sensorHUB – a Novel, Open-source Software Stack for Enhanced Accessibility and Secure Interoperability in IoT Project Management

David Gackstetter[1], Parid Varoshi[1], Senthold Asseng[1,2]

[1] Hans Eisenmann-Forum for Agricultural Sciences, Technical University of Munich, Germany - david.gackstetter@tum.de, pvaroshi@gmail.com

[2] Department of Life Science Engineering, TUM School of Life Sciences, Technical University of Munich, Germany - senthold.asseng@tum.de

Keywords: Internet of Things, Open-Source Software Stack, SensorThings API, Graphical User Interface, Secure Interoperability.

Abstract

The Internet of Things (IoT) seeks to achieve seamless device connectivity and data sharing, thereby enhancing automation, efficiency, and insights in various fields of application. By delivering real-time data, IoT is essential for systems that aim to model and map the physical world to digital formats, such as digital twins or metaverses. In the recent years, there has been a surge in high-quality commercial IoT products available to various sectors. However, these solutions often come with high costs and proprietary interfaces, which hinder seamless interoperability. For users seeking cost-effective and flexible solutions, open-source software and international standards present viable alternatives. Nonetheless, despite technological progress, the adoption of open and standardized IoT software solutions is still predominantly seen among technology-affine groups or within research and development initiatives. We propose the sensorHUB, a novel IoT stack, which builds upon the Open Geospatial Consortium’s (OGC) data standard SensorThings API and enhances existing solutions towards more accessible, production-oriented solutions for diverse application domains and user groups. Key features include the provision of a new, state-of-the-art graphical user interface for managing data flows and the access to related software modules, and a secure and flexible user and access management.

1. Introduction

The Internet of Things (IoT) refers to a network of distributed physical devices, which are typically embedded with sensors, software, and networking technologies to connect and exchange data with other devices and systems (Khanna and Kaur, 2020; Sadeghi-Niaraki, 2023). By providing real time information, IoT play an important in providing actualism for systems aiming to map the physical world to digital representation as in digital twins or metaverses (Moshrefzadeh et al., 2020; Li et al., 2023). Specifically for the Metaverse, IoT sensors represent a key data source in enhancing the functionality and the immersion for the users in several ways, including spatial mapping and navigation of users or the simulation of the actual real-world environment (Han et al., 2023; Asif and Hassan, 2023). In recent years, we have seen growing numbers of high-quality commercial products offering IoT solutions for a broad audience from diverse domains. These are, however, partially costly or restrict interoperability due to proprietary interfaces. Yet, in the context of Internet of Things (IoT) networks, decentralization and distribution are inherent, necessitating standardized interfaces to ensure interoperability among subsystems or networks (Chaturvedi and Kolbe, 2019; Lee et al., 2021). Open-source software, particularly when implementing international standards, provides alternatives for user groups that are cost-sensitive or require adaptable software that still adheres to common quality standards. Open-source initiatives inherently encourage collaboration and innovation, driving the development of versatile solutions that benefit a wide range of users and applications (Steinmacher et al., 2017; Heron et al., 2013). Different initiatives promote the open-source character and standardization of data exchange and formats in the field of IoT, such as OpenAPI or SensorThings API (STA) (Tzavaras et al., 2023), which not only facilitate interoperability but also ensure that IoT solutions are scalable and long-term maintainable. The STA was published by the Open Geospatial Consortia (OGC) in 2016 and has since then proven to be a functional standard for many IoT projects (Liang et al., 2016). However, while STA has seen increasing adoption in IoT projects (Hertweck et al., 2019; Huang and Chen, 2019; Zhang et al., 2023), particularly in open-source solutions, many lack crucial operational features necessary for production-oriented IoT stacks. A popular STA-related server implementation for storing and querying sensor data represents the Fraunhofer SensorThings API (FROST)-server (Fraunhofer Institut IOSB, 2020). Building upon the FROST-servers, several projects have worked on extensions towards scalable and modularizable IoT stacks. Yet, besides the Hamburg Urban Data Platform (HH_UDP) (Landesbetrieb Geoinformation und Vermessung Hamburg, 2022), which represents a well-scalable and production-oriented IoT stack, current open-source solutions show limitations concerning multiuser and multi-project operations. Also, the HH_UDP reveals constraints in terms of user-friendliness and accessibility, due to the lack of a publicly available graphical user interface (GUI). The development of a more intuitive and accessible GUI could significantly enhance user engagement and operational efficiency. In response to these gaps, this study aimed to develop an enhanced, open-source IoT stack that builds upon components from existing frameworks while incorporating new crucial features for practical operations. Specifically, this new stack integrates a modern, web-based GUI application for STA data administration, facilitates interfacing with additional IoT modules for data preprocessing and visualization, and automates user management processes including registration, authentication, and administration within an Identity and Access Management (IAM) system. Consequently, this study aspires to provide a robust and user-friendly IoT stack suitable for multiuser and multi-project scenarios for diverse application domains.

2. Methodology

2.1 Concept Development

Given the described problem situation, the primary requirements of the aspired software solution needed to be defined aligned with common software design principles (Kossiakoff et al., 2020; Hehn et al., 2022; Alabadi et al., 2022):

- Open-source and freeware licensed: free of charge and available for everyone to use and modify.
- Security: control of user and access rights to prevent unauthorized manipulation and misuse of other users' data.
- Reliable and proven: relying on existing software solutions that haven demonstrated their functioning across different projects and that are still maintained.
- Interoperability: usage of standardized interfaces to enable barrier-free communication between the different software modules.
- Accessible and low-barrier management: provision of a graphical user interface for managing data and applications without the necessity for high-level, technical expertise.
- Scalable: potential to flexibly enlarge the number of data servers and auxiliary IoT tools.
- Modularization: possibility to exchange software modules and extent the IoT stack with further features and tools.
- Adaptable: possibility to fine-tune IoT stack to specific local settings, regarding identity providers and layout.

