Adaptive Dashboard for IoT Environments: Application for Senior Residences

Bessam Abdulrazak[1,2](✉) and Amin Rezaei[1]

[1] Ambient Intelligence Laboratory (AMI-Lab), Université de Sherbrooke, Département d'informatique, Faculté des Sciences, QC J1K 2R1 Sherbrooke, Canada
{bessam.abdulrazak,amin.rezaei}@usherbrooke.ca

[2] Centre de recherche sur le vieillissement du CIUSSS de l'Estrie – CHUS, Sherbrooke, Canada

Abstract. Dashboards are powerful electronic tools that can provide actionable insights for healthcare professionals, especially in support of the increasing senior population. With advancements in technology and IoT infrastructure, remote patient monitoring has become a feasible option for healthcare professionals through dashboards. To best serve the diverse needs of healthcare professionals, dashboards should be tailored for each user, considering their roles, interests, and priorities. In this study we proposed AMI-Dash, a solution allows for dynamic design and information visualization to address the diversity in needs and priorities among different dashboard users while maintaining a high-level of performance, as evaluated through several technical aspects.

Keywords: Dashboards · Internet of Things (IoT) · Remote Health Monitoring (RHM) · Healthy Aging · Senior residences · Deployment

1 Introduction

Dashboards can be a useful tool to deliver care to older adults while providing real-time health information to healthcare professionals. The advancement in the Internet of Things (IoT) and sensor technology allows for constant monitoring of seniors' health conditions, which is beneficial in reducing the work burden of healthcare professionals. This is especially important when we consider the shortage of healthcare professionals and the swelling population of older adults, leading to more pressure on the healthcare system and inevitable need for supportive solutions.

The population of older adults is increasing in numerous developed countries, including in Canada, where nearly a third of its population belongs to the baby boom generation [1]. This is putting more pressure on the already stretched healthcare system as the demand for healthcare and life aid services for baby boomers will continue to increase[2]. Remote Health Monitoring (RHM) is a new modern approach that has the potential to help healthcare professionals in Senior Residences by continuously monitoring individuals' health factors and transmitting them to healthcare professionals [3]. This can reduce the burden on healthcare professionals and the pressure on the healthcare system, as well as reduce costs and improve the quality of life for older adults. RHM systems generate heavy streams of data, but it is important to deliver this information to healthcare professionals in an understandable manner. Dashboards are powerful interfaces that can use RHM and act as a medium to deliver meaningful information to healthcare professionals and enable them to make informed and timely decisions. However, there are several problems associated with dashboard development and deployment in healthcare, including limited capabilities to respond to diverse needs, preferences, and roles in healthcare, the complexity of using multiple independent dashboards, and the high cost of dashboard development [4, 5]. The design of dashboards must constantly adapt to changing needs and be tailored to the specific preferences and roles of healthcare professionals. The large number of dashboards can lead to islands of knowledge and make it difficult for healthcare professionals to gain insights they need. Additionally, existing development approaches make it difficult to expand solutions to other healthcare centers, increasing costs. Therefore, the development of dashboards requires a deep understanding of real needs of health professionals and the healthcare workflow.

We aim in this study to achieve a tailored solution for a dashboard in the context of healthy aging and senior residences. We are interested in reducing the information load and mental workload of healthcare professionals and improving their effectiveness by providing them with tailored dashboards. Therefore, the overall aim of this research is to provide a cost-effective and efficient monitoring mechanism to reduce the workload of healthcare professionals. More specifically, the presented study aims to provide personalized dashboards integrated with the AMI-Lab platform [6], a complete IoT infrastructure for dynamic and quick deployment of IoT systems which is already being used in several research projects to follow up remotely older adult status.

The rest of the paper is organized as follows: Sect. 2 introduces the diverse approaches to handling tailored dashboards. Section 3 presents the proposed approach, the implementation as well as the results of the experiments and evaluations conducted. We finally outline the conclusion in Sect. 4.

