is presented below.

- On the Withings server, through a special web page, an user account will be created, subsequently identifiable by an e-mail address and a particular password. Under the authorization of this account, the user may then request an access token (a 40-digit hex string).
- On the smartphone offered by the Withings equipment, the “Health Mate” Android application will be installed from the Application Store. Once installed, the application will be launched.
- Just after the application has been started, a list of types of devices will be displayed when operating the “Device” button. From the list one may choose a medical device. The Bluetooth communication channel of the smartphone will be activated.
- The Bluetooth connector of the IoT medical device is also activated. The device will be located in the proximity of the smartphone and will be set on the patient.
- After the measurement has been performed, the value of the parameter will be conveyed over the secure Bluetooth protocol to the „Health Mate” application; the latter will encapsulate the code of the medical parameter, the measured value and the ID of the device in a HTTPS request to be sent to the Withings server; on the server, this triple information attached to the Request, along with the moment of arrival, will be inserted in the database in strict association with the Withings client account. The measurement can now be retrieved on the second path of information transfer.

Retrieving medical data from the Withings server. The operation is performed by a specialized application that, at a certain pace, sends secure HTTP requests (i.e. over HTTPS protocol) to the Withings server. Each request is issued by its own execution thread generated by an additional execution thread that bears the role of a dispatcher - monitor.

- The application, developed in Visual C++ (Goytia & González, 2014) and assisted by a library developed in C# (Price, 2022), is provided with a grid control, in which, in the beginning, necessary information from the local database is loaded in several rows, according to the same (recurring) pattern. Thus, each item (row) of the grid contains in its left extremity a checkbox used for starting/stopping the uniquely associated execution thread; in addition, each row will contain: its own order index, the ID and the name of its unique physiological parameter, the ID and the value of the associated token, the updating pace (the cadence at which the requests are sent to retrieve data - being set through the ad-hoc, dynamic editing box), the moment of the most recently retrieved data.

- Each request (of POST type) depends on an attached access token, that is refreshed (updated) by the storage server every 3 hours through an additional real-time request issued for this purpose by the running application. The updating process will run spontaneously (without disrupting the "motion" of the application) and will be immediately followed by retaining the new value in the memory of the application.
- All responses received from the storage server are packed in JSON format; after that they are immediately decoded and structured as SQL commands for inserting fresh data into the database.
- Each measurement (coming from a certain device) is associated with the most recent patient who used the respective device, this information being taken from the auxiliary table that automatically records the moment at which a certain medical device is associated to the current patient.
- The data retrieval application is provided with automatic regulation of the periodicity at which it issues requests if the Withings server detects an excessively abundant request torrent for a given time span. The adjustment is done in real time, without disturbing the operation.
- For storing all medical data newly obtained from the Withings server, a MySQL structured database is utilized (Frost, 2020). Immediately after being decoded from the JSON format (as provided by the Withings server), medical data is sent to a Stored SQL procedure where it is associated with the proper patient and inserted into the measurement table. Decoding data from the JSON format and sending it to the SQL procedure is performed by functions developed in C#.Net. In case of medical values beyond usual limits, a new record bearing the value that deserves increased attention will be immediately inserted in the alerts table.
- The application allows the manual saving of the new pace values to the Database, as a batch.

Using data in a web application. This last section of the handling of measured data consists in processing and visualizing in a secure web application all the medical values pertaining to every patient (historical values, interpreted results, alerts etc.). Figure 1 illustrates the functional diagram of the Ro-SmartAgeing system.

Refreshing (updating) the code (token) for data accessing. Every 3 hours, the current code - with which medical data (in general, the latest) is obtained for an ensemble of physiological features (parameters) - loses its validity (expires), so the Withings server sends a warning response, accordingly.

The data recovery application is designed so as to achieve a new access code (or several, if applicable), immediately after being notified on the expiration of the current one, without interrupting the running of the threads that periodically request fresh data.

