The Journal of Systems and Software 216 (2024) 112106

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

In practice

Pandemic startup software engineering: An experience report on the development of a COVID-19 certificate verification system ☆

Richard May [a], Niklas Baron [a], Jacob Krüger [b,*], Thomas Leich [a]

[a] Harz University Wernigerode, Germany
[b] Eindhoven University of Technology, The Netherlands

ARTICLE INFO

Dataset link: https://doi.org/10.5281/zenodo.10968281

Keywords:
COVID-19
Pandemic
Startup
Verification system
Empirical study

ABSTRACT

The COVID-19 virus has caused a global pandemic that has heavily impacted daily life. Rapid advances in testing and vaccinating led to an additional use case besides the well-known contact-tracing apps: certificate-verification systems. Verification systems are often commissioned by local authorities to enable more public life, and are often developed by smaller organizations or startups. So, the development of verification systems differs from other software projects, featuring interesting and unique properties. In this article, we present an experience report on the development of one verification system by a German startup, focusing on three properties: working in a pandemic, developing a product for handling a pandemic, and the startup context. To this end, we surveyed nine startup developers and analyzed the results with two experts from the startup. We found that the developers focused on fast delivery to cope with the time pressure of releasing the verification system, which is why some phases of typical development processes were hardly carried out. As a result, while the verification system is successful, we also identified negative effects of the properties (e.g., programming mistakes, well-being). We discuss our findings to guide researchers and practitioners in preparing for software engineering in future emergencies.

Editor's note: Open Science material was validated by the Journal of Systems and Software Open Science Board.

1. Introduction

The outbreak of the COVID-19 virus in 2019 has caused a global pandemic that threatens the health of individuals all around the world, thus challenging all aspects of our daily life. With tremendous efforts and speed, medical solutions (e.g., antibody tests, vaccines) have been developed, and laws have been enacted to limit the impact of COVID-19 (Velavan and Meyer, 2020; Ciotti et al., 2020). Software engineers have developed various (mobile) apps that help track contacts and ideally break infection chains (Gupta et al., 2021; Abuhammad et al., 2020; Garousi et al., 2022). These apps have typically been developed in large organizations, usually on behalf of governments, for instance, the Aarogya Setuas app in India, the NHS COVID-19 app in England and Wales, the Corona-Warn-App in Germany, or Covid Alert in Canada (Wymant et al., 2021; Erikson, 2021; Munzert et al., 2021; Seto et al., 2021). In addition, other systems have been created to check the quarantine status of an individual, for example, the Alipay Health Code app in China (Liang, 2020; Morley et al., 2020).

The rapid advances in testing and vaccinating have led to new use cases at the end of 2020 that exceed contact tracing apps. Namely, national and local governments have sought solutions for enabling more and more public life, typically by relaxing restrictions for completely vaccinated individuals or those having a recent negative COVID-19 test (Piguillem and Shi, 2020; Manabe et al., 2020; Ward et al., 2022). In this regard, software engineers developed so-called *digital certificate-verification systems* (we refer to verification systems). Such verification systems allow their users (e.g., individuals, event organizers) to manage and verify certificates (e.g., of the European Union) as well as test results to allow access for healthy individuals—thus enabling public life (Karopoulos et al., 2021). Typically, verification systems build on a QR code (Wang and Jia, 2021; Wahsheh and Al-Zahrani, 2021) that includes an individual's contact details, data for verifying their identity, and the certificate of a negative test, complete vaccination, or recovery. Interestingly, while tracing apps and certificates have usually been initiated at national or even higher level, verification systems have often been commissioned and deployed by more local authorities, such

☆ Editor: Marcos Kalinowski.

* Corresponding author.
E-mail addresses: rmay@hs-harz.de (R. May), nbaron@hs-harz.de (N. Baron), j.kruger@tue.nl (J. Krüger), tleich@hs-harz.de (T. Leich).
URL: https://jacobkrueger.github.io/ (J. Krüger).

https://doi.org/10.1016/j.jss.2024.112106
Received 3 January 2023; Received in revised form 30 October 2023; Accepted 20 May 2024
Available online 23 May 2024

as cities, counties, or states. Furthermore, they are often developed by smaller organizations or even startups, sometimes based on their own motivation. Consequently, the development of verification systems can exhibit unique properties that differ from the well-known tracing apps; or any other system.

In this article, we present an experience report from one German startup that developed a verification system for a federal state. This case is highly interesting, due to its unique properties (cf. Fig. 1), namely: (i) developing in a pandemic, (ii) engineering a product for managing the pandemic, and (iii) the startup context. We refer to the interaction of these three properties as *pandemic startup software engineering*. There are few opportunities to understand the challenges developers face in such an emergency situation, which makes this experience report invaluable to obtain insights for future emergency situations (e.g., pandemics, humanitarian crises, natural disasters). Systems for handling emergency situations will likely be needed more often in the future, while they do not follow typical business models (i.e., they represent public goods) and may rapidly be discarded (i.e., when the crisis has been solved); posing novel challenges for organizations and their developers (i.e., regarding funding). To achieve our goal of exploring the development process and developers' challenges, we conducted a qualitative survey with nine of the startup's developers. We analyzed the responses in cooperation with two experts from the startup who did not participate in the survey (the first two authors of this article).

In detail, our contributions are:

- We analyze how the startup's developers were impacted by the three properties (startup, pandemic development, pandemic product) when developing the verification system.
- We collected the developers' experiences of developing the verification system to identify good practices and challenges.
- We discuss how organizations and researchers can build on our insights to manage future emergencies.
- We publish an open-access repository with the summarized anonymous responses to our survey.[1]

Our results indicate that the startup's developers had to take on different roles and experienced high pressure. Particularly the startup context was perceived as the property with the highest impact, positively (e.g., freedom) as well as negatively (e.g., missing experience). We hope that our insights will help organizations and researchers in planning for, mitigating, and handling future emergencies.

The remainder of this article is structured as follows. In Section 2, we describe the related work and setting of our experience report. Then, we describe our study design in Section 3. We report and discuss the results of our study in Section 4 before discussing potential threats to its validity in Section 5. Finally, we conclude this article in Section 6.

2. Case setting

In the following, we provide an overview of the related work and motivate our experience report. Moreover, we describe the startup we investigated and its verification system.

2.1. Motivation, background, and related work

COVID-19 has led to lockdowns all around the world during the last few years, imposing different degrees of restrictions on individuals' ability to move around freely. For instance, such lockdowns involved stay-at-home orders, mask-wearing, and allowing gatherings only for small groups or families. However, lockdowns fundamentally interfere with most democracies' basic rights, and could only be imposed due

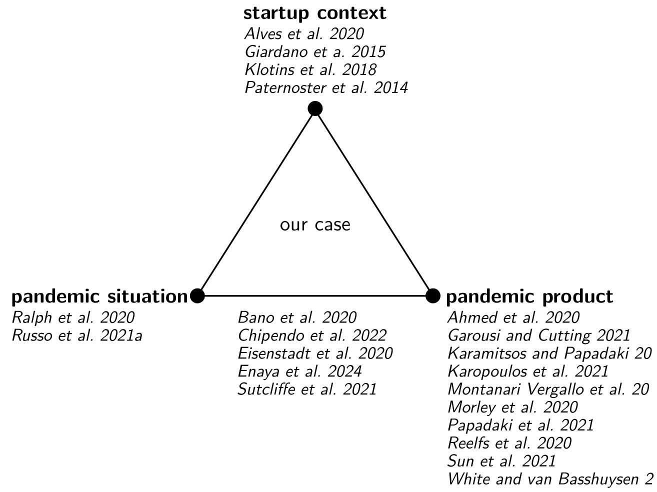

Fig. 1. The three properties covered in our experience report.

to the emergency situation. For this reason, governments and organizations searched for possibilities to at least partly restore public life and reduce the inference with basic rights. At first, contact-tracing apps (Gupta et al., 2021; Wymant et al., 2021; Erikson, 2021; Munzert et al., 2021; Liang, 2020; Morley et al., 2020) were developed to identify and break up infection chains as well as to check quarantine statuses. Later, those apps have been extended to allow users to add digital certificates that verify a recovery, complete vaccination, or recent negative tests, for instance, via the French health pass app (Ward et al., 2022). Software engineering researchers worked on supporting the development of such systems, analyzing the systems' properties, and researching the pandemic's impact on developers.

