CS459 Introduction to Service Computing

PROJECT REPORT

Team AllOps

Luoshan Rosan Zheng 20236421,
Michael Kleefisch 20236416,
Aghiles Gasselin 20236083,

Abstract

Young startups are rich and ideation and drive for innovation to build their vision of the future. Nevertheless, in a high-competition environment on the market, it is often harsh for startups to navigate their goals with limited resources in time, money, and human resources. Furthermore, a lack of well-defined business processes and organizational structure are often causes for failure as the growth of the business itself does not scale with a startup's internal structures. To support startups on their way to success, we developed the platform AllOps, which relieves startups from organizational and bureaucratic tasks such that they are able to focus on their daily businesses.

This report details the development of AllOps, a centralized platform designed to streamline bureaucratic processes in startups. Employing a Service-Oriented Architecture (SOA), AllOps offers a suite of services including daily information overviews, leave request management, IoT-based access management, and finance management. An in-depth elaboration on the architectural design choices are given to illustrate the architectural approach and how SOA aligns with the system goal and value AllOps provides. The report emphasizes the system's user-centric design and the integration of diverse participants like employees, managers, HR, and external entities.

It highlights the benefits and limitations of the platform, emphasizing the importance of scalability, flexibility, and user engagement. Finally, the report discusses team dynamics, project management, and technical challenges encountered, offering insights into the practical application of SOA and new technologies in complex projects.

1 - Introduction

In the modern business ecosystem, startup companies are increasingly recognized as the drivers of innovation, dedicating their efforts to enhance the world we live in. Although their drive for innovation is truly remarkable, startups still face a fair set of challenges, primarily the need to generate revenue in an economical environment characterized by intense competition and rapid market evolution. Especially in the starting phases, startups are forced to keep up with the reduced time-to-market facilitated by existing technology innovations, so that they can survive financially as a certain turnover point needs to be reached first.

For a startup to navigate these turbulent waters towards success, it is critical that it scales accordingly with its growth, which is frequently difficult to achieve. Scaling a business means in this sense not only to satisfy the market's demands but to also grow the business from an organizational perspective regarding its internal structures and defined processes. Failure to do so can result in the startup's downfall, as unscalable systems and processes become ballast to growth. One of the core problems that startups encounter is the lack of well-defined processes, a consequence of the multifaceted roles often assigned to single individuals. This leads to organizational tasks being relegated to the backlog or mishandled, leading to inefficiencies that are hindering the company's growth and sustainability. Furthermore, the tools available to support the organization of a startup are often modular and topic-specific, such as Odoo for time management or Jira for task management. Even though these tools are useful, they can be expensive and are in an enclosed system, creating a fragmented tool-ecosystem, leading to difficult access and integration. Nevertheless, handling bureaucratic tasks, such as spendings documentation, HR management, access management, etc. are critical to a startup's survival but time- and resource-consuming at the same time, which are scarce resources.

To address these challenges, we propose the development of a centralized platform AllOps that brings together all these services handling daily background processes. This platform would not only digitize the bureaucratic load but also provide Business Intelligence (BI) and analytical tools to provide valuable insights for decision-making. Such a solution offers multiple benefits: it saves time, enhances organizational efficiency, and increases employee satisfaction. Moreover, this platform is designed in a user-centric manner to adapt to each startup's unique needs by implementing a Service-Oriented Architecture (SOA). This ensures that the platform can scale with the startup, since the service architecture supports organizational and workload changes. In essence, this platform stands as a beacon of support for startups, streamlining their operations and fostering an environment conducive to innovation and growth.

2 - Background

In our CS459 class, we engaged in an extensive exploration of Service-Oriented Architecture (SOA), a critical architectural model for creating software systems that are adaptable, efficient, and tailored to meet user needs. SOA's emphasis on modularity and integration of diverse services, including Web, IoT-based, and cloud services, aligns perfectly with the goals of our project. The approach allows for the construction of systems that are not only flexible but also capable of evolving with changing requirements, a feature particularly relevant in our dynamic project environment.

Knowing that we would use SOA as a base, we then further delved into microservices, an architectural style representing a significant shift from traditional, monolithic structures. As we have seen, microservices architecture is characterized by its division of software into small, independent services that communicate over well-defined APIs, and offers numerous advantages that are particularly beneficial for our project. This methodology enhances scalability and manageability, essential in handling the complex interactions and multiple components our project entails. Moreover, the independent nature of microservices enables continuous deployment and updates without disrupting the entire system, ensuring a more resilient and robust application, crucial for the real-time demands of our project.