To reach a solution covering all mentioned requirements in a most efficient and effective way, we focus on reusing as many existing solutions as possible. Specifically, our IoT stack shall build upon the Fraunhofer SensorThings API (FROST)-server for the sensor data management, Keycloak for user and access management, and auxiliary services, which are presented in more detail in section 2.2. Given the lack of an available GUI-based web applications for managing multiuser, STA-based IoT projects, an entirely new user interface (UI) was required to be developed, which shall handle login and registration processes, project-oriented data management, and interfaces to supportive tools.

2.2 Software Components

2.2.1 Data Management: SensorThings API (STA) represents an open data standard developed by the Open Geospatial Consortium (OGC) for managing, querying, and accessing IoT sensor data (Liang et al., 2016). It provides a standardized way to describe and query IoT data streams, observations, and sensor metadata. One major feature of STA represents its RESTful architecture, which uses HTTP methods for communication, making it accessible over the web. Moreover, it employs JSON as its primary data format for ease of interoperability and simplicity in data exchange. The API is based on data entities (including ”Things”, ”Sensors”, ”Datastreams”, etc.), which form a hierarchical structure that enables users to organize and query IoT data effectively. Fraunhofer SensorThings API-Server, developed by Fraunhofer Institute for Integrated Circuits (IIS), is a robust and scalable server implementation for STA (Fraunhofer Institut IOSB, 2020). It handles data processing, storage, and retrieval efficiently, serving for potentially large volumes of sensor data. Key features include structured data inspection, advanced querying for time and location-based data filtering, and spatial/temporal indexing for optimized performance. The server offers customization options, authentication, and authorization mechanisms to ensure data privacy and integrity. It is interoperable with existing IoT systems, facilitating seamless integration, making it a comprehensive solution for deploying and managing STA-compliant IoT applications with scalability and flexibility.

2.2.2 Identity and Access Management: Identity and access management (IAM) software represent essential components in operational software architectures. IAM tools provide functionalities to securely manage and control user identities and access to systems, applications, and data (Indu et al., 2018; Chapple, 2021). Keycloak represents one open-source IAM solution by software company Red HAT (Keycloak Authors, 2023), which offers a comprehensive set of features including user authentication, authorization, identity mediation, and numerous others. This platform empowers developers to protect their applications by implementing diverse standardized identity protocols, such as OpenID Connect or OAuth2, and by integrating security measures like single sign-on, social login with diverse identity providers (e.g., Google, GitHub), and multifactor authentication facilitating the smooth integration with existing systems and applications.

2.2.3 Front-end Web Development: The purpose of a front-end web application consists in providing users with a GUI to interact with and access the functionality of the underlying web services, in our case IoT data services. This interface ensures that users can intuitively visualize and manage complex data streams from various IoT devices (Rathinam, 2023; Goh et al., 2023). ReactJS represents a widely used JavaScript library that strongly facilitates front-end development through its declarative and component-based structure (Gackenheimer, 2015). Its efficiency in managing UI state and rendering performance makes it suitable for developing robust and scalable applications. The framework is further characterized by a wide scope of extensions and templates, and a large developer community. This extensive support network ensures continuous improvement and innovation, keeping the platform up-to-date with the latest industry standards. Furthermore, the availability of numerous third-party libraries and tools simplifies the integration of additional features and functionalities.

2.2.4 Complementing Services: Additional backend components, in some context also denoted as middleware (Hadim and Mohamed, 2006), facilitate communication and integration between different software applications or components, enabling them to work together seamlessly. These components act as intermediaries that ensure the efficient exchange of data and functionality across various systems. In our case, this involves tasks related to the registration of new users, the administration of Docker containers (Docker, Inc., 2024), and the maintenance of logging reports. From the scope of open-source software, we identified MySQL as database solution (Oracle Corporation, 2024), offering robust and reliable data management capabilities, and Flask, a Python-based web framework, as a suitable middleware backbone due to its lightweight and modular structure, widespread usage and adaptability, which allows for flexible and efficient development (Pallets, 2023; Pröll et al., 2017).