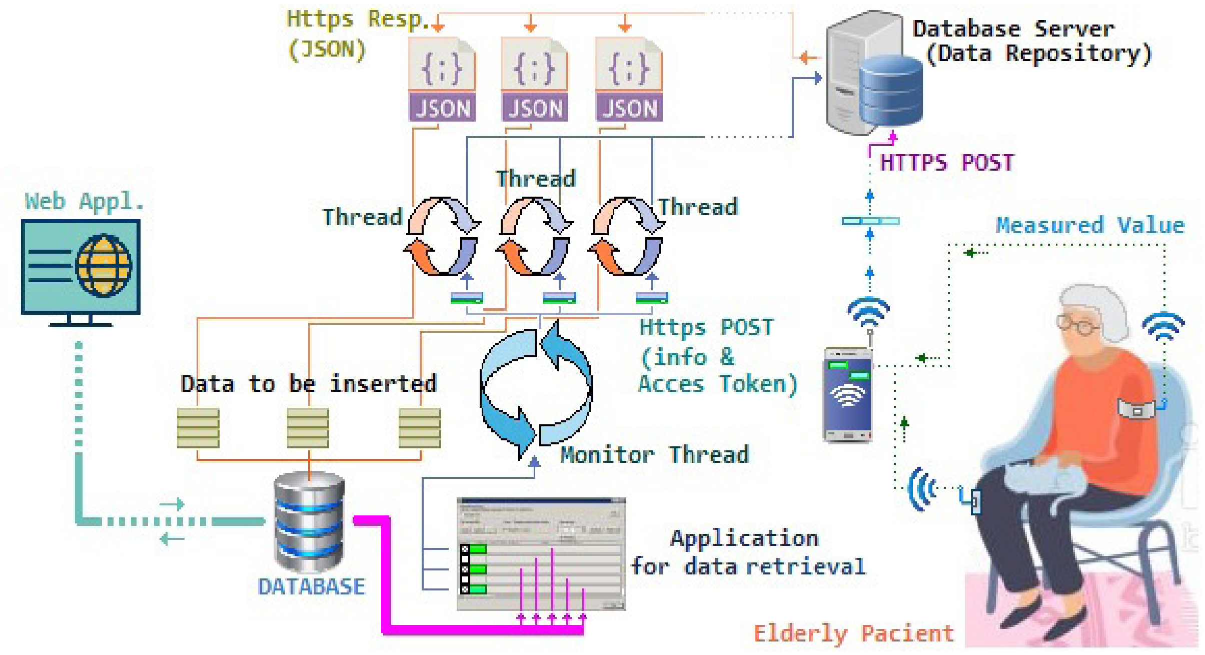


Figure 1. *Functional scheme of the Ro-SmartAgeing System (Anon, Rovicare, Accessed February 2023)*

THE SECURITY MECHANISMS USED WITHIN RO-SMARTAGEING SYSTEM

The first security mechanism

As medical data represents sensitive information, it is principally important that data recovery be carefully restricted solely to authorized applications. For this purpose, the Withings server has been designed to accept only requests that bear a certain identification token (i.e. access token). For further enhancement of the safety of medical data retrieval, every access token must be refreshed every 3 hours. Thus, from a purely technical point of view, the automatic update is done as follows:

- One of the execution threads will receive the notification that the validity of the code (token) has just expired. Immediately, this thread (subsequently playing the temporary role of "guard"), will place in the queue of each congener thread a temporary stop message, at the reception of which all of them will enter the "break" regime (herein, congener = "brother" threads refer to all different threads that use, at some moment, exactly the same access token).
- The thread that randomly happened to be entrusted with the role of "guard" will quit sending data retrieval requests and instead will immediately issue a request for a new code (token).
- Immediately after it is received the new code will be (a) sent to all congener threads, (b) memorized in the database and (c) recorded in the grid -type control in every row associated with the given token.
- The "guard" thread will send to all congener threads a message for resuming the normal data retrieval activity (for each physiological parameter uniquely associated with a certain thread), on the basis of the fresh access code.
- If, in order to obtain a new access code, an additional updating code is required, the "guard" thread will stop the functioning of all congener threads, displaying a message box inside of which a warning informs the administrator that an additional manual update involving the Flask servlet is necessary for acquiring the new access code.

The second security mechanism

Flask Microframework

Flask is a small web framework, which offers useful tools and functionalities that facilitate the creation of web applications with the help of Python.