The adoption of microservices is especially relevant given the project's requirement for a scalable, efficient system capable of integrating diverse functionalities, such as the management of employee access through hardware like Arduino boards and fingerprint sensors. The independent yet interconnected nature of microservices allows for each component of our project to be developed, maintained, and scaled individually, ensuring a streamlined and effective integration of both hardware and software components.

Further augmenting our project's technological stack, we embraced the MERN stack for full-stack development. The MERN stack, comprising MongoDB, Express.js, React, and Node.js, was strategically chosen to support the development of a dynamic and interactive web application. This choice was driven by the need for a system that not only handles the intricacies of service integration and real-time data processing but also remains scalable and responsive to user interactions.

Furthermore, we utilized Scrum, an agile project management framework, to effectively manage and streamline our development process. Scrum, known for its flexibility and focus on continuous improvement, allowed our team to adapt to changes quickly and efficiently. The framework involves roles such as the Scrum Master, responsible for

guiding the team and ensuring adherence to Scrum practices, and the Service Owner, who handles external coordination and communication with stakeholders.

3 - Team Member Introduction

For our team, we combined our diverse backgrounds and skills to tackle various aspects of the project, from document writing and coding to presentation planning and team management. Each team member brought unique strengths to the table, allowing us to distribute responsibilities effectively while ensuring that everyone had a role in each part of the project.

Michael, a second-year graduate student in Computer Science, took on the role of maintaining our code repository and takes care of mainly the backend of the system. With his three years of experience as a full-stack web developer at the ITC of RWTH Aachen University, where he specialized in Vue.js and C#, Michael's expertise was invaluable in overseeing the development and integration of our project's software components.

Aghiles, a fourth-year undergraduate student majoring in Software Engineering, was responsible for communication with the TA and professor while also doing development of front end and hardware capabilities of the system. His experience in working on large-scale Angular and Node.js projects at the University Polytechnique Montreal equipped him with the skills necessary for those tasks.

Luoshan, who is pursuing a Master's degree in Information Systems at TUM, managed our Google Drive space for collaboration and took care of the looks and basic structure of our system. Her background in full-stack development and data analytics, honed through various roles in industries like IT, consulting, health, and automotive, enabled her to efficiently put her experience to use.

Together, we agreed that every voice in the team should be heard, with final decisions made through consensus. This approach ensured a democratic and collaborative environment, preventing any form of team-internal dictatorship, and allowing us to leverage each member's strengths towards the successful completion of our project.

4 - AllOps System

Presenting AllOps, a unified platform designed to simplify and streamline operational processes for startups and small enterprises. Integrating essential functionalities like employee management, cash flow tracking, and security monitoring into one

user-friendly interface, it significantly reduces the administrative burden. Incorporating IoT technologies for enhanced security, AllOps offers a customizable and scalable solution, enabling businesses to focus more on their core activities and less on administrative tasks.

4.1 Goals

The primary SOA goal of our AllOps platform is to streamline and reduce the time startups spend on non-revenue-generating, bureaucratic tasks, thereby cutting labor costs and increasing operational efficiency. Centralization is key in this approach, with all necessary functionalities for various business processes provided within a single, integrated application. This reduces the reliance on multiple independent tools and fosters a more structured and standardized execution of tasks. Additionally, the platform aims to enhance employee satisfaction and efficiency by providing tools for rationalized decision-making and a comprehensive overview of daily tasks. By digitizing and standardizing these processes, AllOps seeks to increase the overall efficiency and productivity within startup environments, a goal that is reflected in its design and functionalities.

4.2 - High-Level Design

The high-level design of AllOps, our project's system, revolves around creating a unified, customizable platform tailored for young businesses, particularly startups and small enterprises. Integrating essential business services, it facilitates daily organizational processes and enhances company security through IoT technologies like Arduino devices with sensors for access monitoring. The platform supports various business processes, including employee management, cash flow management, security and access monitoring, goal management, and analytics for decision-making. These processes are intricately designed with subprocesses and activities streamlined within the system. Furthermore, AllOps is cloud-deployed, ensuring scalability, high availability, and flexibility. The platform can adapt to the evolving needs of businesses, with services independently deployed and accessible through a common user interface, aligning with the dynamic nature of startups and their growth trajectories.