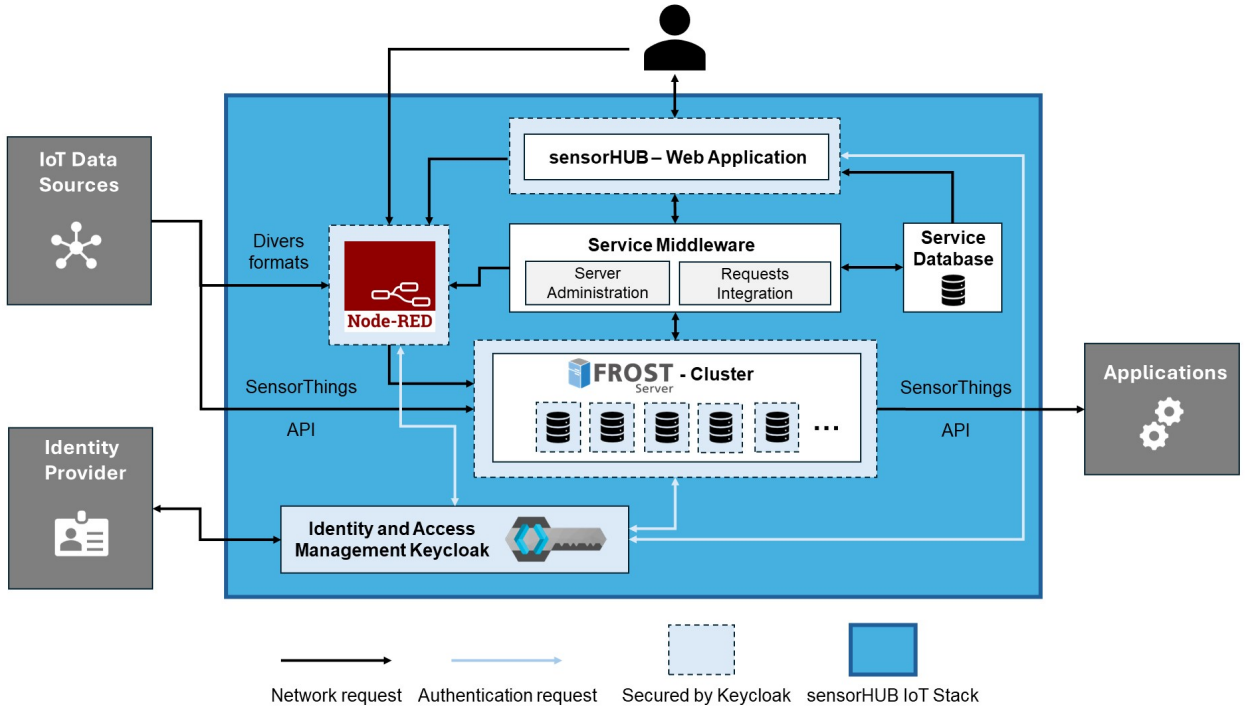


Figure 1. Conceptual architecture of sensorHUB stack with internal components and their interrelations, and links to external modules

3. Results

3.1 sensorHUB - IoT Stack

The sensorHUB stack builds upon existing backend solutions from data management and IAM, and puts a new graphical web application into the center of the architecture (Figure 1), aiming to enhance the latest technology stack towards a novel, user-friendly IoT project solution. The sensorHUB’s graphical web application represents a new front-end application for managing IoT projects (Figure 2). Its streamlined navigation ensures simplicity, allowing users to reach their intended destinations in a few clicks. The primary features include the management and visualization of IoT data. The app further provides a knowledge section, which includes essential training material on the provided software services. Interfaces to complementary IoT software (here: Node-RED) and optional forwarding of login tokens complement the default functionalities.

Aligned with our requirements on interoperability, scalability and open-source, the data-related core of the sensorHUB stack represents a scalable cluster of Docker-containerized FROST-servers. The original sources of IoT data adding data to the FROST-Servers can be manifold; the post requests however need to fulfil the STA, which can be done directly from an external source or indirectly by transforming third-party data formats into STA using optionally provided instances of the Extract-Transform-Load (ETL)-software Node-RED (OpenJS Foundation, 2024).

By integrating Keycloak into our application, we ensure that data and interactions are protected in accordance with users’ access rights. Keycloak provides authentication to the sensorHUB web application and secures its routes by requiring users to authenticate themselves. Due to its decentral approach, further modules implementing the required authentication interfaces can be easily integrated. In addition, several supportive components were required to guarantee the scalability and functioning of the network communication and server administration. Each sub-component is hosted as web application and can therefore be addressed using HTTP/S network communication. In the following we explain each of the stack’s sub-components in more detail.

3.1.1 FROST-Server based Data Management: Building upon the FROST-server as central storage for IoT sensor data (see 2.2.1), we require each in- and outgoing data stream to implement the STA’s specifications (Figure 1). By today, FROST-servers only implement coarse user access rights (read, write, delete, etc.), which apply for each user globally across the respective server instance. Hence, there’s no data entity specific granularity level, which might be required in some project scenarios. The sensorHUB’s current implementation bypasses this lack by providing each IoT project group an own FROST-server instance, which can only be accessed by the assigned users as specified in the Keycloak IAM (see 3.1.2). FROST-server natively supports Keycloak as authentication provider, which significantly facilitates implementation. Principally, each server instance is launched as Docker container using the latest image version and building upon a PostgreSQL database.

3.1.2 Keycloak for Identity and Access Management: Security and trustworthiness comprise major requirements for professional IoT project services. The IAM software Keycloak represents consequently a key component in the IoT stack, as it efficiently and reliably manages the security of data flows and user-specific access rights. As can be seen in Figure 1, Keycloak regulates the access to the sensorHUB’s web-based GUI, the FROST-server instances, and supportive IoT modules (here: Node-RED) using OAuth2 protocol. At first access, users are required to register using the sensorHUB web application. The registration requirements can be defined freely by each hosting organization. In our specific case, users are verified if they are valid members using an LDAP (Lightweight Directory Access Protocol)-authentication service of the Technical University of Munich, before initiating the further registration procedure. Having registered successfully, users need to identify via user credentials or an external identity provider to login.