2 Literature Review

There are several studies on building tailored dashboards with diverse approaches to manage the requirements of each user and how to handle tailored dashboards accordingly. We can categorize the key strategies into three: a) Facilitate source code generation, b) Connecting to existing BI tools and c) Personalized solutions.

a) Facilitate Source Code Generation: Achieving tailored solutions in code level was one of the approaches taken in the literature. A number of studies took software production line in the literature [7]. This approach uses code templates to achieve dynamic creation of dashboards. User requirements and usage context is fed to the generation engine to generate the source code of the intended dashboard. Similarly, A model-driven approach developed by Palpanas *et al.* [8] follows the same approach using defined models. The difference in this study is that information provided as input for their system is fine-grained based on dashboard structure and it asks to provide more details including navigation, access controls and templates. Based on this provided information, a complete functional code for the dashboard is generated. In all mentioned works that use code approach, the developers have access to final source code after generation of the dashboard and they can modify it to meet their desired customization. Therefore, this solution provides a high level of flexibility. On the other hand, this approach depends on the existence of developers and technical people, and it cannot be used by final users of the dashboard. Moreover, existence of source code means that the final solution should pass all steps required for deployment of a software, ranging from compiling the code to deploying it on the final production system. Nevertheless, none of the studies using this approach mentioned provided specific plans for deployment which can be a very resource consuming task if it is not performed properly.

b) Connecting to Existing BI Tools: Interestingly, a number of existing solutions used external BI tools for the final rendering of the dashboard. Tundor *et al.* [9] proposed declaration description of the dashboard, and the final output of their system is a configuration layout which can be used by other BI solutions to represent the designed dashboard. Similarly, Santos *et al.* [10] proposed a knowledge graph which uses an ontology approach to generate dashboards automatically for smart cities. They used API methods to communicate with other BI solutions and reduce the efforts required to create the dashboard. Although leaving the responsibility of rendering the final dashboard can dramatically reduce development time and cost, it has a number of clear drawbacks. The main drawback of this approach is that the software cannot work standalone, and the user needs to link the output of proposed solutions to another external tool to be able to visualize data. Moreover, not all BI tools support common forms of visualization. An element on the designed dashboard might not be able to render on certain BI tools and therefore they cannot be shown on the final dashboard.

c) Personalized Solutions: A number of studies aimed to personalize the dashboard to users based on their need. We witnessed two different perspectives to achieve this goal. One perspective was to perform this procedure automatically based on the behavior and usage of end users while they are using the dashboard. For instance, Belo *et al.* [11] aimed to restructure the dashboard elements based on information collected during end-user usage. The other perspective requires explicit definition of user needs and then tailoring dashboard based on declared needs. This idea can be seen in studies of Ines *et al.* [12] and Tundo *et al.* [9]. More specifically, Ines *et al.* [12] designed a questionnaire to extract the needs of the end user and use collected information later to provide personalized dashboard. Similarly, Tundo *et al.* [9] also need an explicit declaration of intended dashboard. In both perspectives, authors assumed that the dashboard has the required visualization elements to reply to the needs of users and the main goal is to make existing elements more accessible for users. They did not mention how they want to handle the changes in users' needs over time or how to provide different users with different data.

Overall, the existing solutions in literature provided limited flexibility in terms of design and supporting new visual elements, most of the proposed solution focused on linking existing visual element to appropriate data sources and create a dashboard by combining these elements. Similarly, the majority of proposed solutions have not considered the need for adjusting dashboard after its creation and once the dashboard is created, there is no room for customization or trying different possible dashboard designs. In the existing solutions, we did not witness the option to provide a flexibility in designing the whole frame of a dashboard. Although the content of the dashboard is customized in these solutions, the structure and the frame of the dashboard is fixed for all users and use cases, and the need to change the structure is not supported.

After exploring the literature of tailored dashboards, in the next section we present our designed approach.

3 Design Approach

To provide a solution to respond to the identified needs and enable healthcare professionals to easily create dashboards adapted to each deployment, AMI-Dash has been developed in collaboration with other teams in AMI-Lab with agile methodology. This methodology enabled us to assess the system throughout the development phase in several iterations and assure integrity of our solution with other elements of AMI-Platform to ensure that all parts of the system are working together. Dashboard Studio/Designer is the core of our proposed solution which enables healthcare professionals to design their intended dashboards and automatically adapt them to different environments based on information received from a Configuration Tool.

We followed a four-step agile approach to design AMI-Dash adaptive dashboards. In the **1st step**, we defined the requirements for the dashboards by analyzing existing work and grouping the requirements into end-user and technical needs. In the **2nd step**, we proposed a solution based on literature and technology to address the identified requirements, focusing on the concepts of “dynamicity” and “scalability.” In the **3rd step**, we implemented a web-based prototype called AMI-Dash, designed to be user-friendly and adaptable to the real-world environment. In the **4th step**, we evaluated the prototype through a two-part evaluation process, measuring the technical performance against the defined requirements.

3.1 1st Step: Requirements

We have identified five categories of technical requirements that involves dashboards in senior residences:

Req. A) Privacy Prevention and Security: Protected Health Information (PHI) is an important factor in healthcare dashboards. Legislation in many countries including Canada requires least privilege or “minimum necessary” access to personal health care information [13]. In another word, each health provider should be limited to access only portions of information that is essential for them to perform their tasks. However, the complexity of this requirement lies on the fact that health information is usually stored in a central storage for integrity and accessibility reasons [14].