Flask contains two major components, Werkzeug and Jinja2. While Werkzeug is responsible for assuring routing and transmitting requests to web applications or frameworks written in Python programming language (the web server gateway interface, Flask uses Jinja2 as a modeling engine. Natively, Flask does not support Access to databases, authentication of users or any other high-level utilities, but provides support for integrating all these features, which makes Flask one of the most effective microframework for the development of web applications and services (Relan, 2019).

Flask is a WSGI (Web Server Gateway Interface). WSGI is a call interface for web servers, which allows the transmission of requests to web applications or Python programming language. The current version of WSGI 1.0.1 is specified in Python Enhancement Proposals (PEPs). WSGI has been created as an independent interface for implementation between web servers and web applications or workshops to promote a common basis for the development of portable web applications (Taneja & Gupta, 2014).

Flask, like all Python libraries, can be installed with the help of Python Package Index (PPI) and it is extremely easy to configure it and to develop applications in it.

REST APIs using Flask generally use MySQL as their backend database. As it was mentioned, Flask does not come with native support for database access, and to fill this gap, it uses a Flask extension called Flask-SQLAlchemy, which adds SQLAlchemy support to Flask. SQLAlchemy is essentially a SQL Python toolkit and Object Relational Mapper that gives developers the full power and flexibility of the SQL language. SQLAlchemy provides full support for enterprise-level design patterns and is designed for high-performance database access while maintaining efficiency and ease of use (Anon SQLAlchemy, 2023).

OAuth 2.0 Protocol

OAuth 2.0 is a large-scale protocol, which offers a safe way for customers to access protected resources, which need authorization without sharing the customer's real data/beliefs. Before OAuth 2.0, there were many ways of accessing protected resources the only problem being that in none of them an authorization server was used.

OAuth 2.0 works in such a way that the real data of the customer is not sent back (such as the password) instead is sent back in response a token. The token is a string that contains data such as user identity. The client then uses this token in every request he sends to the private service. Authentication services validate and verify the identity of the application. In this way, there is no private data traveling around the Internet, which makes OAuth a safe authorization.

Because authorization protocol works as a means of granting third-party applications a limited access to these protected resources or HTTP services on behalf of the owner of a resource, it is called delegation protocol (Richer & Sanso, 2017).

The OAuth 2.0 Protocol can be defined by four concepts (Jones & Hardt, 2012):