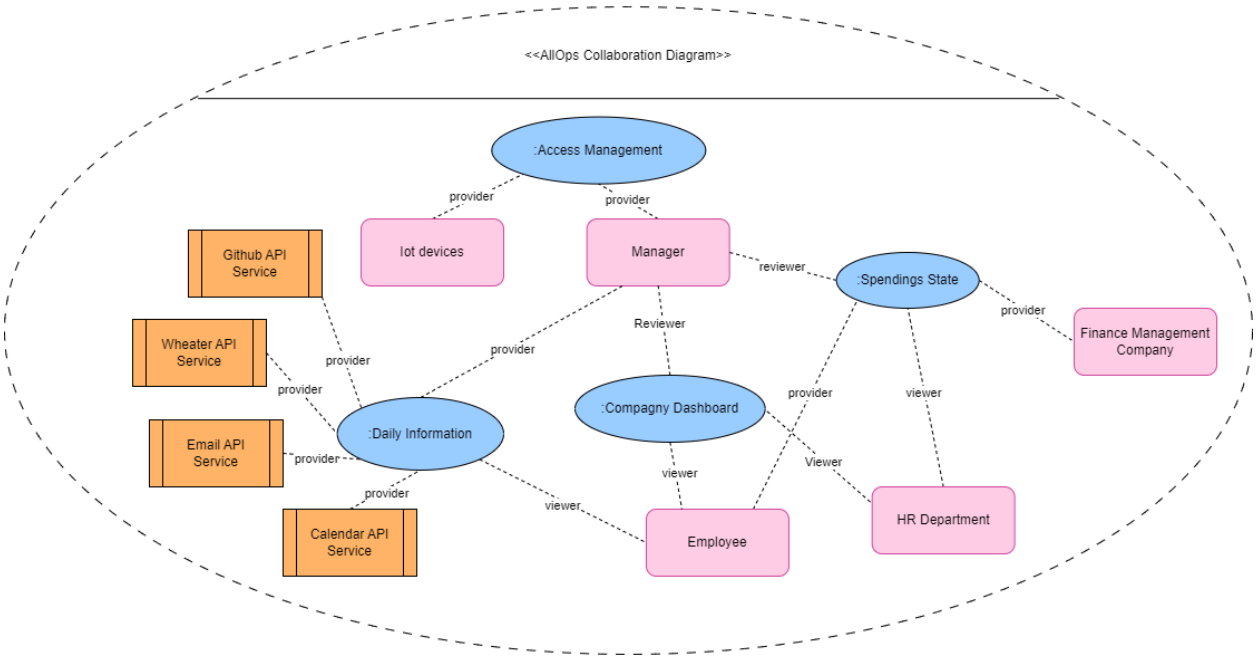

Figure 1: Collaboration diagram of AllOps

4.3 - Low-Level Design

For the implementation of the project, a decision was made to use a mono-repository, which is managed through the version control system GitHub (https://github.com). The use of a mono-repository is advantageous as the team consists of three developers, providing sufficient visibility to coordinate shared work. In addition, the project scope could be estimated based on the previous architecture and design decisions, so that a joint deployment of the applications turned out to be advantageous.

We have limited our choice of languages and technologies for this project. Both the frontend and backend are implemented in TypeScript (https://www.typescriptlang.org). For the frontend, we are using the React framework (https://react.dev), and the web services are realized with Node.js (https://nodejs.org) and Express (https://expressjs.com). To securely and persistently manage the collected data, we have opted for the NoSQL database management system MongoDB (https://www.mongodb.com). Specifically, we have chosen a cloud-based solution to ensure that the data is accessible to all team members at all times.
As a small development team, we intentionally limited the number of technologies used, facilitating seamless collaboration.

In the implementation of AllOps, a Service-Oriented Architecture (SOA) was chosen, implemented through web services. This architecture provides the necessary loose

coupling, allowing for a modular composition of the AllOps application. At the same time, SOA enables the use of external services without the need for new implementations, thereby fulfilling the principle of reusability.
Fundamentally, AllOps is a web application that offers various functionalities, suitable for startups to handle bureaucratic processes. Thus, the core system is a web application. In implementing various functionalities, existing services were leveraged during the Identification Phase of the Service-Oriented Modeling and Architecture (SOMA) Lifecycle. In addition to existing services, this phase also defined additional services needed for the implementation of AllOps. The services identified through Goal-Service Modeling (GSM) are depicted in Figure 2 in an initial relation to each other.