Req. B) Fast Easy Deployment of a Dashboard: Deployment refers to delivering the final version of the designed dashboard to real users. Nowadays more complex software solutions are introduced in numerous domains including healthcare dashboards. These modern solutions usually involve adding third party systems to the solution including databases, firewalls and caching systems [15]. When it comes to deployment of new solutions, all these aspects should be considered to setup which leads to time-consuming procedure and needs of technical individuals and paper works regarding administrative permission.

Req. C) Real-Time Data Visualization if Needed: Real-time data refers to data which collected and processed in near real time and is available immediately after it is collected by sensing devices. This form of data can be specifically beneficial to reduce intervention time and improve patient safety, two important use-cases of dashboards in healthcare and senior residences. In both cases, intervention and safety, it is critical for healthcare professionals to notify any abnormal condition of the patient or the monitored person as quickly as possible.

Req. D) Low Resource Consumption: Dashboards usually include multiple charts and figures along with other visualization elements like maps and images which potentially can consume a high amount of resources. With the emergence of new RHM devices and IoT sensors, measured data for a large variety of health and environmental factors are available. Having such number of measured factors can be interpreted to have an even larger number of visualization elements on desired dashboards, which require handling a large amount of data and processing. Numerous dashboard solutions [16–18] are deployed on devices like tablets and smart TVs with limited computational resources. Therefore, it is critical for dashboard solutions to consume low resources of the system in order to be able to handle a large number of visualization elements and to be hosted on a variety of devices, especially when there is a need to load real-time data.

Req. E) Scalability: Dashboard solution should be able to handle unpredictable requirements in the future. Healthcare domain is very dynamic, and a wide variety of needs exists in this realm. With constant advancements in sensors and healthcare devices, it is highly likely to have new types of devices available in the market. These new devices might need new forms of visualization or new algorithms to enable healthcare professionals to analyze and interpret the data. Therefore, it is essential to be able to support emerging needs in healthcare domain.

3.2 2nd Step: Proposed Approach

Our approach is based on a number of subsystems to form the complete solution (Fig. 1). The main part of our solution is the Page-Designer which is responsible for enabling healthcare professionals to create and design dashboard pages based on their needs and preferences. This designer is equipped with functionality to communicate with IoT infrastructure of AMI-Lab, and visual elements of this page designer has attributes to connect to real devices in IoT infrastructure. As our proposed solution need to integrate the dashboard with IoT systems (fast dashboard creation), we need to have semantic information about the deployed sensors, which is received from Environmental Configuration Tool (AMI-ECT).

Once we can manipulate information on the deployed sensors and the deployment environment, there is a need for manipulating the representation (view) of this information. Chart-Designer provides healthcare professionals with the ability to manipulate dashboard charts with a great level of flexibility. It provides the ability to manipulate easily not only the layout of the pages and the arrangement of elements, but also, add new components and new visualization types that transform raw data to meaningful information. Lastly, Dashboard-Viewer is responsible for rendering the final layout of the designed dashboard for healthcare professionals with all required visualizations. This viewer along with previously mentioned parts of the solution are integrated with Authentication system of AMI-Platform to separate the design space of each healthcare professional and provide user tailored services.

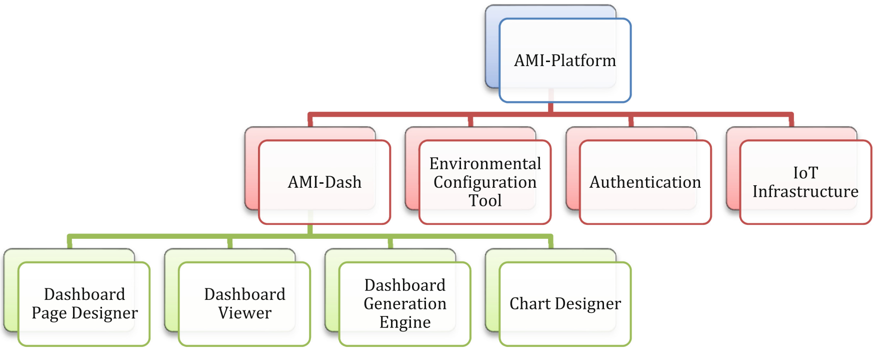

Fig. 1. Overview of proposed solution

Following we discuss how our proposed solution replies to the identified requirements:

Req. A) Privacy Prevention and Security: We integrated an authentication system in our solution which handles all the functionality to define users (healthcare professionals) and assigning their roles. Based on healthcare professional information, dashboard system applies limitations according to their roles and gives them appropriate data access and management permission. As an example, the designer may wish to limit a page to be visible only for healthcare professionals with "Nurse" role. Dashboard Studio can handle that based on their identifier as the logic of showing a page is handled there.