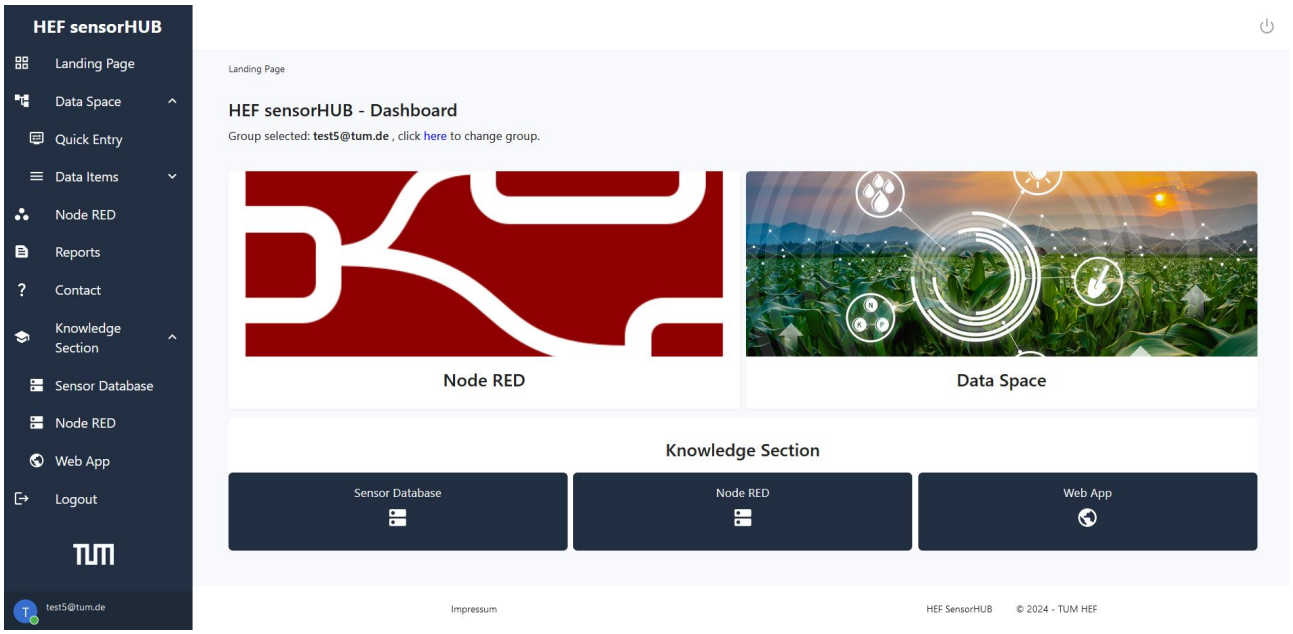


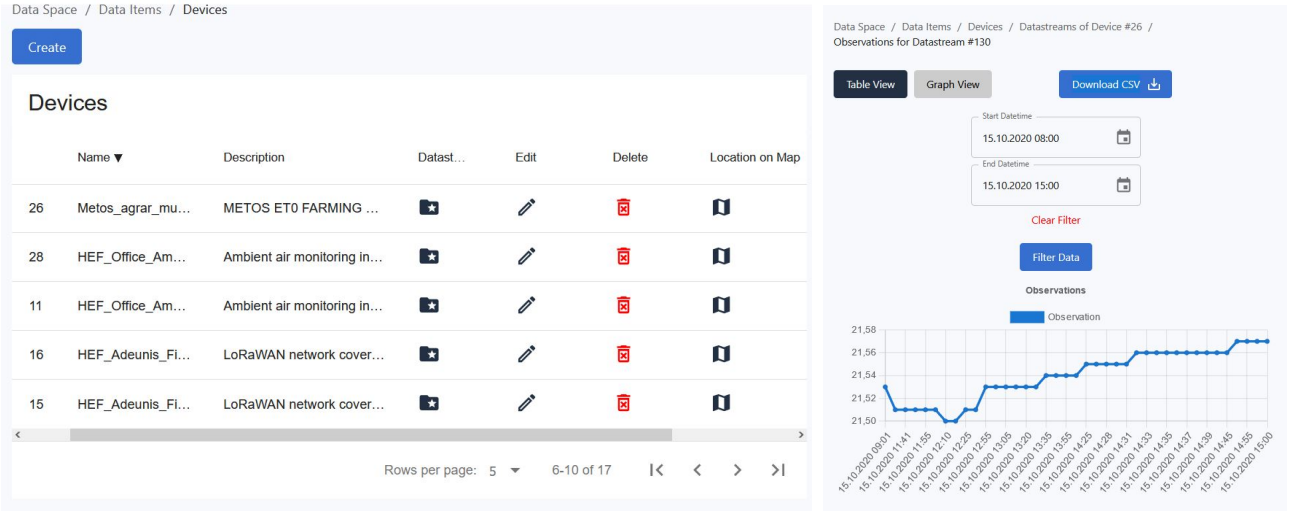
Figure 2. Dashboard of sensorHUB’s web-based graphical user interface for the example of sensorHUB being hosted at Hans Eisenmann Forum for Agricultural Sciences (HEF) of Technical University of Munich (TUM) (images by OpenJS Foundation (2024) and © lamyai/Adobe Stock 2024)

Once users are logged in using a web browser, the OAuth2-access token is stored temporarily, which enables seamless navigation from one application to another based on single-sign-on (SSO) principle. Alternatively, users can also communicate with the FROST-server instances in non-GUI based, headless mode (e.g., via command line), which requires uses to retrieve an access token directly via Keycloak’s API. Beyond user authentication, Keycloak manages the mapping of server instances for both FROST and Node-RED-servers to users and user groups with, if available, related application specific user access types (e.g., read/write for FROST).

3.1.3 Service Middleware: In managing backend functionalities several complementary middleware services have been implemented. Dynamic services being responsible for executing more complex processes and networking with other stack components are based on a Flask web server (see 2.2.4). Storing of related interactions and server states is performed using MySQL database (“Service Database”, Figure 1). One major, involved process entails the execution of the logic for new user creation/registration, which comprises 1) the communication with the web application to retrieve the username, 2) the exchange with Keycloak regarding the state of authentication, and 3) the deployment of one Docker container for each FROST and Node-RED, which are both associated with the user as clients in Keycloak. The service database simultaneously tracks each of these steps’ progress state and potential errors, together with the container-to-user mapping. Besides, middleware services conduct the more processing intensive queries from the sensorHUB web app to shift processing load and avoid lags in the front-end. Furthermore, every interaction with the FROST-server triggers the storage of logs, capturing each activity along with the timestamp of the alteration. The complementary database also serves as a repository for queries generated by users through the contact form within the sensorHUB web app. These queries are forwarded via SMTP mailing service to the support team for further investigation and resolution.