Next, we describe such related work in more detail and compare it to our experience report, for which we depict a summarizing overview in Fig. 1. As we can see in Fig. 1, our case combines a startup context with the development of a pandemic product for managing the respective pandemic. Similar cases are likely to arise in the future for different pandemics or other emergency situations. Yet, existing research has mostly studied these properties in isolation or the combination of two (references in Fig. 1). Our experience reports provides complementary insights on how one startup operated under these conditions, intending to prepare other organizations for future emergencies.

Analyses of COVID-19 Apps. We define apps developed to manage the COVID-10 pandemic as *pandemic products*, meaning that these represent software systems developed under huge societal pressure and with immediate urgency to protect peoples' health or even lives. Particularly the urgency with which such health-related products were needed represents a unique property. There are several studies on the technicalities of COVID-19 apps (mainly contact-tracing apps), either focusing on a single app or on comparing multiple to each other. Reelfs et al. (2020) present and analyze the basic concepts of the German Corona-Warn-App. The authors focus on the app's hosting infrastructure and its generated traffic, especially during local COVID-19 outbreaks. Garousi and Cutting (2021) analyzed the UK's three contact-tracing apps. More specifically, they assess the apps' popularity and users' perception based on app reviews from the app stores. White and van Basshuysen (2021) describe the declining role of contact tracing-apps during the second wave of COVID-19. They primarily criticize that these apps did not inform users quickly enough about infections and that there are also various concerns about data privacy. Morley et al. (2020) focus on privacy, equality, as well as fairness in digital contact tracing. In this context, they introduce ethical guidelines for such apps. Sun et al. (2021) assess 20 apps in the context of security and privacy by applying their own assessment tool. Moreover, they

[1] https://www.dropbox.com/scl/fi/rnt974tqsonwdkdqklkla/survey_results.csv?rlkey=0g28qugu59m2m4j39juic88ub&dl=0 Will be put on Zenodo.

offer concrete security- and privacy-related guidelines, and identify gaps between system requirements and actual performance. Ahmed et al. (2020) provide an overview of contact-tracing apps, especially regarding their architectures as well as security and privacy vulnerabilities. Montanari Vergallo et al. (2021) analyze COVID-19 certificates, highlight their critical aspects and the importance of such certificates in preventing the spread of infections. Karamitsos and Papadaki (2021) as well as Papadaki et al. (2021) focus on privacy concerns related to COVID-19 certificates. Specifically, they propose blockchain techniques to implement secure COVID-19 apps. Finally, Karopoulos et al. (2021) survey 54 techniques and apps related to digital COVID-19 certificates, focusing on the general architectures and varying privacy concerns between countries. In contrast to such studies, *we focus on the development process of a COVID-19 app* (i.e., verification system) instead of its architecture, specific design details, or a systematic comparison of COVID-19 apps.

Engineering COVID-19 Apps. While developing a *pandemic product* already causes unique pressure, we argue that this is further amplified by the fact the respective developers are also exposed to the *pandemic situation*. In particular, development processes, communication, or developers' behaviors may change, and thus deviate from established practices. Only few studies report on certain aspects of the engineering processes for COVID-19 apps, for instance: Bano et al. (2020) emphasize the impact of user requirements in the context of tracing apps. More specifically, the authors compare the evaluation of the apps based on the fulfillment of user requirements and app features derived from these. Sutcliffe et al. (2021) study the impact of values on requirements and software engineering for COVID-19 apps, especially the UK's King's/Zoe app and the NHS COVID-19 app. They conclude that values play a significant role in the success and acceptance of these apps, but there are also reasons why they failed, such as security and privacy issues. Eisenstadt et al. (2020) developed a prototype COVID-19 app for verifying vaccinations and rapid tests. They discuss their proof of concept regarding its architecture, design details, as well as legal regulations and privacy concerns. Chipendo et al. (2022) report on the Trusted Travel platform developed for the verification and authentication of COVID-19 results in Africa. Specifically, the processes for laboratories are described in the context of the platform. Closest to our experience report is the work by Enaya et al. (2024), who analyzed the development of the German Corona-Warn App at SAP. Due to the similar nature (pandemic product, pandemic situation), there are some shared findings. Still, we not only provide supportive evidence from another system, we also shed light into startup development—which differs from development processes at large companies like SAP. In contrast to such studies that research requirements and user perceptions, *we focus on the actual software development of a verification system in a startup and the involved developers*.

Empirical Studies on Pandemic Software Engineering. Emergencies like the *pandemic situation* impact all people, including software developers. However, how developers are impacted in their work or daily life, and within the specific context of developing a pandemic product may exhibit unique properties. So far, researchers have studied the former: effects of the pandemic on software developers in general. Ralph et al. (2020) conducted a survey among 2225 international developers to analyze how the pandemic impacts their well-being and productivity. They found that there is a strong negative effect and that different individuals need support in different ways by their organizations to minimize these negative effects. Russo et al. (2021a) conducted a study with software engineers to analyze the typical working day in home office.f The authors identified that software engineers do not work more from home, but the time spent was unrelated to their well-being and productivity. In contrast to such studies, *we focus on more qualitative insights on the development of a pandemic product and the involved developers*—but some of our findings are naturally in line with these studies.

Startup Development. Startups are newly founded companies with a business idea that has great potential to scale, often starting within immature or non-existing markets. We can clearly map this *startup context* onto our case company in the COVID-19 pandemic: That company was founded anew to develop a verification system, which was a completely new type of system with the potential to scale towards national or even international use. Despite this potential, that startup also faced the typical issues of uncertainty whether it would succeed and whether the business model was sustainable (e.g., at the end of the pandemic). Recent literature reviews show that many works report on startup software engineering (Paternoster et al., 2014; Klotins et al., 2018). For example, Giardino et al. (2015) conducted interviews with developers in startups to extract the specifics of startup software engineering. In particular, the authors extracted properties, such as speed-up developing, evolutionary deployment, low product quality, and lack of resources. Alves et al. (2020) presented a study on requirements engineering in startup environments. They conclude that these analyses are usually conducted in a pragmatic and informal manner. In contrast to such studies, *we focus on a startup in an emergency situation (i.e., the COVID-19 pandemic) that developed a system for managing that emergency*. Consequently, some of our findings are similar to such studies, but our case exhibits different properties regarding the pandemic situation and product, allowing to study how startup developers can deal with these special properties.

Our Case. While COVID-19 testing, contact tracing, vaccinations, and certificates were a step towards allowing more freedom to individuals, existing contact-tracing apps typically displayed certificates only. However, they usually had no support for test stations, laboratories, event organizers, or authorities to manage data associated with the certificates automatically. For this purpose, local authorities and COVID-19 test stations in Germany started to commission smaller organizations (or bought usage licenses from them later on) with developing verification systems.

The startup we report on was commissioned by a local authority to develop a verification system for a German federal state, and started to sell licenses to other authorities later on to scale its business. This case is strongly different from the related work because it combines the three properties of *pandemic startup software engineering* that have been covered mainly independently before, as we display in Fig. 1. First, the verification system was developed in an *emergency situation*, more specifically, the COVID-19 pandemic. We argue that this can fundamentally change how developers and teams can interact with each other or stakeholders. Second, the verification system represents a *pandemic product* intended to help manage the pandemic, which has severe consequences. For instance, there is high time pressure to release the system, the system will likely be discarded at some point in the future (i.e., when the pandemic is resolved), it represents a public good (i.e., cannot be commercialized like typical systems), and it involves sensitive personal data (e.g., contacts, health status). Finally, this system has been developed in a company that was founded as a *startup* instead of a large organization. Startups drive highly innovative products, and thus are an ideal setting to develop such a novel system fast. Still, they also exhibit highly different and unconventional work practices compared to other organizations (Paternoster et al., 2014; Klotins et al., 2018; Giardino et al., 2015). Consequently, we provide insights into a unique combination of the three properties, which have previously been investigated more independently (cf. Fig. 1).