Req. B) Fast Easy Deployment of a Dashboard: We proposed a subsystem called Dashboard-Viewer in our solution to automate the deployment procedure. Dashboard-Viewer is responsible for reading the configuration of requested dashboard from the data source. The input of Dashboard-Viewer is a JSON object that contains information about the elements inside the dashboard and their arrangement inside the final layout of rendered dashboard. This JSON object starts with a ROOT child which lists all its children and therefore enables the render engine to detect how to start rendering elements. Once configuration is loaded, a render engine in Dashboard-Viewer will parse the configuration to extract items and their arrangement in the dashboard and start rendering the designed dashboard and load its content. As our Dashboard-Viewer is always running on the server, there is no need to configure any additional system and a dashboard is instantly available once its design is finished.

Req. C) Real-Time Data Visualization if Needed: To support visualization of real-time data, dashboard components can receive live updates from their data sources and visualize data in a real-time manner. For each component there is a long-running background task which is responsible for regularly updating data of the component on regular intervals. We also enabled healthcare professionals to adjust update intervals based on the sensitivity of the element's data. For instance, they might set update intervals for a heartbeat chart to half a second while setting the update interval of sleep status to five seconds since it has less sensitive data.

Req. D) Low Resource Consumption: Dashboard-Viewer is the core part of the system responsible for rendering layout and contents of the dashboard, making it the main target for our efforts to make it efficient. One of our initiatives is to limit refreshing dashboard charts to the time that they are visible, this approach can greatly reduce resource consumption. If a chart is not visible on the page, it will stop sending any request until it becomes visible again. Using Async load methods is another optimization we added. All data elements inside the dashboard send their requests for data in asynchronous mode, which means that loading the data of each chart is not part of loading the dashboard itself. We also enabled the compression on our web server to reduce the size of information received from the server, which minimizes the size of data exchange.

Req. E) Scalability: All the elements inside the dashboard are components, and each component can be programed to perform a specific task or visualize information in a certain way. Being components, plays a key role in our implementation as it enables the system to accept new components so that in case we need a new visualization method or dashboard element, we can simply develop and add these components to our system.

3.3 3rd Implementation of AMI-Dash

AMI-Dash is composed of the following three important components (Environment Configuration Tool, Chart Designer and Dashboard Designer).

a) Environment Configuration Tool: We built the Environmental Configuration Tool (AMI-ECT) to generate the semantic information about the deployment place precisely so that it can be consumed by the dashboard studio to drive the generation engine adaptively. AMI-ECT provides a 3D designer in which healthcare professional is able to drag-and-drop physical elements that exist in the deployment place to create a floor plan of a building, as well as place sensors on it and specify their types (Fig. 2). The output of this component is an "Environment Model" which precisely describes the deployment area and contains all data needed to adapt the dashboard design.

b) Chart Designer execution can be seen in the Fig. 3. At the very first step the healthcare professional is supposed to create a new chart and specify the title (name) of this new chart and declare if it is a static chart or a template chart. If they choose template charts, the design can be automatically adapted to new environments provided by AMI-ECT. Once this initial information is provided by the healthcare professional, the next step is to specify how to fetch data from data sources and define its filters and timeframe that should be captured from the source. The easiest option is to ask the end user to write a query string. However, there are several clear drawbacks to that approach. First, the healthcare professional needs to have technical knowledge about the database and its structure and the knowledge to write appropriate queries. Apart from that, even if they had the knowledge working which queries, by writing a raw query we limit our logic to a specific database type used in our implementation and if there was any change in the database or its structure, it would directly affect the way end-user will need to define the query. Therefore, we equipped our system with a number of data panels that can help healthcare professionals to define query and filters.

An important aspect of the chart designer is the ability to design templates. Healthcare professionals can define charts as a static chart that will be shown as they are designed initially, or they can use templates which will be adapted to different spaces declared by AMI-ECT. For example, in one deployment we might have a hundred patients equipped with an identical sensor (e.g., blood pressure). In this situation, the design of the chart is the same in all these cases and the only change is refining the data source of the chart. We tackled this problem by invoking a template and linking the blood pressure visualization of all these patients to one template.