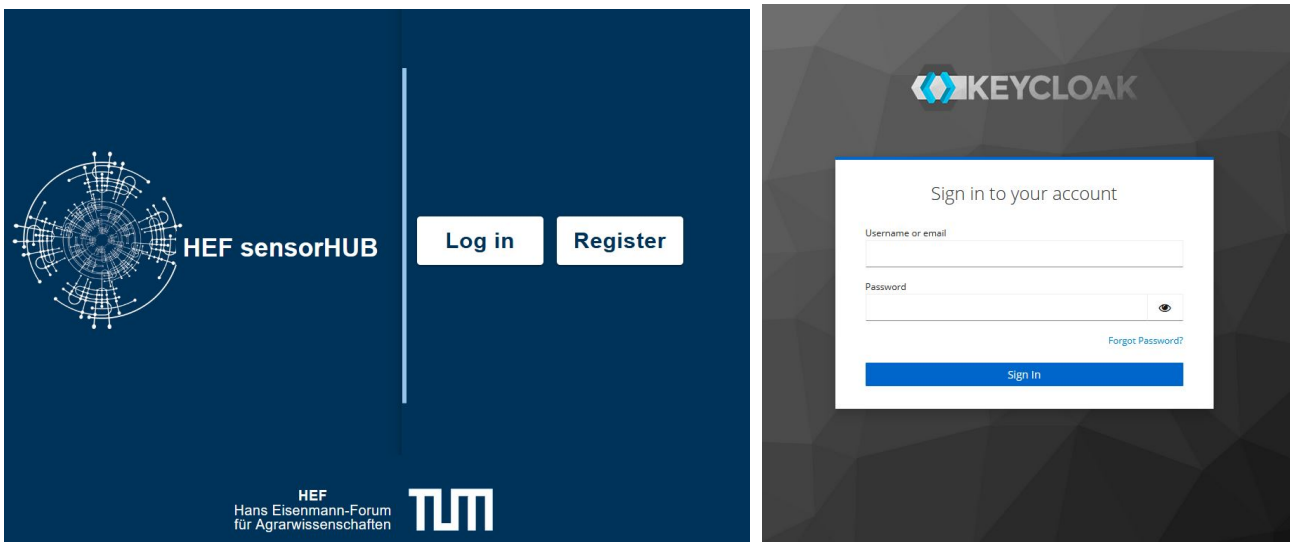
3.1.4 Management Web Application: From a user perspective the sensorHUB web-based GUI represents the central access to the stack’s services including data processing, project management functionalities, and knowledge acquisition. With ReactJS, the web app builds on a widely used, state-of-the-art front-end web framework (see 2.2.4). For facilitated development, we apply pre-configured designs and code modules from Material UI (Material UI SAS, 2024), a widely used, open source ReactJS component library. The app is designed as reactive web application, enabling its usability from diverse end device platforms. User access the app using a landing page, which either forwards to the login or a registration procedure both being regulated by Keycloak’s authentication mechanism (Figure 4). After a successful login, the user is required to select from the scope of available FROST-servers (a user may be assigned several projects with related servers) before being forwarded to the app’s dashboard (Figure 2). Contents are organized using a sidebar, which supports users in navigating to their intended destinations. In the following, the app’s main components are presented in more detail:

a) *Data Space*: The Data Space represents the web app’s core section offering a Quick Entry and detailed access to the sensor data via Data Items (Figure 2). The latter section is structured by the STA data entities (i.e., Things, Sensors, Measurement Property, Datastreams, etc.). This approach gives users full range of detail and control to view, create, modify, and delete STA data entities in each of the respective sub-sections (Figure 3a). Apart from minor adjustments for enhanced understandability (e.g., for Devices instead of Things), we closely aligned with the original STA terminology. The Quick Entry provides direct shortcuts to two major, data-related functionalities: checking sensor observations and registering new IoT devices. The first one forwards to the Devices section (Figure 3a), where users can further navigate to each IoT device’s related data streams with stored observations. The latter can be filtered by time, exported as CSV file, and visualized in table format or via simple graphs (Figure 3b). Beyond, the



(a) (b)

Figure 3. Data space sub menus (a) for managing Devices (in STA terminology ”Things”) and (b) for investigating received sensor data observations using filter options and available viewing modes (table or graph)



(a) (b)

Figure 4. (a) Landing page with login and registration options (here: example of sensorHUB hosted at Hans Eisenmann Forum for Agricultural Sciences (HEF) of Technical University of Munich (TUM)) and (b) Keycloak-based login mask for registered users

Quick Entry section offers users a simplified way to register new devices. The involved procedure can also be implemented manually using the related subsections in Data Items; the Quick Entry’s visual pipeline however substantially facilitates this process by offering a step-by-step logic, which automatically relates all newly created entities to each other.

b) *Knowledge Section*: The sensorHUB’s Knowledge Section offers instructions to the essential components of the software stack. It firstly provides instructive material on all sensor data related aspects (”Sensor Database” in Figure 2), including backgrounds and hands-on documentations to the STA standard and the FROST-server. Beyond, this section supplies documentation and tutorials on the Node-RED application, and a comprehensive overview to the sensorHUB web app itself. Future content can easily be implemented for additionally included IoT modules.

c) *Supportive Functionalities*: The sensorHUB web app also provides assistive functionalities aiming to enhance the user experience. These include account management functionalities, such as automated password changes using the Keycloak API endpoint, a reporting service, which provides a protocol on conducted manipulations and errors of the FROST-server, and a contact form to communicate with the support team regarding issues or feedback.

d) *Interfaces to Complementing IoT Modules*: To support a seamless user experience between the included modules in the IoT stack, the web app also implements graphical interfaces, which directly forward the user accordingly. In its current form, the IoT stack integrates a link to the user’s personal Node-RED instance both in the sidebar and on the landing page (Figure 2).

3.1.5 Complementary IoT Modules: IoT sensor data pipelines typically comprise several steps ranging from the physical IoT device sending data to a network, which forwards the data to receiving applications for data transformation, storage, and post processing. Given the divers range of required utilities, we designed the sensorHUB in a modularized way, to be flexible towards integrating additional IoT modules.
Currently, the stack includes the open-source ETL software Node-RED to enable users to build automated pipelines for receiving data from various data sources using common network protocols (e.g., MQTT, HTTP, TCP), and transforming received data packages to the STA format before posting to the FROST-servers. The project envisions also to include the open-source data visualization tool Grafana (Grafana Labs, 2024), which offers native support for STA data sources.