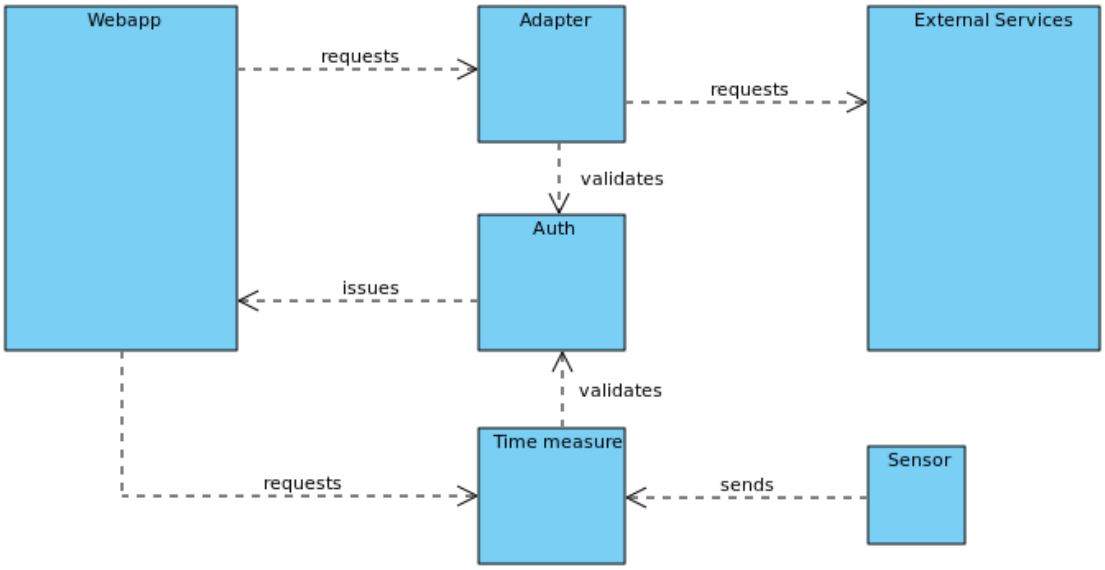

Figure 2: High level implementation view

To access the publicly available APIs of external web services, a certain authentication through a token is required. Manual authentication of each user is impractical for the use case in AllOps, which is why an Application Server (Adapter in Figure 2) is used. Acting as an intermediary between the web application and external web services, the Adapter-Service is used by the users of the web application to utilize the functionalities of these external services through AllOps without having to register with each external web service beforehand. This middle-tier system has been additionally enhanced with caching at suitable points. The caching ensures reduced load on external interfaces and enables a faster response to the web app's requests.
Since the interfaces of our web services, following the SOA principle of Network orientation, are publicly accessible over the Internet, they must be protected to limit the use of non authorized persons, because the services of these applications should only be available to AllOps users. For this reason an authentication layer has been added. The implementation was realized by creating a service which specialized on Authentication (Auth in Figure 2). Upon the initial opening of AllOps, the user is

prompted to authenticate. This authentication is done through a password, which is transmitted as a hash to the Auth service. The Auth service compares the password-hash with the stored hash in the user database and, upon successful authentication, issues a JSON Web Token (JWT). Such a token is a string that encodes data in JSON format and is protected against manipulations by signing the token with a signature. This token is securely stored on the website and attached to every request to the web services. Through this process, the user can authorize themselves to another system, simultaneously, the JWT performs user authentication and transmits the necessary user data to the service.

In addition to the software-based systems, there is also a hardware system that handles time tracking in the startup through a fingerprint sensor. For the implementation of this service, a prototype was developed that uses a Raspberry Pi to operate the sensor (see figure 7) and forward the data to the Time Measure Service (see Figure 2).

The idea is to install a fingerprint sensor on each side of the door. This allows distinguishing between joining and leaving the startup. When activated, the Raspberry Pi sends a request to the service, appending the measured fingerprint hash and timestamp. Additionally, a flag indicates whether it is an exit or entry. The Time Measure Service compares the fingerprint hash with the stored database and responds in the case of a successful match. To ensure the security of message exchange, asynchronous encryption is used. Upon receiving the response, the Raspberry Pi is signaled to open the door. Figure 3 shows a typical interaction sequence between all the involved participants.