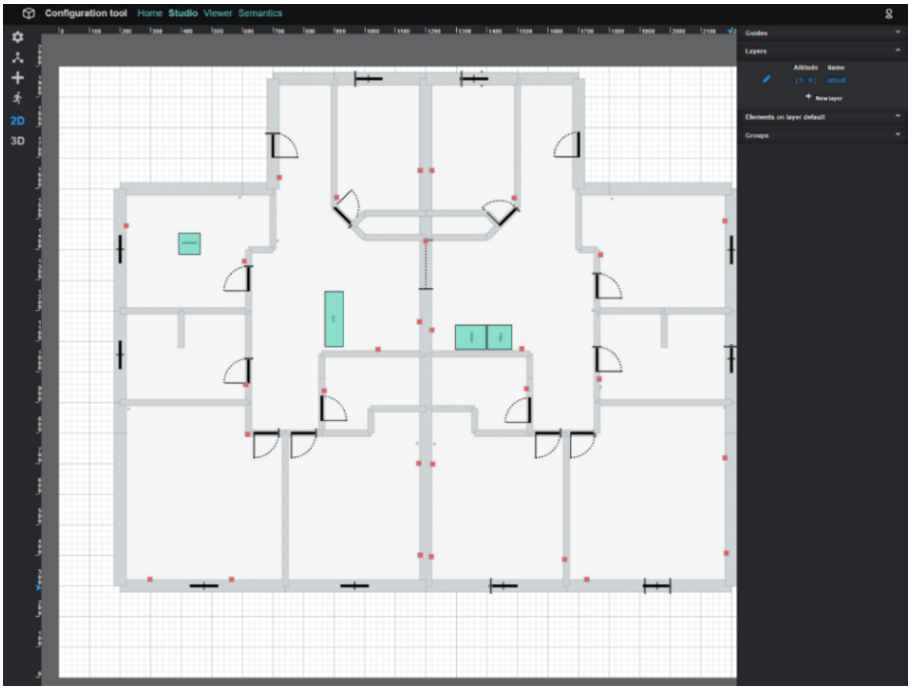

Fig. 2. Example of running configuration tool.

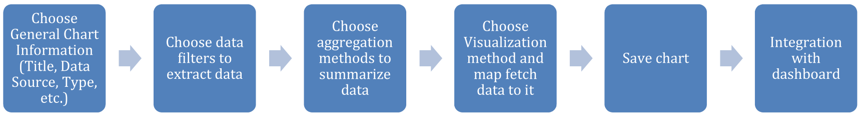
Choose General Chart Information (Title, Data Source, Type, etc.)
Choose data filters to extract data
Choose aggregation methods to summarize data
Choose Visualization method and map fetch data to it
Save chart
Integration with dashboard

Fig. 3. Overview of all steps in designing charts with AMI-Dash

C) Dashboard Designer: The cornerstone of our solution lies in this part of the system as it enables healthcare professionals to design dashboard pages and adapt them to the real-world environment based on the description provided through AMI-ECT. Dashboard Studio consists of five main components which will be explained in the following sections. Figure 4 illustrates the high-level design of AMI-Dash Studio:

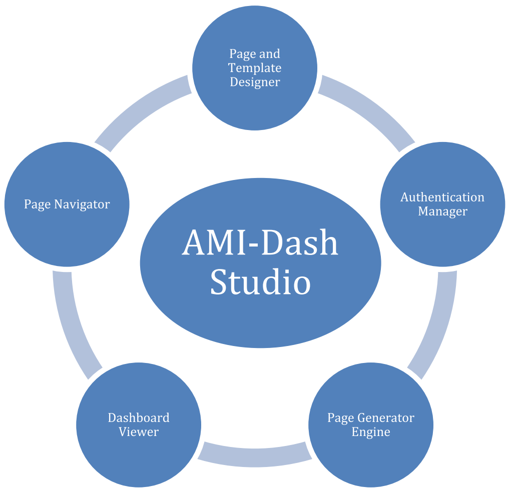

Fig. 4. Elements of Dashboard Studio

AMI-Dash studio (Fig. 4) consists of five main parts:

- **Page and Template Designer**: The main element in our solution in which healthcare professionals can design their desired Dashboards.
- **Page Generation Engine**: This engine is responsible for adapting healthcare professional designed dashboards to the arrangement of real-world environment declared by AMI-ECT.
- **Dashboard Viewer**: This part of the system handles rendering designed dashboard for final usage by healthcare professionals.
- **Page Navigator**: Enable navigation among dashboard pages
- **Authentication Manager**: Handles healthcare professional permission to limit their access to certain pages if needed.