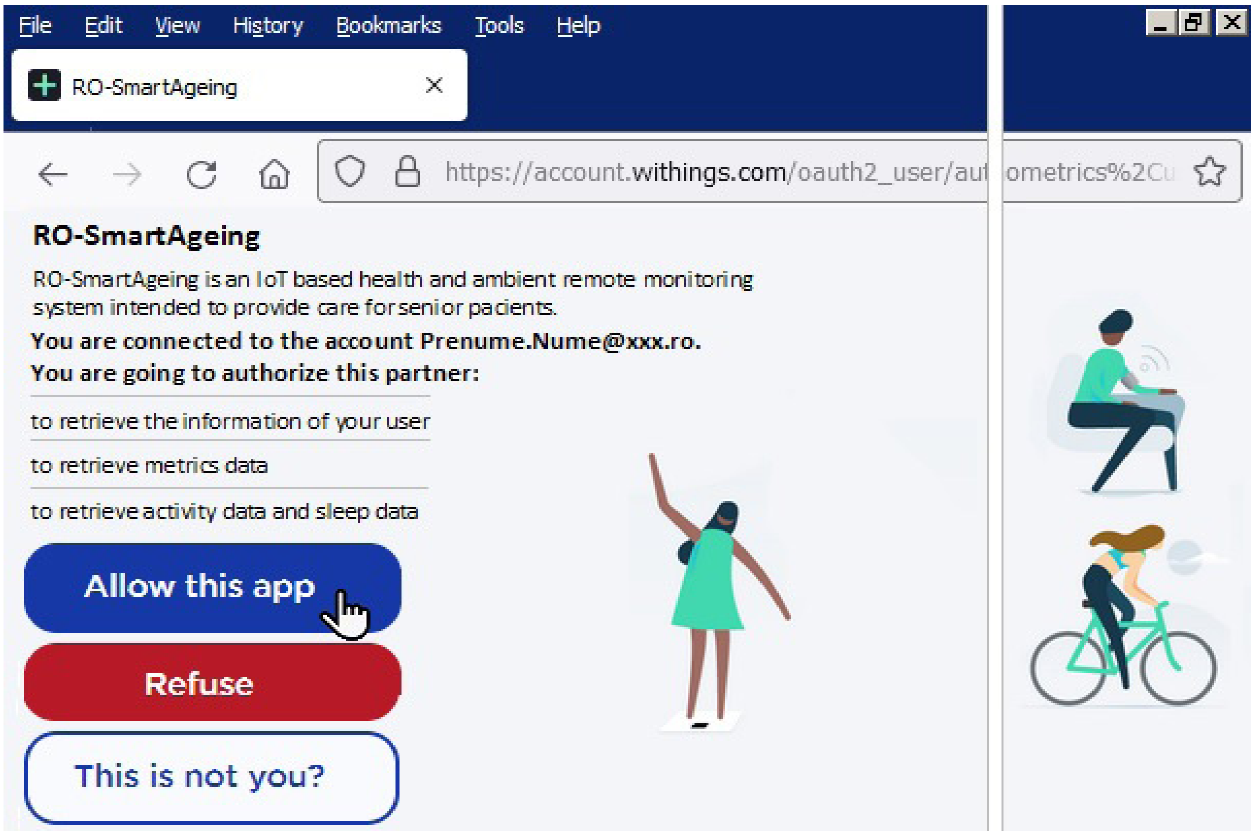
- Resource owner: the organization that provides access to shared resources.
- Resource Server: Server that hosts protected resources and deals with requests regarding these resources through access tokens.
- Customer: Application that sends requests for accessions to protected resources, once the authorization has been granted.
- Authorization server: Server that provides access tokens to a customer, who has obtained the authorization.

Implementation of the Flask Environment within the Ro-SmartAgeing System

The portal dedicated to Withings developers allows the creation of applications that use Withings devices and data stored in the Withings databases (Anon Withings Developer Portal, March 2023). Access to these data is grantes only to users who have given their agreement beforehand. These solutions include the use of Withings devices, Withings API, Withings Mobile SDK, Withings Logistics and more. Within the Ro-SmartAgeing project, the variant of using public API (made available by Withings) was chosen. The Flask framework is installed on a Ubuntu virtual machine (Anon Ubuntu, 2023) running in the IaaS infrastructure. The connection between the Ubuntu virtual machine and the Withings Cloud is made using the OAuth 2.0 Protocol (Anon Withings Authorization Oauth 2.0, 2023).

Within the project, a small web server developed in Python is used; it connects to the Withings developer API using the Oauth 2.0 protocol. This server is installed with the help of a virtual environment in Python 3 (Virtual Environment).

Figure 2 illustrates the moment in which the Withings client (Prenume.Nume@xxx.ro) is finally enabled to authorize the Withings server to issue useful pieces of information such as a new authorization token, or metrics (info categories that are permitted to be retrieved). The application designed to retrieve medical data will then use the authorization token to request a fresh access token; the latter will be then utilized to request further possible fresh medical data. To emphasize the role of Figure 3, it must be mentioned that it represents the dashboard sent by the Withings server after the successful authentication had just been completed. This dashboard is the necessary place to to perform the manual achievement of the above-mentioned infos by clicking the navy blue button.



RO-SmartAgeing

RO-SmartAgeing is an IoT based health and ambient remote monitoring system intended to provide care for senior pacients.

You are connected to the account Prenume.Nume@xxx.ro.