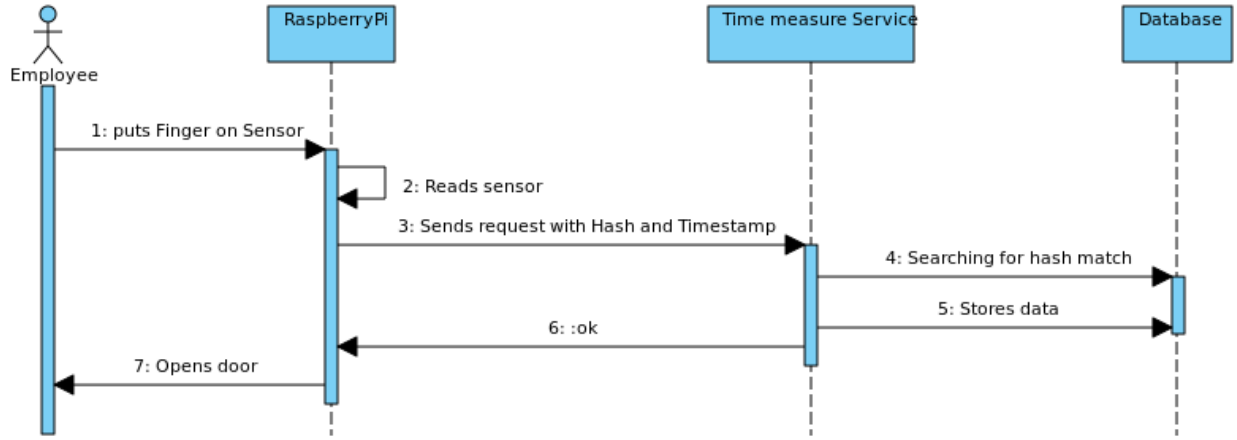

Figure 3: Sequence diagram of opening the door at Startup

Upon request from the web application, the time measure service analyzed the collected data and response with information about the working hours of each employee.

4.3 Results

After completing the first version of AllOps, various views were implemented. The login page shown in Figure 4 is displayed upon the initial login or after logging out. Users are prompted to authenticate themselves using a username and password set during registration.

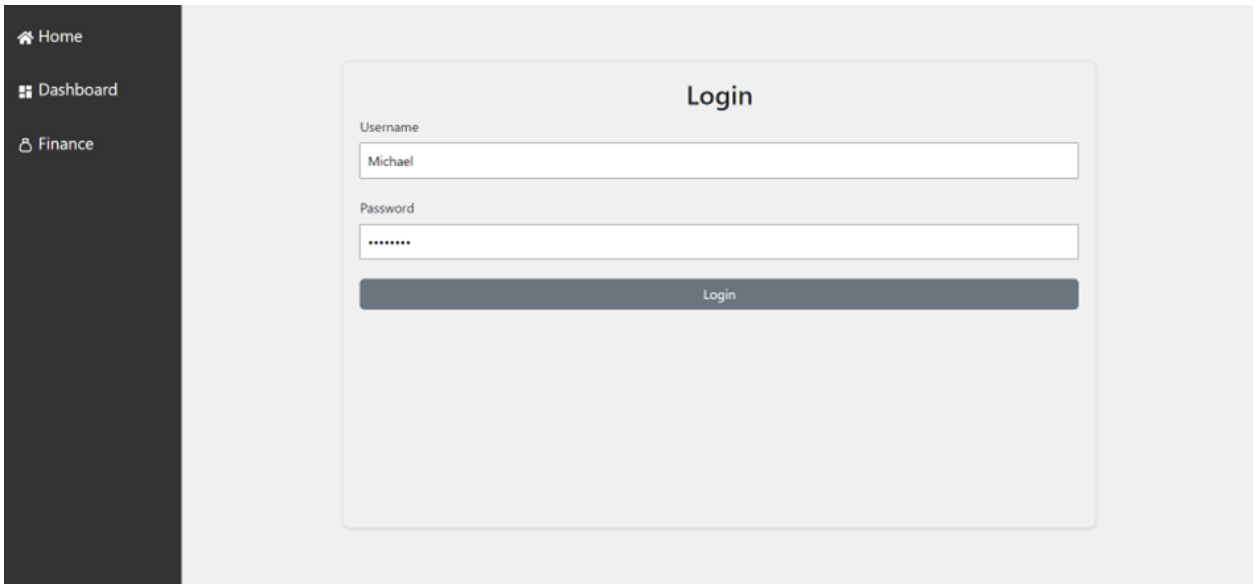

Figure 4: Login page of AllOps

After logging in, the homepage opens, providing a consolidated overview of upcoming events and tasks. In addition to user-specific information, general data such as weather information is also provided. Figure 5 shows the homepage with possible content.