3.2 Open-Source Deployment

The software is available as open-source under the CC-BY licence, inviting the community to engage via the sensorHUB GitHub project (Gackstetter and Varoshi, 2024). Contributions play an elementary role in the open-source community, offering opportunity to learn and innovate. All contributions are warmly welcomed, and interested parties can find additional information in the contribution guidelines. Maintenance of the software is intended for the upcoming years, with the potential for extension based on the feedback and resonance of the user community. Regarding deployment, all components and the entire stack operate on Docker, ensuring straightforward deployment and scalability as well as facilitating updating and rollback capabilities. The software has undergone testing on both Windows and UNIX systems to ensure compatibility and dependability. Detailed installation instructions and requirements are available to assist users in seamlessly integrating the software into their systems.

3.3 Applications

IoT sensors are vital for diverse domains and particularly for mapping real-time, physical states to digital counterparts, as for digital twinning and metaverse applications. By providing links between physical environments with digital platforms, these devices promote innovations in various industries such as agriculture, healthcare, smart cities, or manufacturing. Originally, we developed our proposed IoT stack within the context of digital twinning and agricultural research data management (Moshrefzadeh et al., 2020; Gackstetter et al., 2021). Specifically here, IoT sensors assist for example in monitoring ambient parameters including soil moisture levels or temperatures, but also crop health in real-time, jointly optimizing irrigation schedules and enhancing crop yields. We consider the sensorHUB’s applicability, however, independent of agriculture likewise for any other IoT application domain: In healthcare for instance, wearable IoT devices track vital signs and activity levels, enabling remote patient monitoring and personalized healthcare interventions. In smart cities, IoT sensors monitor traffic flow, air quality, and energy consumption, facilitating data-driven decision-making for urban planning and resource allocation (Khanna and Kaur, 2020; Asif and Hassan, 2023). The stack can be used for single projects only but shows the largest potential and suitability for multi-project and multiuser application areas. This might apply in particular for application scenarios with a central hosting institution, which supplies the IoT stack as a service for several departments or consortia, covering several IoT projects and related diverse user constellations. Taking the open-source character into account, we see the stack especially interesting for research and development, and educational environments. Here, for instance a university or school facility can provide the stack as a service to support learning and experimenting with IoT sensors, and beyond, also as supporting software for instance for developing metaverse-related experiments and projects.
Moreover, the sensorHUB represents a strong tool in the context of research data management, which becomes increasingly important in academia. This refers not only to local storage but also to institution-overarching data sharing under the FAIR principles (Chue Hong et al., 2021; Lamprecht et al., 2020; Specka et al., 2023), for example, within multi-institutional data catalog systems, as it was designed for within this research project. The stack’s versatility extends to aiding collaborative research efforts by providing a centralized platform for data management and sharing, fostering interdisciplinary collaboration and knowledge exchange. Its compatibility with emerging technologies and standards ensures its adaptability and hence long-term suitability in evolving research landscapes.

4. Discussion

Evaluating the sensorHUB against the initially defined requirements (see 2.1) points out that the current implementation already satisfies a large share of common quality standards of professional software development:
The sensorHUB shows strongest with respect to interoperability, modularization and adaptability. Due to the usage of free, open, and internationally recognized data standards and software components, interfacing the individual modules within the stack and beyond with external applications is strongly facilitated. Consequently, institutions can easily extend and adapt the stack according to their individual needs.
Given the diverse range of IoT data sources, we decided against tailoring the sensorHUB to only specific ones, and instead chose to include an ETL module. With Node-RED in this case, users have the possibility to customize data pipelines more freely. At the same time, it requires a basic level of programming skills from the users, which might pose some barrier to the usability, despite Node-RED’s rather intuitive visual programming approach. To alleviate this obstacle, we provide learning material, hands-on tutorials and programming templates. Generally, the integration of an ETL is optional; hosting organizations can also decide against including this module. Besides, the sensorHUB already offers basic data visualization; with the open-source software Grafana (Grafana Labs, 2024); we’re, however, currently working on attaching an additional, external module for more sophisticated data visualization tool in a future version.
When it comes to security and reliability, the sensorHUB strongly benefits from the usage of existing and proven, open-source components, including Keycloak, FROST, or ReactJS. These software solutions have demonstrated their usability for several years already, also due to continuous maintenance by active communities. There can never be guaranteed one hundred percent of security; yet, additional measures such as restricting the availability of the IoT stack to an institutions’ subnetwork can further reduce cybersecurity risks. Despite, the individual subcomponents showing reliable for many years, the overall stack is still in an early stage, which implies potential demands for further improvements. These may refer for instance to further features of the web app: for example, currently

users still need to contact the administrator for granting specific users/user groups access to a FROST instance, for which future versions might include an automated service.
Beyond, the current version is still only tested on a few selected users, which provided feedback based on use cases from the field of agricultural sciences and smart cities. Collecting feedback from larger numbers of users and more diverse domains will benefit the usability of the web app and the overall stack's generalizability. Hence, dedicated user studies evaluating the measurable benefits and weaknesses, and the level of accessibility of the sensorHUB compared to other systems are yet to be conducted. Furthermore, there's still potential for improvement related to processing efficiency and server management. The sensorHUB is so far tested on a few dozen server instances. Beyond this, we recommend the usage of more sophisticated docker orchestration solutions (e.g., Kubernetes (Google, 2024)) to facilitate upscaling and server administration. Together with efficiency enhancements of the web app, this orchestration feature depicts a major future work of the sensorHUB project.
Overall, given the positive resonance from first, testing research projects, the sensorHUB's first version shows promising for further extension. Representing a university project with limited resources and still several required improvements ahead, we warmly invite the community to participate in this open-source project by providing feedback, studies on applicability and user experiences, or by directly collaborating with us via the sensorHUB GitHub repository (Gackstetter and Varoshi, 2024).