To support organizations and developers in other emergency situations, we need to understand such properties' impact on, for instance, development processes and developers' mental health. By qualitatively examining the details of this case, we aim to guide other organizations on how to develop systems for managing emergencies. Arguably, such situations will occur more frequently in the future, not only in the form of pandemics (e.g., monkey pox), but also as humanitarian crises (e.g., due to wars, consequences of climate change) or natural disasters

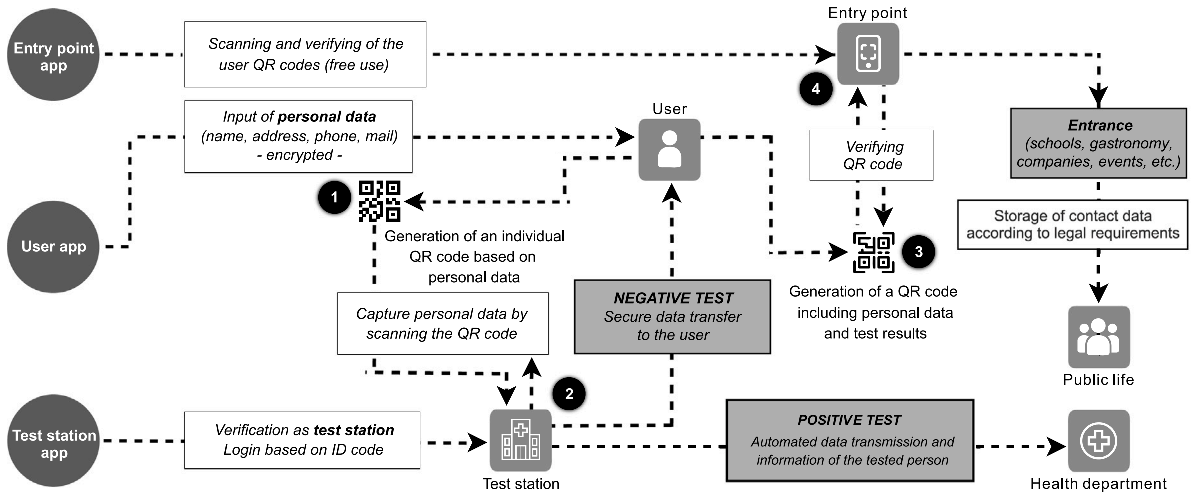

Fig. 2. Design of the verification system.

(e.g., floods, hurricanes). For this reason, we argue that even though we provide only a single case, our experience report is of utmost interest to the research community, since the opportunities to systematically elicit such insights are rare.

2.2. The startup

The subject company was founded in October 2020 as a startup, meaning that it started with a small, flexible team that focused on one innovative idea, while also taking a high risk of failure to fill a market niche caused by the COVID-19 pandemic. The key purpose of the startup was to develop a verification system that can store encrypted digital certificates and verify them in a tamper-proof way. By connecting all involved parties (i.e., individuals, test stations, health departments, event organizers), the system was intended to enable more public life, such as visiting cinemas or museums. Moreover, the system should be able to connect with existing contact-tracing apps. Ensuring that only verified (i.e., healthy, less infectious) individuals can enter events significantly reduces the risk of further COVID-19 infections. Initially, the startup involved four members: (i) a managing director, (ii) a project manager, (iii) a front-end developer, and (iv) a back-end developer.

In March 2021, the system was deployed in a one-time test run for monitoring the impact of mass testing in outdoor food-service areas at a wedding in a county with around 200,000 inhabitants. Since this test run was successful, further cities, counties, individual authorities, and test stations became interested in the system; significantly increasing the number of users and allowing the startup to scale up. Precisely, the system reached a maximum of 50,000 daily downloads at the beginning of April 2021, with weekly averages of around 12,000 downloads and users. To handle the growing interest, the startup expanded over the next few months to increase its capabilities, for instance, for development, design, and customer support. Moreover, the startup joined with a contact-tracing app provider in April 2021 to extend the functionalities of its system and provide better integration with existing COVID tracing apps in Germany. In December 2022, the verification system reached over 2 million downloads and processed about 20 million rapid tests—with the startup involving 17 members in three departments due to its rapid growth: execution and finance (2), operations (7), and software development (8).

The startup followed an agile-like development process that is typical for startups, building on three pillars:

1. Technologically, the system was developed using the hybrid development framework Flutter[2] and the JavaScript runtime Node.js.[3]
2. For its project management, the startup used Kanban with notion.so,[4] including feature and issue boards.
3. To organize the development, daily meetings based on agendas have been installed.

However, due to the pandemic situation (e.g., lockdowns) and a team distributed across Germany, meetings were held through video conferences from home office. So, the startup could also take advantage of employing freelancers for temporary support without interference with its newly established development process.

2.3. The verification system

From the beginning, the verification system was designed by involving all relevant stakeholders: individuals, test stations, health departments, and event organizers (referred to by entry points). In Fig. 2, we depict the overall structure of the system and how it is used by each stakeholder. For clarity and simplicity, we omit details and refer only to the primary use case: verifying tests at test stations (vaccinations and recovery certificates are verified identically). On a high level, the system consists of three hybrid applications available for Android and iOS: (i) a user app, (ii) a test station app, and (iii) an entry point app. Next, we describe the overall workflow of the system based on the four major steps we highlight in Fig. 2.

❶ Generate Personal Code. Initially, the user app generates a personal, encrypted QR code (AES-256) based on the individual's locally stored personal data. This code includes the individual's contact details (e.g., address), data for verifying their identity (e.g., identity card number), and a placeholder for test results. The code serves as a badge for verifying the individual's health status.

❷ Perform COVID-19 Test. The test station app can read and decrypt the encrypted QR code of an individual to access their data. For this reason, not every individual or organization can use the test station app. Precisely, only the responsible health department issues QR codes that enable accounts for the test station app, limiting its use to licensed authorities. The results of a COVID-19 test can be entered into the test station app, which transmits them to the tested individual in an

[2] https://flutter.dev/.
[3] https://nodejs.org.
[4] https://www.notion.so/.

encrypted way (RSA-2048). If the result is positive (i.e., infection with COVID-19), the responsible health department is notified automatically and receives the individual's contact details.

❸ **Register Test Result.** The system informs individuals via e-mail about their test results, attaching a password protected pdf to secure their personal data—which is also why the startup does not use push notifications in the user app itself. Note that this is an additional notification, since the user app retrieves the result from a server as soon as the individual decides to do so. If the result is negative, the individual's QR code is re-generated by adding the test result and time of testing. Otherwise, the QR code is not re-generated, but the individual receives instructions on how they should behave (e.g., social distancing, performing a more reliable test).

❹ **Verify Test Result.** Using the entry point app, event organizers (e.g., at cinemas, concerts, restaurants) can now verify the personal QR code of negative tested individuals to allow them access to more public life. To account for different policies, the validity of tests can be checked against a specified time frame (i.e., up to 48 h after the test). Scanning an individual's QR code returns a green arrow (i.e., valid certificate) or a red cross (i.e., no test result available, positive test result, or expired certificate). Also, the entry point app stores each individual's contact data for a specified period of time (e.g., 14 days) following the legal requirements of the authorities. This allows to automatically trace infection chains and inform users who had contact with another positive tested individual.

3. Data collection

Next, we describe the research objectives of our experience report, the design of our qualitative survey, its conduct, and our analysis process.

3.1. Research objectives

To understand how the startup developers were impacted by the properties of their environment (cf. Fig. 1), we defined three research objectives (ROs):

- RO_1 Investigate the impact of pandemic startup software engineering on developers' roles and responsibilities.
- RO_2 Elicit which of the three properties had what impact on the development process.
- RO_3 Collect developers' experiences regarding good practices and challenges of engineering the verification system during the COVID-19 pandemic.

With RO_1, we aimed to understand how the development process looked like, and how it changed due to the specific situation of the startup. For RO_2, we were concerned with which of the aforementioned properties caused the changes we identified. Finally, for RO_3, we elicited the developers' experiences of developing the verification system to understand practices that worked well, and those that did not.

3.2. Study design

In Fig. 3, we provide an overview of the study design we employed to address our research objectives. Namely, we first discussed the startup's situation, development process, and our research objectives among the authors of this article—involving two experts from the startup (i.e., the first two authors are the business manager and head of software development). Based on roughly eight hours of exploratory discussions between all authors, we iteratively designed a qualitative survey that we distributed among the developers of the startup (explained shortly)—with the two experts not participating in the survey to avoid biases. Then, we analyzed the results among all authors.

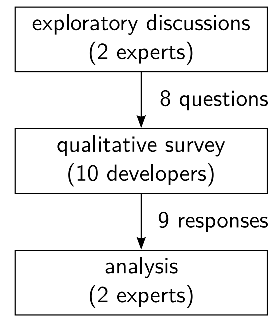

Fig. 3. Overview of our methodology.