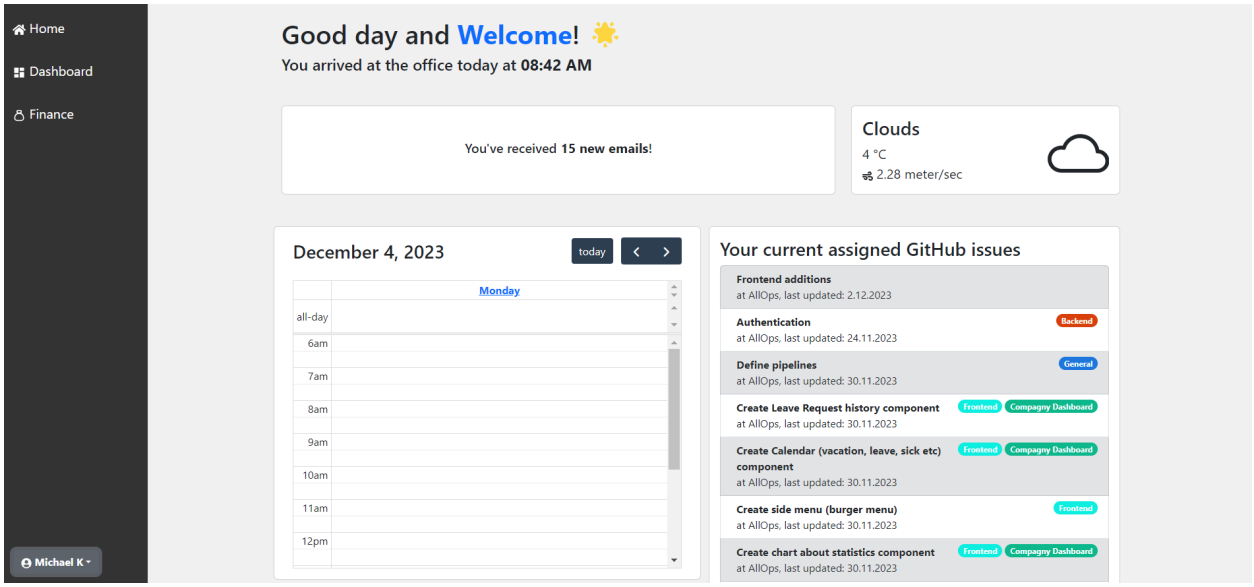

Figure 5: Startpage of AllOps

For various use cases, navigation on the left side can be used to switch between different main categories. In addition to many forms for submitting requests, AllOps also provides an overview accessible only to the company's management. The view depicted in Figure 6 as an example offers an overview of the startup's statistics. For instance, revenues and expenses are visualized, and there is a calendar overview of absent employees.