3.4 4th Step Evaluation

We conducted an evaluation study to understand to which extent the proposed solution has replied to technical requirements (identified in 1st step). Following details about the study.

Evaluation Setting: AMI-Platform has been deployed and running for months now and in several projects located in Canada. We used our solution in two project settings:

- ***Behavior change detection in apartment deployments***: AMI-Lab platform is deployed in a number of apartments, in which older adults are equipped with a variety of environmental and health sensors which continuously monitor the following parameters: Motion, Temperature, Humidity, Illuminance, Sleep factors.
- ***Medical intervention monitoring***: using two wearable devices: 1) Apple Watches to follow up Heartrate, Oxygen level of blood, and Number of steps walked; 2) Dexcom Sensors to follow up sugar level of blood.

Evaluation Metrics: To validate the proposed solution, we targeted the full-wing metrics that are representative of the technical requirements:

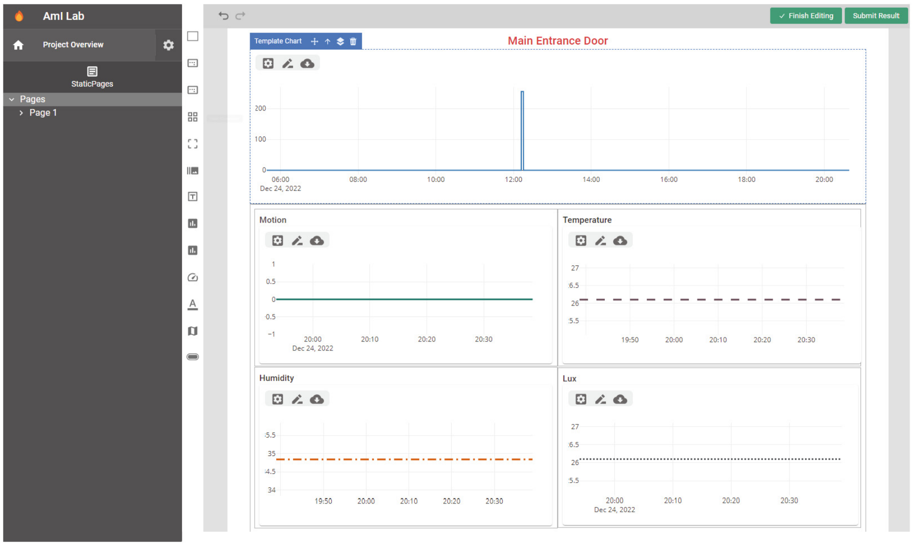

Fig. 5. Snapshot of dashboard page designer

Req. A) Privacy Prevention and Security: Limiting data access to certain roles was an important identified problem. In all deployments we needed to deliver different dashboards to our partners. Our implemented version successfully handled loading information for all users and no issue regarding authentication is reported. To assure confidentiality in the communication channel, we used a tool called "Charles Proxy" to sniff data exchange between our proposed dashboard and its server. Manual check of headers and content of sending and received packets, confirms that all exchanged data is encrypted using standard TLS methods, which makes content confidential, even if it passed unsecured channels as it uses end-to-end encryption. By taking these two steps, our results confirm that our implementation is capable of replying to the identified requirement.

Req. B) Fast Deployment: The proposed solution is deployed in several real apartment with the aim of monitoring older adults at home. Dashboards deployed in these apartments have different floor plans, however, they mostly include five door sensors, seven multi-sensors which measure a number of parameters together (motion, temperature, humidity, and lux). A sleep mat is also part of the project to measure the sleep-related parameters. Similarly, in another project, we needed dashboards to visualize measured data of the deployed Apple Watches and Dexcoms. Once the installment of physical devices was finished, we started to design required charts for these projects in Chart Studio, followed by creating desired dashboard using AMI-Dash Studio, we were able to simply drag and drop the required visual elements and connect them to the appropriate data sources. The whole process of designing the required charts and the dashboard itself took roughly 20 min for apartment deployments. Finally, once the design step is finished, the dashboard is immediately accessible with no need of performing complex procedures. It worth mentioning that developing dashboards using ordinary methods can take several months to even years.