General Survey Design. We constructed our survey based on the guidelines by Groves et al. (2011). First, we decided to employ an internet-based survey (using Google Forms[5]) to prevent transcription errors that may occur during an interview and save the participants' time; particularly since some have moved to other organizations. To still elicit detailed insights, we relied on open-ended questions and free-text answers to allow our participants to elaborate on the issues they considered important. All participants received the same questions, independent of their roles or previous experiences. We also expected that a short qualitative survey would have a higher acceptance rate than a more extensive interview, resulting in a higher response rate (Mathers et al., 1998; van Selm and Jankowski, 2006). Moreover, we conducted the survey anonymously (i.e., we did not collect contact data or used URL tracking), voluntarily, and hoped to encourage participants to provide honest answers by highlighting that the startup (i.e., the two experts) wanted to understand how to improve. By employing this survey design, we aimed to achieve a higher objectivity and quality in the responses.

We derived a first set of potential questions during the exploratory discussions of our research objectives. Based on these discussions, the two startup experts derived a number of concrete questions, formulated clearly and consistently according to the participants' educational level as well as the startup's terminologies (Glasow, 2005). The third author reviewed the questions and discussed them with the two experts, who revised the questions afterwards (e.g., reordering them to avoid biases). Finally, our survey comprised a general introduction section and eight questions that we divided into three thematic sections. We display all questions, their answers, and their identifiers (which we use throughout this article) in Table 1.

Section: Roles and Responsibilities. First, we asked for the participants' experience of developing software systems in years ($\mathbf{Q_1}$), their roles in developing the verification system ($\mathbf{Q_2}$), and their participation in specific development phases ($\mathbf{Q_3}$). The first two questions are based on established guidelines for asking about developers' experiences (Siegmund et al., 2014). For the last question, we built upon the six development phases of classical development processes, such as the waterfall model (Alshamrani and Bahattab, 2015). However, the two experts of the startup adapted them to the specifics of developing the verification system (e.g., the terminology), leading to six phases:

1. conception (e.g., economic and general planning);
2. requirements analysis (e.g., user surveying);
3. system design (e.g., architecture and mock-up design);
4. implementation (e.g., back-end and front-end);
5. testing and quality assurance (e.g., user testing); and
6. deployment as well as maintenance (e.g., support).

[5] https://www.google.com/forms/about/.

Table 1
Questions in our survey.

id	Question	Answer options
Section: roles and responsibilities RO_1		
$\mathbf{Q_1}$	How many years of experience in software development do you have?	*Multiple choice*: ❍ <3 ❍ 3 – <5 ❍ 5 – <10 ❍ 10 – 15 ❍ >15
$\mathbf{Q_2}$	What has been your role in the project?	*Multiple response*: ❑ back-end developer ❑ front-end developer ❑ project manager ❑ system architect ❑ tester ❑ UI/UX designer ❑ *<free text>*
$\mathbf{Q_3}$	From our point of view, the project involved six (possibly recurring) development phases. In which phase(s) have you been involved?	*Multiple response*: ❑ conception ❑ requirements analysis ❑ system design ❑ implementation ❑ testing and quality assurance ❑ deployment and maintenance
Section: development process RO_2		
$\mathbf{Q_4}$	Does the verification system's development process based on these six phases differ from your previous projects (e.g., missing phases)? If yes, what was different?	*Free text*
$\mathbf{Q_5}$	In your opinion, what were the biggest drivers for these differences?	*Four-level Likert scales*: ❍ no driver ❍ low driver ❍ medium driver ❍ major driver (one scale for each: *startup environment*; *pandemic situation*; *pandemic system development*)
Section: experiences RO_3		
$\mathbf{Q_6}$	What worked well in the development process?	*Free text*
$\mathbf{Q_7}$	What did not work well in the development process? What could be improved?	*Free text*
$\mathbf{Q_8}$	Additional comments (Is there anything that is important to you that has not been part of this survey?)	*Free text*

For these questions, we employed multiple choice and multiple response answers. We used multiple response answers for the participants' roles and participation in phases, since it is typical in startups that developers switch between those (e.g., due to a small team size). These questions were mainly related to RO_1.

Section: Development Process. Second, we included two questions that focused on the development process of the verification system. Essentially, we aimed to understand whether (and if so, which) the participants perceived any major differences in developing the verification system compared to other software projects they worked on ($\mathbf{Q_4}$). For this purpose, we referred to the previous phases to define a consistent level of granularity (e.g., focusing on whether and why some phase may have been insufficiently fulfilled instead of technical details, such as problems with a tool). To obtain detailed insights and allow participants to elaborate, we used a free-text answer. Based on our exploratory discussions, we expected that there were differences, which is why we wanted to explore which of the properties we defined had what impact ($\mathbf{Q_5}$). For this purpose, we defined a four-level Likert scale on which each participant could rank the three properties: 1. the pandemic situation, 2. the development of a product for managing the pandemic, and 3. the startup context. These questions were mainly concerned with RO_2.

Section: Experiences. Finally, we asked three questions regarding the participants' positive ($\mathbf{Q_6}$), negative ($\mathbf{Q_7}$), or other ($\mathbf{Q_8}$) experiences regarding the development process of the verification system. We intentionally used open-ended questions with free-text answers to elicit feedback that would not otherwise come up, and thus to gain additional insights (Glasow, 2005). Specifically, we aimed to elicit detailed qualitative feedback on the participants' experiences in pandemic startup software engineering. These questions were mainly concerned with RO_3.

3.3. Conduct

On July 19, 2021, we sent the survey to all developers who participated in the verification system's development. As developers, we defined everyone who has been directly involved in the development process at any time, for instance, front-end and back-end developers, system testers, or UI/UX designers. Moreover, we included permanent employees and freelancers, also considering those who were no longer employed by the startup. By also considering former employees, we aimed to increase the sample size and mitigate biases caused by the current employment of permanent employees. Overall, we identified ten developers who could participate in the survey, which is a typical team size in startups (Almeida et al., 2003; Giardino et al., 2015). Since we focused on gaining qualitative insights instead of a quantitative study, fewer but more detailed responses are typically sufficient and more valuable.

In the end, we received nine responses (90% response rate). One freelancer informed us that they could not participate, due to a lack of time and privacy concerns. Seven out of eight questions were answered in detail by all participants. Solely $\mathbf{Q_8}$ was answered only by three of the nine participants. However, this question is concerned with additional feedback only, which is why this does not pose a threat to our study's validity. During our analysis, we found that the answers were of high quality (e.g., regarding the level of detail, clarity, writing) and they did not seem to tend towards particularly positive or negative feedback by any participant. So, even though we have a small sample size, we argue that the qualitative insights are reliable, honest, and of high interest for practice and research.

3.4. Analysis

We analyzed and discussed all responses among all authors. Namely, the two experts used open-coding to label the free-text responses and open-card-sorting (Zimmermann, 2016) to identify and connect themes within the codes. The experts then provided additional context on the participants' answers, matched them to the overall development process, and consulted additional documentation to clarify details. For instance, one participant mentioned that "since April 2021, less meetings [...]" have been a problem, with the experts agreeing and providing the context that the startup joined its efforts with a contact-tracing app provider and had just finished its first real-world test run (cf. Section 2.2). Note that even though the questions in each section of our survey are aligned to one specific research objective, we always considered all answers to complement details and obtain a better understanding. After discussing the data among all authors, the two experts derived an initial set of key insights that we collaboratively refined, extended, and clarified throughout continuous discussions.

4. Results and discussion

In this section, we describe and discuss the results of our study, focusing on each research objective individually and integrating the experts' perspectives into the discussions.

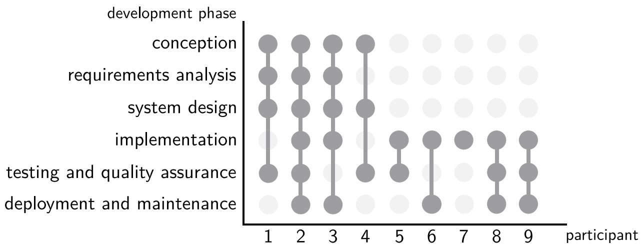

Fig. 4. Participation in development phases (Q_3).

Table 2
Participants' experiences and roles.