5. Conclusions

Already today IoT plays a key role in various domains. In the future, IoT will be even more intensely involved in bridging the physical and the digital world as in digital twinning or metaverse applications. Open and standardization technologies are enablers towards ensuring that the next generation of virtual environments becomes accessible for everyone. Aligned with open science principles of resource sharing and open knowledge, we provide sensorHUB, a novel, open-source IoT stack for both professional and educational IoT projects, which builds upon OGC's SensorThings API to guarantee interoperability and efficient data management. With its focus on secure multi-project environments and graphical interfacing, sensorHUB aims to provide a user-friendly, extendible, and operational IoT stack solution.

Acknowledgements

We would like to express our sincere gratitude to all contributors of the sensonrHUB's development in the recent years. Specifically, our thanks go to the team and the member institutions of the World Agricultural Systems Center - Hans Eisenmann-Forum for Agricultural Sciences (HEF) of Technical University of Munich (TUM) for their user oriented feedback and concrete technical developments. Beyond, we also thank the TUM Chair of Geoinformatics, which provided initial ideas, existing concepts and technological preworks as well as support during the project development. Finally, we are grateful to HEF for providing the resources and working environment for this project.

References

Alabadi, M., Habbal, A., Wei, X., 2022. Industrial Internet of Things: Requirements, Architecture, Challenges, and Future Research Directions. *IEEE Access*, 10, 66374–66400. doi.org/10.1109/ACCESS.2022.3185049.

Asif, R., Hassan, S. R., 2023. Exploring the Confluence of IoT and Metaverse: Future Opportunities and Challenges. *IoT*, 4(3), 412–429. doi.org/10.3390/iot4030018.

Chapple, M., 2021. *Access control and identity management*. Information systems security & assurance series, third edition edn, Jones & Bartlett Learning, Burlington, MA.

Chaturvedi, K., Kolbe, T. H., 2019. Towards Establishing Cross-Platform Interoperability for Sensors in Smart Cities. *Sensors*, 19(3). doi.org/10.3390/s19030562.

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., Struck, A., Lee, A., Loewe, A., van Werkhoven, B., Jones, C., Garijo, D., Plomp, E., Genova, F., Shanahan, H., Leng, J., Hellström, M., Sandström, M., Sinha, M., Kuzak, M., Herterich, P., Zhang, Q., Islam, S., Sansone, S.-A., Pollard, T., Atmojo, U. D., Williams, A., Czerniak, A., Niehues, A., Fouilloux, A. C., Desinghu, B., Goble, C., Richard, C., Gray, C., Erdmann, C., Nüst, D., Tartarini, D., Ranguelova, E., Anzt, H., Todorov, I., McNally, J., Moldon, J., Burnett, J., Garrido-Sánchez, J., Belhajjame, K., Sesink, L., Hwang, L., Tovani-Palone, M. R., Wilkinson, M. D., Servillat, M., Liffers, M., Fox, M., Miljković, N., Lynch, N., Martinez Lavanchy, P., Gesing, S., Stevens, S., Martinez Cuesta, S., Peroni, S., Soiland-Reyes, S., Bakker, T., Rabemanantsoa, T., Sochat, V., Yehudi, Y., WG, R. F., 2021. FAIR Principles for Research Software (FAIR4RS Principles). doi.org/10.15497/RDA00068.

Docker, Inc., 2024. Docker. https://www.docker.com/.

Fraunhofer Institut IOSB, 2020. FRaunhofer Opensource SensorThings-Server. https://github.com/FraunhoferIOSB/FROST-Server.

Gackenheimer, C., 2015. *Introduction to react: Using react to build scalable and efficient user interfaces*. Springer, New York.

Gackstetter, D., Moshrefzadeh, M., Machl, T., Kolbe, T. H., 2021. Smart rural areas data infrastructure (sradi) – an information logistics framework for digital agriculture based on open standards. *41. GIL-Jahrestagung, Informations- und Kommunikationstechnologie in kritischen Zeiten*, Gesellschaft für Informatik e.V, Bonn, 109–114.

Gackstetter, D., Varoshi, P., 2024. HEF sensorHUB. https://github.com/HEFLoRa/HEF-sensorHUB.

Goh, H.-A., Ho, C.-K., Abas, F. S., 2023. Front-end deep learning web apps development and deployment: a review. *Applied intelligence*, 53(12), 15923–15945. doi.org/10.1007/s10489-022-04278-6.

Google, 2024. Kubernetes. https://kubernetes.io.

Grafana Labs, 2024. Grafana. https://grafana.com.

Hadim, S., Mohamed, N., 2006. Middleware: Middleware Challenges and Approaches for Wireless Sensor Networks. *IEEE Distributed Systems Online*, 7(3), 1. doi.org/10.1109/MDSO.2006.19.

Han, Y., Leung, C., in Kim, D., 2023. Iot–assisted metaverse services. D. T. Hoang, D. N. Nguyen, C. T. Nguyen, E. Hossain, D. Niyato (eds), *Metaverse Communication and Computing Networks*, Wiley, 241–265.

Hehn, J., Mendez, D., Brenner, W., Broy, M. (eds), 2022. *Design Thinking for Software Engineering*. Progress in IS, Springer International Publishing, Cham.

Heron, M. J., Hanson, V. L., Ricketts, I., 2013. Open Source and Accessibility: Advantages and Limitations. *Journal of Interaction Science*, 1(1), 2. doi.org/10.1186/2194-0827-1-2.

Hertweck, P., Hellmund, T., van der Schaaf, H., Moßgraber, J., Blume, J.-W., 2019. Management of sensor data with open standards. *ISCRAM*.