Req. D) Low Resource Consumption: CPU was in idle mode for our dashboard, suggesting enough efficiency to even run the dashboard in other media like smartphones and smart TVs.We run the dashboard with 30 figures and 5 images in the dashboard along with normal elements (Texts, Buttons and Containers). As it can be seen in Fig. 6, the CPU consumption is very little and 97% of the times the

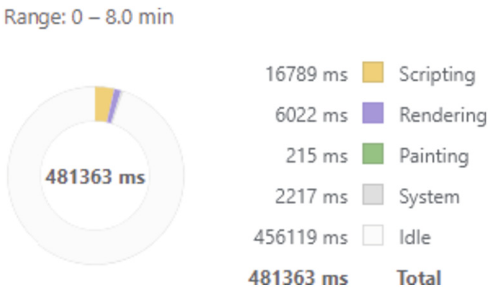

Fig. 6. Overall resource consumption of our Dashboard Studio

The result of our comparison shows that there is almost no difference between the visualization of the designed dashboard and the normal web page. Additionally, our dashboard is used for a period of two months by AMI-Lab partners and no complaint regarding performance was received.

In another scenario, we measured memory consumed by the dashboard while our dashboard was running and contained a certain number of charts. Each time we added a new chart and checked how much memory is consumed by our dashboard. Overall, the results show a reasonable memory consumption (around 190 MB with 30 charts). The trend of memory consumption was increasing as the number of charts increased. However, after adding 12 charts this increase started to be less considerable. This has roots in our implementation methods in which when a chart is not visible, our system stops sending request and allocating resources for it.

Req. C) Real-Time Data Visualization: In order to measure the delay in visualization, we manually inserted a new set of data to our data storage and check how long it takes

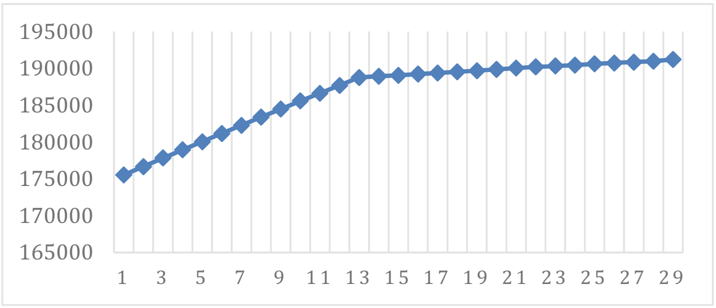

Fig. 7. Memory consumption and number of charts on the dashboard

for the new data to be reflected on the dashboard. To this end, we used software called "Postman" which is designed to interact with web interfaces and enable us to directly communicate with our backend services and add new data to our data source. Our data storage model contains a timestamp field, which indicates the exact time a new document is inserted in our system. We used this field along with another function which returns exact current time (server time) and we added a temporary label on charts to show the difference (delay). We set the refresh time of the chart at 100 ms and observed the measured delay and in all cases the delay was less than 100ms, proving the capability of our system to show real-time data.

Req. E) Scalability: As we used our solution in real deployments of AMI-Lab we faced few use-cases which were not originally defined in our solution. However, as we have used a component-based approach in our design, we are able to easily develop new components to respond to any new need. As an example, a colleague needed to annotate chart data shown on the dashboard, which even needed interaction with data sources to store annotated information. The integration process took less than 30 min for integration, and the added component worked smoothly with no functional error.

4 Conclusion

The study presented in this paper aims to propose a form of tailored dashboards that can allow healthcare professionals to create their own dashboards for senior residences. The motivation behind this study is to support healthy aging and help healthcare professionals deal with the increasing number of senior residents. The need for this study arose due to the limitations of existing technological solutions such as Remote Health Monitoring (RHM) and the Internet of Things (IoT) monitoring which do not provide a medium for healthcare professionals to understand the data generated by these devices.

The study followed an agile approach in four steps to reach its final solution, AMI-Dash. In the first step, the requirements for adaptive dashboards in senior residences were identified. In the second step, the solution was outlined based on the context of work, best practices in the literature of tailored dashboards and stacked technological solutions in AMI-Lab. In the third step, a prototype was implemented based on the outlined solutions. Finally, in the fourth step, an evaluation was performed to measure the usability of the solution for the end-user.

The real needs of healthcare professionals working in senior residences were explored through the literature of healthcare dashboards and the studies that were close to the context of work. The analysis of these studies led to the extraction of a number of requirements for dashboards in senior residences. The final dashboard designer was easy for healthcare professionals to use and provided the option to adapt the dashboard to different deployment places[1].

There are some limitations to the study such as the current implementation being coupled with the AMI-Platform and the lack of integration with other systems and managerial aspects. However, future work can involve integrating the solution with other IoT infrastructures and existing management tools in senior residences.