Question	Answer	# responses
$\mathbf{Q_1}$: experience	<3 years	2
	5 to 10 years	4
	>15 years	3
$\mathbf{Q_2}$: roles	Back-end developer	3
	Front-end developer	3
	Project manager	2
	System architect	3
	Tester	2
	UI and UX designer	2

4.1. RO$_1$: Roles and responsibilities

Results. In Table 2, we summarize our participants' responses regarding their software development experiences ($\mathbf{Q_1}$) and roles in the startup ($\mathbf{Q_2}$). We can see that four participants have between 5 and 10 years of experience, three have more than 15 years of experience, and only two are relatively new with fewer than 3 years of experience. There is an almost even distribution of our participants across the six roles we identified in our discussions (i.e., two to three developers for each role). Moreover, our data shows that most participants had more than one role.

In Fig. 4, we display in which development phases our participants have been involved ($\mathbf{Q_3}$). The majority (7) contributed to the implementation of some part of the verification system. Moreover, six of our participants have been involved in testing and quality assuring (e.g., user tests, UI/UX adjustments), as well as five in the deployment and maintenance. Other typical phases involved fewer participants, namely four for each the conception and system design, as well as three for the requirements analysis. Five participants who worked in the implementation phase have also been involved in the deployment and maintenance of the verification system. Interestingly, one participant contributed to all phases. Also, while almost all participants contributed to multiple phases, we can clearly observe that five participants have never been involved in the scoping of the verification system (i.e., conception, requirements analysis, system design).

Discussion. Developers' experience is a major indicator for the success of young organizations (Lasch et al., 2007; Muñoz-Bullon et al., 2015), and thus invaluable for a startup to succeed (Kajko-Mattsson and Nikitina, 2008; Grilli, 2011; Salamzadeh and Kawamorita Kesim, 2015). In the subject startup, we found a mix of less experienced and more experienced developers. According to the two experts, this constellation resulted from the specific composition of the startup team, which is based on an inexperienced initial team without any established workflows. Later, the startup hired more experienced developers (e.g., freelancers) to improve its software development. Consequently, the lower experience of some developers (i.e., developers with fewer than 3 years) was compensated by the high experience (i.e., more than 15 years) and in-depth knowledge of other developers.

The more inexperienced members of the startup caused a few issues (e.g., mistakes in maintaining the server infrastructure). This situation was intensified by the time pressure caused by the pandemic and the many corresponding feature changes (e.g., due to unexpected changes in laws). However, while hiring experienced developers improved the system's quality, it also led to interpersonal problems (e.g., between the operations teams and the development team). For example, the startup founders have a greater emotional attachment to the verification system and the underlying idea than those who were just hired (potentially only to support the development for a short period). Since the freelancers solved system-related issues more objectively than the founders, personal problems arose, for example, regarding the perceived and actual usefulness of features.

The lockdown in Germany forbade in-person group meetings, and thus massively restricted in-person communication. While the developers resolved this issue by communicating through video calls, these could not fully replace in-person communication. The founders therefore set up at least two in-person meetings a year under strict health procedures to improve the teamwork. Overall, the founders learned from their mistakes and the advice of the more experienced developers, as evidenced by the system's success and the team's continuous growth. Moreover, the lower experience of some developers was not only compensated by the higher experience of others, but also by additional factors, such as a proactive customer approach (Groenewegen and de Langen, 2012) and the strong need for the verification system.

Insight$_1$: Experience and Team Composition

The inexperience of individuals (e.g., startup founders) in the team was compensated by the higher experience of other individuals (e.g., freelancers). Still, interpersonal problems did occur, mainly due to health restrictions, personal experiences, or emotions.

Due to the small team size, the startup developers had to take on several roles. Thus, their contributions to the development were significantly higher than in traditional organizations with clear roles and tasks (Almeida et al., 2003; Giardino et al., 2015). Unfortunately, this situation also led to overlapping responsibilities, causing various problems (e.g., interpersonal problems). In addition, our survey results (cf. Table 2) show a slightly stronger focus on the actual programming-related roles (e.g., back-end developers). This confirms the perceptions of our experts that the startup focused on fast implementation and deployment of the verification system to take full advantage of the market niche caused by the pandemic.

Since the startup lacked experts for several roles (e.g., project management, design, user testing), these were often handled by members of the operations team. However, this situation caused several problems. For instance, early on the project management tool notion.so was used, which provided several advantages for the initially small scale of the verification system (e.g., programmable Kanban boards). As the team grew (i.e., more developers were involved), notion.so became complicated and confusing, which is why Jira was introduced. While this may represent a typical change for growing startups, our experts argue that it was mainly due to the inexperience of the initial team and an underestimation of the project's complexity (e.g., fluctuating requirements and laws). Still, the inexperience also had advantages, for example, it led to more light-weight workflows and proactive customer approaches—often resulting in successfully completed tasks and positive user feedback.

Insight$_2$: Experience and Roles

Due to the small size of the startup team, some roles were filled by less experienced individuals. This led to interpersonal problems and a loss of quality compared to traditional organizations, but also to new ways of working that improved the personal development of the involved developers.

Our results on the development process (cf. Fig. 4) show that the developers had to participate in several phases within the scope of their roles. Again, this situation caused the risk of overlapping areas of responsibility, which can lead to several problems, such as developers performing tasks assigned to someone else (Giardino et al., 2015). Furthermore, we can see in Fig. 4 that the startup focused mainly on writing code and deploying the system as fast as possible. This is emphasized by the fact that five of the seven participants who implemented code have also worked on deployment, and those who deployed and maintained the system have worked on the implementation, too. Interestingly, in our experts' experience, this particular overlapping of responsibilities helped to make decisions more quickly.

The high pressure regarding potential competitors and developing a product for managing the pandemic resulted in a lack of time. As a consequence, the requirements analysis was minimized (Alves et al., 2020), leading to several issues (e.g., choosing the right development technology, more costly refinements of requirements in later phases, meeting deadlines). Interestingly, data security and privacy requirements were actually the only ones that were analyzed intensively in advance. This was mainly due to the fact that the data being processed, transferred, or stored is particularly sensitive (e.g., health data). Therefore, it has to be protected according to data privacy regulations, for example, the European General Data Protection Regulation (Albrecht, 2016). Furthermore, data security and privacy concerns were identified early on as points of criticism and weakness of existing contact-tracing apps (Wen et al., 2020; Rowe, 2020; Biddle et al., 2022) and verification systems (Karamitsos and Papadaki, 2021; Papadaki et al., 2021). While logical, we find it interesting that the specific emergency situation of a pandemic involving health data essentially reversed typical priorities of software developers. Instead of focusing on delivering new features and caring about security later, security became the focal point of the development.

The verification system uses only the most essential data in a decentralized-like architecture to address such requirements and achieve a competitive advantage. In fact, this concept was perceived very positively by the media and authorities, in contrast to the designs of competing verification systems. So, the concept was crucial, particularly in the context of the pandemic, to achieve a competitive advantage. Nevertheless, it is not clear if a more comprehensive requirements-analysis process would have allowed meeting politics-imposed deadlines enforced due to the pandemic. These deadlines could not be postponed, since they were defined by laws and public or media expectations. In this context, the experts argue that the startup did benefit from one major strength of startups: adopting any development style or workflow to achieve their goal as fast as possible (Giardino et al., 2015). For this reason, we think that startups may be more successful in emergency situations compared to other organizations, due to their flexibility regarding development processes and technologies. However, the overall situation caused high psychological pressure for all developers. Specifically, they were responsible for developing a system that works and offers its users the possibility of participating in public life as soon as possible. Programming mistakes or bugs (e.g., incorrect transmission of test results) would have been fatal for the startup and the developers' reputation.

Insight$_3$: Development Phases

Due to the small size of the team, individuals participated in several development phases. In the particularly stressful context (e.g., social responsibility of a pandemic product, health data), the focus was on fast deployment and security. Particularly, requirements engineering was typically skipped to meet government-enforced deadlines, which, in turn, resulted in requirements not being fulfilled.

Summary and Recommendations. The developers of the startup needed to take on multiple roles and phases of the development process, due to the small team size and high pandemic-caused pressure (e.g., time, reputation, social responsibility) of developing a pandemic (i.e., emergency) product. Their focus was mainly on fast delivery, and thus development, testing, deployment, as well as security. While this encourages the personal development of inexperienced team members (e.g., learning from mistakes), the potential for errors should be mitigated by involving more experienced developers and building on lessons learned. So, to better understand and manage software engineering in emergencies, we argue that researchers should investigate and organizations can benefit from the flexibility of agile/startup development processes, the integration of experienced software developers in teams with changing roles, and the clear definition of the most critical requirements. While communication between team members is key for any organization, it is even more important in an emergency situation in which in-person communication is difficult to achieve. To investigate our insights in more detail, we now study how the three properties we identified impacted our participants before reporting their personal experiences.