You are going to authorize this partner:

to retrieve the information of your user

to retrieve metrics data

to retrieve activity data and sleep data

Figure 2. *Dashboard for authentication and access permissions, Ro-SmartAgeing system (Anon Withings Authorization Oauth2, 2023)*

WITHINGS MEDICAL DEVICES USED IN THE RO-SMARTAGEING SYSTEM

Withings devices were chosen due to their potential to process and store medical data without having to subscribe to the Withings Cloud. Additionally, an API for communication and information transmission is provided in order to integrate a third-party application, such as the Ro-SmartAgeing system and it can also be used for systems with up to 5,000 users.With the help of APIs made available through the Whitings developer portal (Anon Withings Developer Portal, Accessed February 2023), applications can be created using Withings devices and the variety of health data they record: weight, fat, body temperature, sleep activity, blood pressure, heart rate, ECG, PWV and more. Listed below are some of the Whitings devices used in this project, which are also illustrated in Figure 3.

- *Tensiometer* - WITHINGS BMP CORE BLOOD Pressure Monitor. The measured parameters are: systolic blood pressure; diastolic blood pressure; pulse; Electrocardiogram (ECG).
- *Body Scale* - Withings Full Body Scale. The measured parameters are: weight; body mass index (BMI); percentage of body fat; percentage of body water; bone mass; muscles.
- *Sleep Monitoring device* - Withings Sleep Analyzer. The measured parameters are: sleep duration; sleep score; Sleep apnea.
- *Intelligent watch* - Smartwatch Withings Move ECG (Figure 2); The measured parameters are: pulse; Electrocardiogram (ECG); duration of activity performed; Location of activity carried out; calories burnt during physical activity.
- *Thermometer* - the measured parameter is temperature.

Withings devices have so far only been tested at laboratory level and could be tested or used for the purpose of home health monitoring for various individuals by disseminating and implementing the system within specific institutions and beyond.

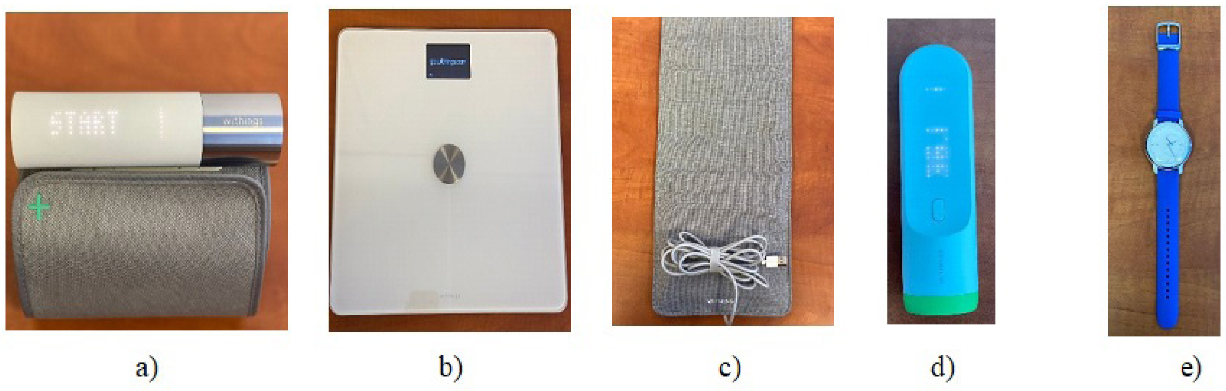


Figure 3. *a) Tensiometer - WITHINGS BMP CORE BLOOD Pressure Monitor, b) Body Scale - Withings Full Body Scale, c) Sleep Monitoring device - Withings Sleep Analyzer, d) Thermometer, e) Intelligent watch - Smartwatch Withings Move ECG*

CONCLUSION

As the use of IT health monitoring systems becomes more and more widespread, the implementation of security mechanisms proves to be extremely necessary. The specific case study presented in this paper is centered on RO-Smart Ageing, an IT system that represents a plain telemedicine solution, based on the collection, remote storage and real-time processing of medical data.

The paper also focuses on presenting two important security mechanisms used by RO-SmartAgeing system: the first pertains to securing access to medical data (i.e. it manages the spawning of an access token), the second is related to securing the access token itself (i.e. it manages the spawning of a refresh token necessary for generating the access token that protects access to medical data). Thus, the access token is randomly generated by the Withings storage server every 3 hours, whereas the refresh token is normally generated by the Withings storage server once every year. However, if the Withings server detects suspicious inconsistency in relation to the IP Address from which requests are poured, then one must expect that refreshing the refresh token be imposed much more frequently than once every year.