Huang, C.-Y., Chen, H.-H., 2019. An Automatic Embedded Device Registration Procedure Based on the OGC SensorThings API. *Sensors*, 19(3). doi.org/10.3390/s19030495.

Indu, I., Anand, P. R., Bhaskar, V., 2018. Identity and access management in cloud environment: Mechanisms and challenges. *Engineering Science and Technology, an International Journal*, 21(4), 574–588. doi.org/10.1016/j.jestch.2018.05.010.

Keycloak Authors, 2023. Keycloak Documentation. https://www.keycloak.org/documentation.

Khanna, A., Kaur, S., 2020. Internet of Things (IoT), Applications and Challenges: A Comprehensive Review. *Wireless Personal Communications*, 114(2), 1687–1762. doi.org/10.1007/s11277-020-07446-4.

Kossiakoff, A., Seymour, S. J., Flanigan, D. A., Biemer, S. M., 2020. *Systems Engineering Principles and Practice*. Wiley.

Lamprecht, A.-L., Garcia, L., Kuzak, M., Martinez, C., Arcila, R., Del Martin Pico, E., Del Dominguez Angel, V., van de Sandt, S., Ison, J., Martinez, P. A., McQuilton, P., Valencia, A., Harrow, J., Psomopoulos, F., Gelpi, J. L., Chue Hong, N., Goble, C., Capella-Gutierrez, S., 2020. Towards FAIR principles for research software. *Data Science*, 3(1), 37–59. doi.org/10.3233/DS-190026.

Landesbetrieb Geoinformation und Vermessung Hamburg, 2022. HH UDP IoT: This is the HH UDP IoT helm chart repository and cookbook. This is a manual and a software package for deploying a FROST-Server-based SensorThingsAPI IoT-Platform to a Kubernetes-Cluster. https://gitlab.opencode.de/lgvhh/udp/hh-udp-iot.

Lee, E., Seo, Y.-D., Oh, S.-R., Kim, Y.-G., 2021. A Survey on Standards for Interoperability and Security in the Internet of Things. *IEEE Communications Surveys & Tutorials*, 23(2), 1020–1047. doi.org/10.1109/COMST.2021.3067354.

Li, K., Cui, Y., Li, W., Lv, T., Yuan, X., Li, S., Ni, W., Simsek, M., Dressler, F., 2023. When Internet of Things Meets Metaverse: Convergence of Physical and Cyber Worlds. *IEEE Internet of Things Journal*, 10(5), 4148–4173. doi.org/10.1109/JIOT.2022.3232845.

Liang, S., Khalafbeigi, T., van der Schaaf, H., Miles, B., Schleidt, K., Grellet, S., Beaufils, M., Alzona, M., 2016. OGC SensorThings API Part 1 : Sensing Version 1.0. https://docs.ogc.org/is/15-078r6/15-078r6.html.

Material UI SAS, 2024. Material-UI: React components for faster and easier web development. https://mui.com.

Moshrefzadeh, M., Machl, T., Gackstetter, D., Donaubauer, A., Kolbe, T. H., 2020. Towards a Distributed Digital Twin of the Agricultural Landscape. *Journal of Digital Landscape Architecture*. doi.org/10.14627/537690019.

OpenJS Foundation, 2024. Node-RED. https://nodered.org/docs/.

Oracle Corporation, 2024. MySQL. https://www.mysql.com/.

Pallets, 2023. Flask Documentation. https://flask.palletsprojects.com/en/2.3.x/.

Pröll, S., Zangerle, E., Gassler, W., 2017. *MySQL: Das umfassende Handbuch*. Rheinwerk Computing, 3., aktualisierte und erweiterte auflage, 1. korrigierter nachdruck edn, Rheinwerk Verlag, Bonn.

Rathinam, S., 2023. Analysis and comparison of different frontend frameworks. S. Prabhu, S. R. Pokhrel, G. Li (eds), *Applications and Techniques in Information Security*, Communications in Computer and Information Science, 1804, Springer Nature Singapore, Singapore, 243–257.

Sadeghi-Niaraki, A., 2023. Internet of Thing (IoT) review of review: Bibliometric overview since its foundation. *Future Generation Computer Systems*, 143, 361–377. doi.org/10.1016/j.future.2023.01.016.

Specka, X., Martini, D., Weiland, C., Arend, D., Asseng, S., Boehm, F., Feike, T., Fluck, J., Gackstetter, D., Gonzales-Mellado, A., Hartmann, T., Haunert, J.-H., Hoedt, F., Hoffmann, C., König, P., Lange, M., Lesch, S., Lindstädt, B., Lischeid, G., Möller, M., Rascher, U., Reif, J. C., Schmalzl, M., Senft, M., Stahl, U., Svoboda, N., Usadel, B., Webber, H., Ewert, F., 2023. FAIRagro: Ein Konsortium in der Nationalen Forschungsdateninfrastruktur (NFDI) für Forschungsdaten in der Agrosystemforschung. *Informatik Spektrum*, 46(1), 24–35. doi.org/10.1007/s00287-022-01520-w.

Steinmacher, I., Robles, G., Fitzgerald, B., Wasserman, A., 2017. Free and open source software development: the end of the teenage years. *Journal of Internet Services and Applications*, 8(1). doi.org/10.1186/s13174-017-0069-9.

Tzavaras, A., Mainas, N., Petrakis, E. G., 2023. OpenAPI framework for the Web of Things. *Internet of Things*, 21, 100675. doi.org/10.1016/j.iot.2022.100675.

Zhang, M., Yue, P., Hu, L., Wu, H., Zhang, F., 2023. An interoperable and service-oriented approach for real-time environmental simulation by coupling OGC WPS and SensorThings API. *Environmental Modelling & Software*, 165, 105722. doi.org/10.1016/j.envsoft.2023.105722.