4.2. RO$_2$: Properties' impact

Results. Regarding **Q$_4$**, we did not identify similarities in the participants' responses: everyone indicated other differences in the development phases. However, the answers mainly refer to the project management of all phases. For instance, the participants wrote that the phases could hardly be separated from each other, because everyone wanted to be involved in every phase (cf. **Insight$_3$**). They also stated that a structured agile process (Hazzan and Dubinsky, 2008) with iterative phases (e.g., Scrum) and comprehensive user tests were missing. Interestingly, only three participants wrote that there were hardly any differences to previous projects in startups. This indicates that the other participants considered this startup and its context to be somewhat different.

In Fig. 5, we display our participants' assessment of which of the three properties (cf. Fig. 1) caused differences in the development process **(Q$_5$)**. We can see that five participants chose the startup context as a major driver. Two participants each considered this property as a medium and low driver. None of the participants stated that the startup context had no impact. Both, the pandemic situation and pandemic product development, were mentioned three times as the strongest and two times each as a medium driver. The pandemic situation was assigned twice and pandemic product development once as a low driver. Particularly, the latter was named as no driver in three cases. So, the startup context seems to be the most important property when it came to changes in the development process, but the pandemic situation and pandemic product apparently still impacted most participants somewhat.

Discussion. The startups' development process primarily lacked a planned project management, for example, regarding the distribution of roles and phases or comprehensive user testing. Our experts reason that this is partly due to inexperience and partly due to the strong focus on fast implementation and deployment (i.e., the impact of pandemic product development). This focus, in turn, originated from the aforementioned pandemic-related time pressure and consequent

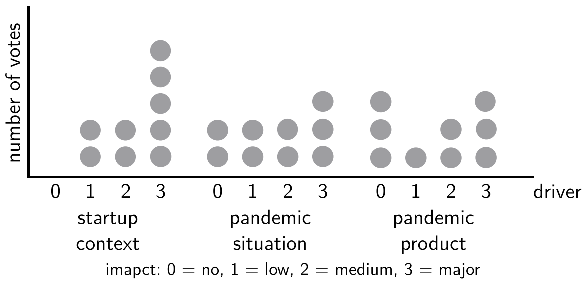

Fig. 5. Assessments of the three properties' impact (Q_5).

Table 3
Positive and negative feedback regarding the development process of the startup.

Question	Answer	# responses
$\mathbf{Q_7}$: *positive*	Communication	3
	Organization	3
	Personal development	3
$\mathbf{Q_8}$: *negative*	Communication	5
	Analyses, planning, and documentation	3
	Workload and pressure	3
	Compliance	2
	Employee fluctuation	1

mental stress. While we assume that more thoughtful project management would have taken more time in the beginning, it would likely have saved even more time later on. Due to the inexperience of the initial team, this lesson learned emerged only after months of development, also because the emergency situation asked for fast development instead of long periods of planning and preparing. We argue that this situation showcases a conflict between establishing effective processes and the real-world in an emergency situation. Since three participants mentioned not noticing any differences to previous development projects, this may mean that the specifics of the subject startup are not unusual among startups. Therefore, our findings regarding development phases should be transferable to other startups, with the added insights regarding the influences of pandemic software engineering.

Insight$_4$: Perceived Differences

Due to the pressure of developing a pandemic product, the startup lacked systematic project management, which could have prevented some delays and problems later on. This situation is arguably similar to other startups (as indicated by three participants). However, the pandemic situation and pandemic product are special circumstances asking for fast delivery, which challenged the startup in establishing project management.

According to our participants, the startup context had the most impact on the development process. Specifically, most characteristics of the startup context are related to inexperience and the small team size. Both factors together usually lead to mistakes made by one or more individuals (e.g., skipping requirements analysis). However, these mistakes also result in new ways of working that would not arise in this form in traditional organizations with a well-defined development process (e.g., rigid or agile). Particularly, development phases are skipped to achieve goals before the deadline, even though the long-term impact can be severe. Namely, such practices lead to a lower half-life and maintainability of the system if it is not improved later on by more experienced developers (e.g., freelancers).

The inexperience of the startup team was especially noticeable in the context of their evolutionary-like deployment process (Giardino et al., 2015). For instance, the first version of the test station app was released in the context of a practical test run. So, while time could be saved, this test run had to compensate for the too short internal test phase. Although there was no user test before, the test stations used the system regularly and tested hundreds of individuals a day. This test strategy was not planned or even expected, but could not be avoided due to the rigid time constraints. This was a high risk, since the system was not perceived as ready to use on a large scale. However, due to a lot of first-level and freelancer support, the test run was completed successfully.

Both, the pandemic situation and pandemic product development, are perceived to have a similar (medium) impact on the development process, but also the whole startup. However, as we already exemplified above, many differences perceived by the participants and experts seem to actually stem from these two properties, but are noticed somewhat less. In this context, major impact factors are the pandemic situation itself (e.g., COVID-19), customer (i.e., politicians) decisions (e.g., laws or deadlines), as well as the working situation (e.g., well-being due to home office). Moreover, the experts note that developing a pandemic product includes media pressure (e.g., social media), direct user feedback (e.g., through social media), and social responsibility (e.g., data privacy, serious consequences of mistakes). Both drivers are arguably similar for other emergency situations (e.g., developing in a situation of and for handling a natural disaster), whereas the startup context as the perceived main driver would not change in such a situation.

Insight$_5$: Properties' Impact

The startup context was perceived as the major property causing differences in the development process (e.g., project management). However, our experts argue that some causes are more closely connected to the pandemic situation and pandemic product development (e.g., time and personal pressure), which are transferable to other emergency situations.

Summary and Recommendations. Pandemic startup software engineering can lead to differences in development processes, which are closely related to other typical startups. However, our insights indicate that differences and problems are amplified by the pandemic situation and pandemic product development (e.g., performance pressure). Interestingly, most developers seem to associate such issues only with the startup context. We argue that further research is needed to better understand how emergencies impact software development processes, and how organizations can employ more lightweight and flexible project management to rapidly developed new systems. To understand potential problems and their connections to the properties in more detail, we next analyze the participants' personal experiences.

4.3. RO$_3$: Participants' experiences

Results. In Table 3, we summarize the participants' positive ($\mathbf{Q_7}$) and negative experiences ($\mathbf{Q_8}$) regarding the overall development process. Not surprisingly, the detailed responses are not identical, but they are concerned with similar issues. Thus, we used open-coding and open-card-sorting methods to derive suitable topics (i.e., not based on an existing categorization) that we could use as higher-level classifications to achieve a better understanding of the responses. In this context, we organized all positive responses into three categories: organization, communication, and personal development. Regarding communication, three participants described the collaboration between the development team and the operations team as well as the communication among the developers as excellent and efficient (e.g., using video conferences). For the organization, the useful restructuring of the team during the implementation phase (i.e., hiring freelancers), the strong focus on the use case of the verification system (i.e., detailed concept), and the structured execution of meetings (e.g., based on agendas) were pointed out positively. Moreover, the direct customer communication, interaction, and feedback (i.e., politicians, test stations, entry points, end users) were reported as positive experiences during the project. Three participants stated that there was a great learning process that helped them develop personally.

Regarding the negative feedback, we organized the identified themes into five categories. Interestingly, while communication was also mentioned as a positive experience, five participants considered it insufficient. Moreover, three participants criticized the lack of requirements analysis, planning, and documentation; as well as the workload and pressure. Two participants stated that there was poor compliance with processes, responsibilities, and priorities. One participant mentioned that the fluctuation among the employed developers was too high.

Three participants shared further feedback ($\mathbf{Q_8}$) in addition to the positive and negative statements. One participant stated that experience, technical understanding, and financial resources led to interpersonal problems. The second participant described that requirements often changed due to the pandemic situation, which made their work highly challenging. Finally, one participant criticized insufficient trust and appreciation during stressful and excessive workloads.