Concerning the first security mechanism, this paper emphasizes that the C++/C# application described in Chapter 1 ensures that obtaining every new access token be a fully automatic process. Concerning the supplementary (second) security mechanism (the one that manages the spawning of a refresh token, as it was mentioned in the previous paragraph), this paper highlights the usage of the Flask microframework, a coding solution that has been preferred due to its simplicity and flexibility in operation; as it was created based on a generous documentation, Flask appears suitable for a fast and safe development of various RESTful APIs.

This research also highlights the OAuth 2.0, a standard protocol used for authentication and authorization in the context of developing REST APIs and securing web applications, which can be considered a reliable security protocol. Authentication involves the process of verifying who the authenticated user is, but it does not involve granting access to all resources, this is where authorization comes in, which consists in authorizing an authenticated user to maintain control over the resources he/she has access by using a control list.

The RO-SmartAgeing system reveals the potential of the emerging technologies to induce a positive transformation of healthcare services provided to older adults. After briefly presenting the utility and the functioning of the System as a whole, the paper focuses on the functional traits of the above-mentioned security mechanisms, along with the context in which they act in order to provide a safe transfer of sensitive data to and from the storage server. The proposed sensors that collect the medical parameters of the monitored patient at regular intervals and transmit them to the cloud platform to be stored, are also presented.

ACKNOWLEDGMENT

This work was carried out within the project “Noninvasive monitoring and health assessment of the older adults in a smart environment (RO-Smart Ageing)” (funded by the Ministry of Research, Innovation and Digitalization) and the project PN-III AAL2017-63-vINCI - “vINCI: Clinically-validated INtegrated Support for Assistive Care and Lifestyle Improvement: the Human Link” (EU AAL 2017 Programme and UEFISCDI Romania).

REFERENCE LIST

Alexandru, A., Ianculescu, M., Giura, I. E. & Pop, F. (2022) Managing Cybersecurity Threats for Seniors' Digital Needs Using Age-Friendly Remote Healthcare Monitoring Model. In: *Proceedings of The 10th Edition of IEEE International Conference on e-Health and Bioengineering, EHB 2022,* November 17-18, 2022, Iasi, Romania. pp. 1-4.

Băjenaru, L., Balog, A., Dobre, C., Drăghici, R. & Prada, G. I. (2022) Latent profile analysis for quality of life in older patients. *BMC Geriatrics.* 22, 848. doi: 10.1186/s12877-022-03518-1.

Băjenaru, L., Marinescu, I. A., Dobre, C. & Tomescu, M. (2020) Integrated support based on IoT technologies for improving the elderly's quality of life. *Romanian Journal of Information Technology and Automatic Control.* 30(2), 53-66. doi: 10.33436/v30i2y202005.

Cîrnu, C. E., Rotună, C. I., Vevera, A. V. & Boncea, R. (2018) Measures to Mitigate Cybersecurity Risks and Vulnerabilities in Service-Oriented Architecture. *Studies in Informatics and Control.* 27(3), 359-368. doi: 10.24846/v27i3y201811.

Dobre, C., Băjenaru, L., Drăghici, R., Prada, G.-I., Balog, A. & Herghelegiu, A.M. (2023) Sustainable Health-Related Quality of Life in Older Adults as Supported by the vINCI Technology. *Sensors,* 23(4), 2287.

Frost, O. R. (2020) *PHP & MYSQL Guida completa alla programmazione web lato server e database SQL.* Edizione Stampata in Italiano, Independently published.

Goytia, J. L. L. & González, A. G. (2014) *Programación orientada a objetos con C++ y Java.* Ciudad de México, Grupo Editorial Patria.

Ianculescu, M., Paraschiv, E-A. & Alexandru, A. (2022) Addressing Mild Cognitive Impairment and Boosting Wellness for the Elderly through Personalized Remote Monitoring. *Healthcare.* 10(7), 1214. doi: 10.3390/healthcare10071214.

Jones, M. & Hardt, D. (2012) *The OAuth 2.0 Authorization Framework: Bearer Token Usage.* https://tools.ietf.org/html/rfc6750 [Accessed 15th January 2023].