References

1. Verma, J., Samis, S.: Canada's population is aging. In: Healthc Pap, vol. 11(1), pp. 3–5, (2011). http://europepmc.org/abstract/MED/21464621
2. Parker, M.G., Thorslund, M.: Health Trends in the Elderly Population: Getting Better and Getting Worse (2007). http://gerontologist.oxfordjournals.org/
3. Tun, S.Y.Y., Madanian, S., Mirza, F.: Internet of things (IoT) applications for elderly care: a reflective review. In: Aging Clinical and Experimental Research, April 1, vol. 33(4), pp. 855–867. Springer Science and Business Media Deutschland GmbH (2021). https://doi.org/10.1007/s40520-020-01545-9
4. Dowding, D., et al.: Dashboards for improving patient care: review of the literature. Inter. J. Med. Inform. **84**(2), 87–100 (2015). https://doi.org/10.1016/j.ijmedinf.2014.10.001
5. Rabiei, R., Almasi, S.: Requirements and challenges of hospital dashboards: a systematic literature review. BMC Med. Inform. Decis. Mak. **22**(1), 287 (2022). https://doi.org/10.1186/s12911-022-02037-8
6. Abdulrazak, B., Paul, S., Maraoui, S., Rezaei, A., Xiao, T.: IoT architecture with plug and play for fast deployment and system reliability: AMI Platform. In: International Conference On Smart Living and Public Health, Springer Science and Business Media Deutschland GmbH, pp. 43–57 (2022). https://doi.org/10.1007/978-3-031-09593-1_4
7. Vázquez-Ingelmo, A., García-Peñalvo, F.J., Therón, R., Amo Filvà, D., Fonseca Escudero, D.: Connecting domain-specific features to source code: towards the automatization of dashboard generation. Cluster Comput **23**(3), 1803–1816 (2020). https://doi.org/10.1007/S10586-019-03012-1
8. Palpanas, T., Chowdhary, P., Mihaila, G., Pinel, F.: Integrated model-driven dashboard development. Inf. Syst. Front. **9**(2–3), 195–208 (2007). https://doi.org/10.1007/S10796-007-9032-9
9. Tundo, A., Castelnovo, C., Mobilio, M., Riganelli, O., Mariani, L.: Declarative Dashboard Generation
10. Santos, H., Dantas, V., Furtado, V., Pinheiro, P., McGuinness, D.L.: From Data to City Indicators: A Knowledge Graph for Supporting Automatic Generation of Dashboards (Apr 2017)
11. Belo, O., Rodrigues, P., Barros, R., Correia, H.: Restructuring dynamically analytical dashboards based on usage profiles. In: Andreasen, T., Christiansen, H., Cubero, J.-C., Raś, Z.W.

[1] The results of the ergonomic evaluation of our AMI-Dash are the subject of another article.

(eds.) ISMIS 2014. LNCS (LNAI), vol. 8502, pp. 445–455. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08326-1_45
12. Ines, D., Sebastien, I., Jean-Marie, G., Madeth, M., Serge, G.: Towards adaptive dashboards for learning analytic an approach for conceptual design and implementation. In: CSEDU 2017 - Proceedings of the 9th International Conference on Computer Supported Education, vol. 1, pp. 120–131 (2017). https://doi.org/10.5220/0006325601200131
13. Summary of privacy laws in Canada - Office of the Privacy Commissioner of Canada. https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/02_05_d_15/ (Accessed 30 Nov 2022)
14. Dash, S., Shakyawar, S.K., Sharma, M., Kaushik, S.: Big data in healthcare: management, analysis and future prospects. J. Big Data **6**(1), 1–25 (2019). https://doi.org/10.1186/S40537-019-0217-0/FIGURES/6
15. Wurster, M., Breitenbücher, U., Falkenthal, M., Krieger, C., Leymann, F., Saatkamp, K.: The essential deployment metamodel: a systematic review of deployment automation technologies. Softw. Intensive Cyber-Phys. Syst. **35**(1–2), 63–75 (2020). https://doi.org/10.1007/S00450-019-00412-X/FIGURES/3
16. Hoogeveen, I.J., et al.: A preliminary study of telemedicine for patients with hepatic glycogen storage disease and their healthcare providers: from bedside to home site monitoring. J. Inherit. Metab. Dis. **41**(6), 929–936 (2018). https://doi.org/10.1007/S10545-018-0167-2
17. Dowding, D., Merrill, J.A., Barrón, Y., Onorato, N., Jonas, K., Russell, D.: Usability evaluation of a dashboard for home care nurses. Comput. Inform. Nurs. **37**(1), 11–19 (2019). https://doi.org/10.1097/CIN.0000000000000484
18. Yoo, J., Jung, K.Y., Kim, T., Lee, T., Hwang, S.Y., Yoon, H.: A real-time autonomous dashboard for the emergency department: 5-year case study. JMIR Mhealth Uhealth 6(11) (2018). https://doi.org/10.2196/10666