Discussion. The communication perceived as positive was particularly noticeable in terms of customer support. Namely, the operations team handled first-level support and worked closely with the developers to solve problems as soon as possible. In this context, the most successful concepts and lessons learned (e.g., regarding the organization) arose from openly discussed mistakes within the team. Specifically, potential problems (e.g., interpersonal or code-related) could be identified early on, and were usually resolved quickly. Note that, while in the beginning of the startup all team members (i.e., founders and first developers) had contact with the customers, this shifted exclusively to the operations team (including customer support). The main contact person took on a product owner-like role, asking for and defining the corresponding requirements. Depending on (unexpected changes in) legal requirements due to the pandemic, arrangements had to be adjusted efficiently, which was a particular burden for the communication with customers, also causing additional costs for the startup and the customers. Overall, the freedom of a startup project allowed for more personal development, which improved the developers' motivation. Specifically, due to the small team size, every developer could get involved in any phase (cf. **Insight$_3$**), if basic skills and the motivation were available. Note that this situation also brought disadvantages, especially with regard to the quality of the outcomes, which would have been significantly higher with more experienced developers.

Insight$_6$: Good Practices

Even though the startup worked in a pandemic, its greatest advantages of freedom and flexibility still remained. This allowed the developers to directly interact with their customers and gain experiences in different roles and development phases. While a stricter organization could have benefited the actual development, the situation improved motivation and engagement.

It is interesting that the communication within the team was perceived problematic, while the communication with customers seems to have been far better. The criticism regarding internal communication has several reasons, primarily due to the founders' inexperience. Precisely, tasks were not communicated clearly enough or personal as well as development-related problems were not always understood. Accordingly, frustration arose in some cases, which led to interpersonal problems and unproductivity. This aspect was intensified by the pandemic situation, which affects developers' productivity and overall well-being (Ralph et al., 2020; Silveira et al., 2021; Russo et al., 2021b). In fact, the entire team could never meet in person during the initial development, but only via videoconferencing. This resulted in even more conflicts, which are also represented in the criticism regarding communication and compliance. As mentioned, the founders aimed to mitigate this issue by implementing at least a couple of in-person meetings a year under strict health regulations.. Moreover, due to insufficient analyses, planning, and documentation, maintaining the system was also difficult for the developers. Particularly, the high workload and pressure resulted in minimal documentation to save time. However, as developers (e.g., freelancers) changed, additional onboarding was necessary in each case (Krüger and Hebig, 2020), which took a lot of time. In the end, such pitfalls increased the overall costs and caused various mistakes, for instance, due to insufficient documentation (e.g., regarding the app navigation).

Insight$_7$: Pitfalls

Most pitfalls we identified are related to the startup context with partly inexperienced team members, but were amplified by the pandemic situation and pandemic product development. This insight confirms our previous findings that the pandemic impacted developers differently than they perceived.

The additional feedback we received was mainly related to interpersonal problems, and thus confirms our previous insights. Some participants could not work well under the conditions of the startup (i.e., inexperience as well as time and performance pressure), especially if there was not enough emotional attachment to the verification system. This led to several conflicts that could not always be resolved through open communication. Our experts experienced that this was intensified by the pandemic situation and the psychological stress (e.g., losing one's own reputation). Overall, we argue that the development of the verification system required a high level of risk-taking and heavily impacted developers' well-being, due to its unique properties. In particular, while time and personal pressure may also be present in projects without any emergency context, the pandemic situation and the development of the pandemic product were additional burdens for every person in the startup. So, it is an important research direction to provide better support for developers in emergency situations to reduce their stress and improve their well-being.

Insight$_8$: Well-Being

The startup's unique situation increased the typical startup pressure and stress on its developers, leading to dissatisfaction, unproductivity, and poor well-being.

Summary and Recommendations. The most significant advantages of a startup are the learning opportunities and consequent personal development. Still, there are challenges regarding interpersonal conflicts, workload and well-being, clear communication and documentation, as well as compliance—mostly caused by inexperience and pressure. Our participants' responses confirmed our previous insights, particularly that the pandemic considerably impacted developers. Interestingly, our participants perceived most issues to simply relate to the startup context, which may indicate that they underestimate the actual causes (i.e., due to comparing to previous non-pandemic projects). In future emergencies, we would recommend organizations to establish direct communication not only to the customer, but also within the development team; and to pay particular attention to the stress and well-being of their developers.

5. Threats to validity

In the following, we discuss threats that may impair the validity of our experience report.

Construct Validity. The construct validity of our study may be affected by the subjective perception of the participants. Specifically, we did not clearly define what development experience means in **Q1**. So, we do not know whether our participants considered their programming skills or anything else as experience. However, we assume that our participants reflected on experience based on the time they spent in previous projects, since we also defined the answer options in years. We are also aware that open-ended questions (**Q4**, **Q6**, **Q7**, and **Q8**) require precise answers (Glasow, 2005). However, some participants may not answer in detail to save time. We mitigated such threats by using a small number of questions, through personal contacts, and by assessing the (high) quality of the responses.

Internal Validity. A threat regarding the internal validity arises for our analysis of the participants' free-text answers. For instance, some participants indicated a positive experience that others considered as negative, due to subjective perceptions and needs (e.g., regarding communication). We aimed to avoid misinterpretations by employing open-coding and open-card-sorting methods, and by relying on the knowledge and documentation of the two experts, who added further details, contexts, and reasonings. Since participation was anonymous, it is possible that a participant responded to the survey more than once. While we cannot check for this since we uphold the anonymity, multiple responses by one participant are unlikely due to the personal contacts, time required, and the fact that exactly one former employee notified us not to answer (which aligns to the number of responses and the response rate of 90%).

External Validity. We could only elicit nine responses, which may affect the external validity of our findings. However, we point out that startups are characterized by their small team size (Almeida et al., 2003; Salamzadeh and Kawamorita Kesim, 2015; Giardino et al., 2015), we received responses from 90% of the possible participants, and a smaller number of participants is typical for qualitative research. Furthermore, a survey with more developers would not have been possible, due to the unique properties of this case. Namely, we do not have access and are not aware of another case like the one we describe in this article. For this reason, we could not further mitigate this (inherent) threat to the external validity—but some details of our findings are identical to those of related studies (cf. Section 2), which increases our confidence in their validity.

Conclusion Validity. The conclusion validity may be threatened by our interpretation of the data. To limit this threat, we involved experts from the startup, consulted documentation, and discussed the insights extensively. Still, other researchers may derive other insights, which is why we publish the anonymous responses to our survey for replications.[1]

6. Conclusion

In this article, we presented an experience report on the development of a digital COVID-19 certificate verification system by a startup during the pandemic. For this purpose, we conducted a qualitative survey with nine of the startup's developers and analyzed the findings with two experts from the startup. We identified insights on startup and pandemic software engineering that are in line with related work (e.g., no clear responsibilities, negative impact on well-being), but they also provide a more detailed understanding of the interactions of these properties:

- Developing a pandemic product within the pandemic impacts software developers, particularly by increasing pressure and stress due to the public interest in the system and its social importance.
- The pandemic situation challenges the coordination and communication between developers, which can easily lead to more mistakes, unclear responsibilities, and frustration.
- Developers associate most of the problems with the startup context, while the causes seem to be amplified by the pandemic and developing a pandemic product.

We argue that these insights can be transferred to other emergency situations, and thus lead the design and development of software that can help mitigate these situations. Still, we strongly recommend to conduct further research based on our results, especially regarding the impact of emergency situations on startups and other organizations. For instance, analyzing and comparing repositories of startups with small and medium-sized enterprises that developed emergency systems could lead to valuable insights. Based on the research directions we sketched throughout this article, we emphasize the need to develop new methods and tools to support developers' well-being, processes, and coordination as preparation for future emergencies.

CRediT authorship contribution statement

Richard May: Conceptualization, Methodology, Validation, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Niklas Baron:** Conceptualization, Methodology, Validation, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Jacob Krüger:** Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing, Visualization, Supervision. **Thomas Leich:** Conceptualization, Validation, Writing – original draft, Writing – review & editing, Supervision, Project Administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The results of our survey are available on Zenodo: https://doi.org/10.5281/zenodo.10968281.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jss.2024.112106.

References

Abuhammad, S., Khabour, O.F., Alzoubi, K.H., 2020. COVID-19 contact-tracing technology: Acceptability and ethical issues of use. Patient Prefer. Adherence 14, 1639–1647. http://dx.doi.org/10.2147/PPA.S276183.