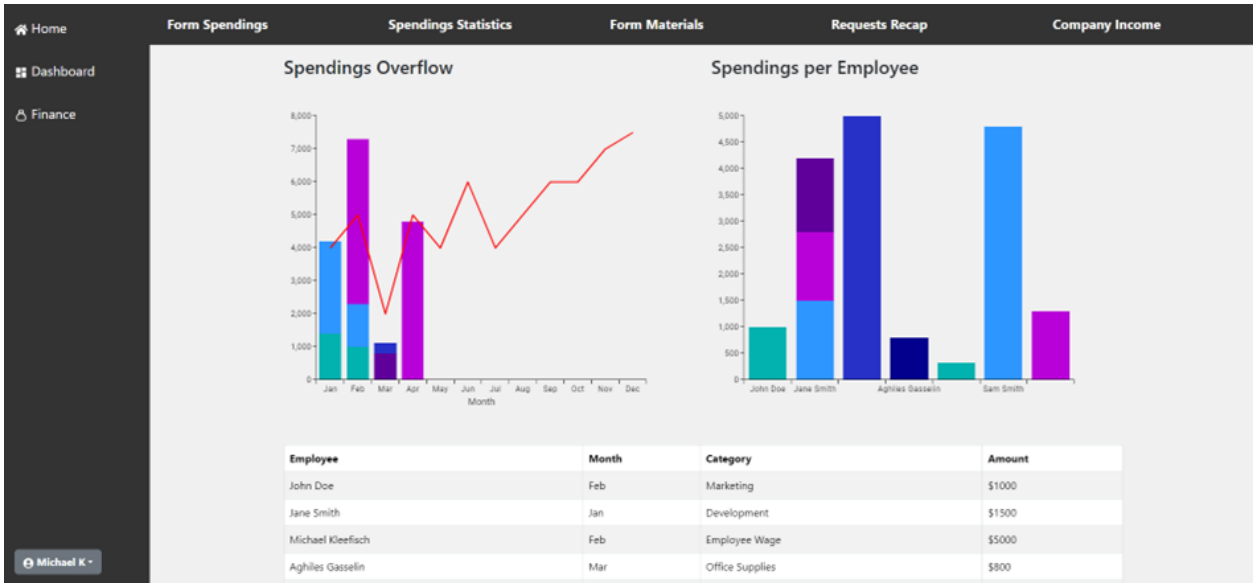

Figure 6: Spending statistics of a sample startup

Regarding the automatic time tracking, a prototype was developed that confirmed the feasibility of the project. Figure 7 shows the prototype, which consists of a Raspberry Pi and a connected fingerprint sensor. The door opening is indicated by the illumination of an LED.

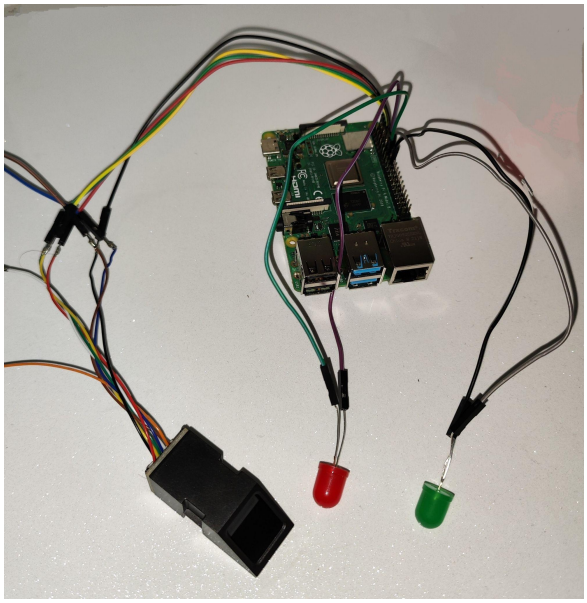

Figure 7: Fingerprint sensor and led for door opening and logging

5 - Discussion

In this project, the system's integration of various components and services into a single cohesive environment is a notable advantage. This integration aligns well with the SOA design concepts emphasized in the course, particularly the focus on creating a user-centric system that integrates a variety of services such as Web, IoT-based, and cloud services. The centralization of operations not only makes the system user-friendly and efficient, but also significantly simplifies management processes, which is particularly beneficial for startups and small organizations seeking a fast organizational boost. However, the system is not without its cons. The lack of real-world deployment limits our ability to test its practical efficacy and user experience in actual scenarios. Additionally, there are hardware limitations, such as the absence of actual door unlocking mechanisms, indicating challenges in integrating physical components with the software architecture. There's also a pressing need to integrate more services to enhance the system's functionality and cater to a broader range of user needs.

Beyond these immediate pros and cons, other important considerations include the potential for scalability, inherent in its SOA and microservices architecture, but yet to be fully tested in a deployed environment. Security is another critical concern, especially with the integration of hardware like fingerprint sensors, necessitating rigorous attention to data security and privacy. Furthermore, the integration of various services and hardware components could potentially complicate maintenance and troubleshooting, requiring specialized skills and resources, an important factor for future development and improvements. These aspects - the advantages of integration and centralization, the challenges in deployment and hardware limitations, and additional considerations like scalability, security, and maintenance complexity - collectively paint a comprehensive picture of our system's potential, its current limitations, and the areas that need attention for its enhancement and successful implementation in real-world scenarios.