Kubitschke, L., Meyer, I., Müller, S., Stellato, K., & Di Lenarda, A. (2016) Digital Technologies as a Catalyst for Change towards Integrated Care Delivery: Hype or Reality?. *International Journal of Reliable and Quality E-Healthcare (IJRQEH).* 5(2), 31-49. doi:10.4018/IJRQEH.2016040102.

Martins, G.S., Santos, L. & Dias, J. (2015). The GrowMeUp Project and the Applicability of Action Recognition Techniques. In: *Third Workshop on Recognition and Action for Scene Understanding,* REACTS 2015, September 5, 2015, Valletta, Malta. Ruiz de Aloza.

Mateo-Abad, M., Fullaondo, A., Merino, M., Gris, S., Marchet, F., Avolio, F., Graps, E., Ravic, M., Kovac, M., Benkovic, V., Stevanovic, R., Zwiefka, A., Davies, D., Mancin, S., Forestiero, A., Stafylas, P., Hurtado, M., d´Angelantonio, M., Daugbjerg, S., Pedersen, C.D., Hammerschmidt, R., Stroetmann, V., Azkargorta, L., Giné, A., Verdoy, D., Soto-Gordoa, M., Mora, J., Mar, J., Vergara, I., de Manuel Keenoy, E. & CareWell project group. (2020) Impact Assessment of an Innovative Integrated Care Model for Older Complex Patients with Multimorbidity: The CareWell Project. *International Journal of Integrated Care.* 20(2), 8. doi: 10.5334/ijic.4711.

Meyer, I. & Müller S. (2014) D1.2 *Pilot level pathways and integration infrastructure.* http://beyondsilos.eu/fileadmin/beyondsilos/d1.2_v1.0_beyondsilos_pilot_level_pathways_and_integration_infrastructure.pdf [Accessed 15th February 2023].

Price, M. J. (2022). *C# 11 and .NET 7 – Modern Cross-Platform Development Fundamentals: Start building websites and services with ASP.NET Core 7, Blazor, and EF Core 7.* 7th ed. Birmingham, Packt Publishing.

Relan, K. (2019). *Building REST APIs with Flask: Create Python Web Services with MySQL.* Apress.

Richer, J. & Sanso, A. (2017). *OAuth 2 in Action.* Shelter Island, Manning Publications.

Spinsante, S., Poli, A., Mongay Batalla, J., Krawiec, P., Dobre, C., Băjenaru, L., Mavromoustakis, C. X., Constantinou, C., Molan, G., Drăghici, R., Herghelegiu, A. M., Prada, G. I. & Gonzalez-Velez, H. (2023) Clinically-validated Technologies for Assisted Living The vINCI Project. *Journal of Ambient Intelligence and Humanized Computing.* 14, 2095-2116. doi:10.1007/s12652-021-03419-y.

Taneja, S. & Gupta, P. R. (2014) Python as a Tool for Web Server Application Development. I*nternational Journal of Information, Communication and Computing Technology.* 2(1), 77–83.

Vevera, A. V. & Albescu A. R. (2018) Human resource vs. cyber security. *Romanian Journal of Information Technology and Automatic Control.* 28(4), 67-74.

**OAuth 2.0. (2023) Oauth 2.0. https://oauth.net/2/ [Accessed 15 January 2023].

**Rovicare. (n.d.) *Care Transitions Software for Hospital at Home*. https://www.rovicare.com/who-we-serve/hospital-at-home [Accessed 15 January 2023].
**SQL Alchemy. (n.d.) *The Python SQL Toolkit and Object Relational Mapper*. https://www.sqlalchemy.org/ [Accessed 15 January 2023].
**Ubuntu. (2023) *Guide to Azure Cloud security with Ubuntu*. https://ubuntu.com/ [Accessed 15 January 2023].
**Withings. (2023) *Withings Developer Portal*. https://developer.withings.com/ [Accessed 15 January 2023].
**Withings Authorization. (n.d.) https://account.withings.com/oauth2_user/authorize2. fig. = "side_picture.png".
**Withings. (n.d.) *Withings developer documentation*. https://developer.withings.com/api-reference/[Accessed 15 January 2023].