7 - Conclusion

To summarize the results within this project, we have developed the platform AllOps to facilitate bureaucratic processes in young businesses for more efficiency in daily operations through a centralized platform by applying a SOA-based approach. With a business-centered perspective we have analyzed crucial business processes that are exposed to inefficiencies in the real world. The services we implemented to relieve our user’s pain points cover a user overview for daily information to kick-off a workday, leave request management, an IoT-based access management service, and finance management. While the implementation details stand on their own, this report focuses on the motivation, design, representation, and discussion of our system. We have

illustrated the participants of our system, which are members within an organization like employees, managers, and a HR department, as well as outsiders such as a finance management company and IoT devices, and modeled the way they interact with each other through services. Afterwards, we elaborated on the implementation details in order to realize our platform while also providing extracts from our end product. Finally, we evaluated our system and approach to determine our system's benefits but also highlight its shortcomings.

Building on the summary of results for the AllOps platform, our project's journey has imparted several key lessons that have shaped our understanding and approach to developing such a comprehensive system.

One of the primary learnings was that the creation of UML-diagrams helped us a lot in visualizing and understanding the system, its components, and their relationship with each other. Apart from the architectural understanding, it helped us to implement the system in a user-centric manner by having a clear vision of how the participants interact with each other through the services. All in all, it really helped us in our team work, however, it's important to note that while beneficial, the process of creating these diagrams were time-consuming, reflecting a trade-off between clarity and efficiency.

Our application of SOA was a learning curve in itself. We explored both top-down and bottom-up approaches, understanding the nuances and applicability of each in various contexts of our project. This exploration was necessary in refining our approach to SOA, enhancing the platform's scalability and flexibility by also defining rationales for our design and architectural choices.

The project also gave us the opportunity to learn and apply new technologies such as React and the Adafruit library and broadened our technical skills, while showing us how different technologies can be synergized in a complex project. Utilizing existing code and libraries played a significant role in accelerating our development process and also gave us the opportunity to explore open APIs. By leveraging these resources, we were able to focus more on the unique aspects of our project rather than reinventing the wheel, thereby making development faster and easier.

Finally, managing expectations among teammates, especially regarding technology choices and design decisions, was a critical aspect of our project. It involved constant dialogue and compromise, ensuring that all team members were aligned and contributing effectively towards our common goal. In addition, working as a team within this relatively short time frame of roughly three months underscored the importance of collaboration and effective communication. It highlighted how diverse perspectives and expertise can leverage our capabilities to solve complex problems, but at the same time that if done wrongly, it can also make collaboration more difficult.

In conclusion, while AllOps successfully addresses several operational inefficiencies in young businesses, our journey through its development was equally about learning and adapting. These lessons, ranging from technical skills to team dynamics and project management, have been invaluable, providing us with a richer perspective and a robust foundation for future endeavors.

8 - Future Work

As we look towards the future of our platform, our vision encompasses a comprehensive expansion and enhancement strategy, focusing not just on new services, but also on technical robustness and adaptability across various industries.

Our current platform includes solely the most important processes within a business context, thus the integration of a suite of additional services that cater to the evolving needs of modern businesses is crucial. These will include, but are not limited to, a Carpool Management system for efficient fleet management, an Objective Key Results (OKR) Management tool for performance tracking, and a centralized repository for Company Resource Documentation. Additionally, systems for handling office requests and resource requests for office supplies and work-related resources may be developed to increase employee happiness and employee participation. These examples represent just a fraction of the potential expansions, with plans to explore further services, especially in the realm of human resources processes.
To follow the SOA strategy of keeping the services aligned with business processes and providing a user-centered platform it is crucial to analyze the customer’s needs through thorough customer analysis. This will involve actively seeking feedback and insights from users to identify additional needs and areas for improvement. Such a customer-centric approach will inform the continuous refinement and enhancement of both existing and new services.
Considering the current state of the platform, the platform might be more tailored towards technical startups. Recognizing the diverse nature of the business world, the platform's applicability should be extended beyond the tech industry. This expansion will involve customizing the platform to meet the specific needs of different industries, making it a versatile tool for a wide array of businesses.

In regard to the technical setup of the platform, different environments dedicated to thorough testing are necessary to ensure the reliability and effectiveness of these services. These environments facilitate various test types, such as unit and integration tests, crucial for maintaining the platform’s integrity. Parallelly, the implementation of a Continuous Integration/Continuous Deployment (CI/CD) pipeline will streamline the development process, enabling quicker and more efficient updates and deployments.

This is especially crucial as the platform needs to adapt to the continuously changing business environments of the platform users.

By considering and implementing the named future work needs, the platform will not only become comprehensive and efficient but also adaptable and user-friendly to cater to the dynamic needs of businesses and help startups not only grow from a financial aspect, but also scale from an organizational and process-related perspective.