Ahmed, N., Michelin, R.A., Xue, W., Ruj, S., Malaney, R., Kanhere, S.S., Seneviratne, A., Hu, W., Janicke, H., Jha, S.K., 2020. A survey of COVID-19 contact tracing apps. IEEE Access 8, 134577–134601. http://dx.doi.org/10.1109/ACCESS.2020.3010226.

Albrecht, J.P., 2016. How the GDPR will change the world. Eur. Data Prot. Law Rev. 2, 287–289.

Almeida, P., Dokko, G., Rosenkopf, L., 2003. Startup size and the mechanisms of external learning: Increasing opportunity and decreasing ability? Res. Policy 32 (2), 301–315. http://dx.doi.org/10.1016/S0048-7333(02)00101-4.

Alshamrani, A., Bahattab, A., 2015. A comparison between three SDLC models waterfall model, spiral model, and incremental/iterative model. Int. J. Comput. Sci. Issues 12 (1), 106–111.

Alves, C., Cunha, J., Araújo, J., 2020. On the pragmatics of requirements engineering practices in a startup ecosystem. In: International Requirements Engineering Conference. RE, IEEE, pp. 311–321. http://dx.doi.org/10.1109/RE48521.2020.00041.

Bano, M., Zowghi, D., Arora, C., 2020. Requirements, politics, or individualism: What drives the success of COVID-19 contact-tracing apps? IEEE Softw. 38 (1), 7–12. http://dx.doi.org/10.1109/MS.2020.3029311.

Biddle, N., Edwards, B., Gray, M., Hiscox, M., McEachern, S., Sollis, K., 2022. Data trust and data privacy in the COVID-19 period. Data Policy 4, http://dx.doi.org/10.1017/dap.2021.36.

Chipendo, T., Marembo, T., Munemo, C., Chituri, H., Shumba, E., Bangure, D., Maruta, T., 2022. Implementation of the COVID-19 laboratory testing certification program (CoLTeP). Pan Afr. Med. J. 42.

Ciotti, M., Ciccozzi, M., Terrinoni, A., Jiang, W.-C., Wang, C.-B., Bernardini, S., 2020. The COVID-19 pandemic. Crit. Rev. Clin. Lab. Sci. 57 (6), 365–388. http://dx.doi.org/10.1080/10408363.2020.1783198.

Eisenstadt, M., Ramachandran, M., Chowdhury, N., Third, A., Domingue, J., 2020. COVID-19 antibody test/vaccination certification: There's an app for that. Open J. Eng. Med. Biol. 1, 148–155. http://dx.doi.org/10.1109/OJEMB.2020.2999214.

Enaya, M.F., Klingbeil, T., Krüger, J., Broneske, D., Feinbube, F., Saake, G., 2024. A case study on the development of the German Corona-Warn-App. jss 213, 1–13. http://dx.doi.org/10.1016/j.jss.2024.112020.

Erikson, S., 2021. COVID-apps: Misdirecting public health attention in a pandemic. Glob. Policy 12, 97–100. http://dx.doi.org/10.1111/1758-5899.12888.

Garousi, V., Cutting, D., 2021. What do users think of the UK's three COVID-19 contact-tracing apps? A comparative analysis. Health Care Inform. 1–7. http://dx.doi.org/10.1136/bmjhci-2021-100320.

Garousi, V., Cutting, D., Felderer, M., 2022. Mining user reviews of COVID contact-tracing apps: An exploratory analysis of nine European apps. J. Syst. Softw. 184, 1–75. http://dx.doi.org/10.1016/j.jss.2021.111136.

Giardino, C., Paternoster, N., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, P., 2015. Software development in startup companies: The greenfield startup model. IEEE Trans. Softw. Eng. 42 (6), 585–604. http://dx.doi.org/10.1109/TSE.2015.2509970.

Glasow, P.A., 2005. Fundamentals of Survey Research Methodology. Technical Report MP 05W0000077, Mitre Corporation.

Grilli, L., 2011. When the going gets tough, do the tough get going? The pre-entry work experience of founders and high-tech start-up survival during an industry crisis. Int. Small Bus. J. 29 (6), 626–647. http://dx.doi.org/10.1177/0266242610372845.

Groenewegen, G., de Langen, F., 2012. Critical success factors of the survival of start-ups with a radical innovation. J. Appl. Econ. Bus. Res. 2 (3), 155–171.

Groves, R.M., Fowler, Jr., F.J., Couper, M.P., Lepkowski, E., Tourangeau, R., 2011. Survey Methodology. Wiley.

Gupta, R., Pandey, G., Chaudhary, P., Pal, S.K., 2021. Technological and analytical review of contact tracing apps for COVID-19 management. J. Locat. Based Serv. 1–40. http://dx.doi.org/10.1080/17489725.2021.1899319.

Hazzan, O., Dubinsky, Y., 2008. Agile Software Engineering. Springer.

Kajko-Mattsson, M., Nikitina, N., 2008. From knowing nothing to knowing a little: Experiences gained from process improvement in a start-up company. In: International Conference on Computer Science and Software Engineering. CSSE, IEEE, pp. 617–621. http://dx.doi.org/10.1109/CSSE.2008.1370.

Karamitsos, I., Papadaki, M., 2021. Blockchain digital test certificates for COVID-19. In: Modern Management Based on Big Data II and Machine Learning and Intelligent Systems III. IOS Press, pp. 133–143.

Karopoulos, G., Hernandez-Ramos, J.L., Kouliaridis, V., Kambourakis, G., 2021. A survey on digital certificates approaches for the Covid-19 pandemic. IEEE Access 9, 138003–138025. http://dx.doi.org/10.1109/ACCESS.2021.3117781.

Klotins, E., Unterkalmsteiner, M., Gorschek, T., 2018. Software engineering in start-up companies: An analysis of 88 experience reports. Empir. Softw. Eng. 24 (1), 68–102. http://dx.doi.org/10.1007/s10664-018-9620-y.

Krüger, J., Hebig, R., 2020. What developers (care to) recall: An interview survey on smaller systems. In: International Conference on Software Maintenance and Evolution. ICSME, IEEE, pp. 46–57. http://dx.doi.org/10.1109/ICSME46990.2020.00015.

Lasch, F., Le Roy, F., Yami, S., 2007. Critical growth factors of ICT start-ups. Manag. Decis. 45, 62–75. http://dx.doi.org/10.1108/00251740710718962.

Liang, F., 2020. COVID-19 and health code: How digital platforms tackle the pandemic in China. Soc. Media+ Soc. 6 (3), 1–4.

Manabe, Y.C., Sharfstein, J.S., Armstrong, K., 2020. The need for more and better testing for COVID-19. JAMA 324 (21), 2153–2154. http://dx.doi.org/10.1001/jama.2020.21694.

Mathers, N.J., Fox, N.J., Hunn, A., 1998. Surveys and Questionnaires. NHS Executive Trent.

Montanari Vergallo, G., Zaami, S., Negro, F., Brunetti, P., Del Rio, A., Marinelli, E., 2021. Does the EU COVID digital certificate strike a reasonable balance between mobility needs and public health? Medicina 57 (10), 1077. http://dx.doi.org/10.3390/medicina57101077.

Morley, J., Cowls, J., Taddeo, M., Floridi, L., 2020. Ethical guidelines for COVID-19 tracing apps. Nature 582 (7810), 29–31. http://dx.doi.org/10.1038/d41586-020-01578-0.

Muñoz-Bullon, F., Sanchez-Bueno, M.J., Vos-Saz, A., 2015. Startup team contributions and new firm creation: The role of founding team experience. Entrep. Reg. Dev. 27 (2), 80–105. http://dx.doi.org/10.1080/08985626.2014.999719.

Munzert, S., Selb, P., Gohdes, A., Stoetzer, L.F., Lowe, W., 2021. Tracking and promoting the usage of a COVID-19 contact tracing app. Nat. Hum. Behav. 5 (2), 247–255. http://dx.doi.org/10.1038/s41562-020-01044-x.

Papadaki, M., Karamitsos, I., Themistocleous, M., 2021. COVID-19 digital test certificates and blockchain. J. Enterp. Inf. Manag..

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, P., 2014. Software development in startup companies: A systematic mapping study. Inf. Softw. Technol. 56 (10), 1200–1218. http://dx.doi.org/10.1016/j.infsof.2014.04.014.

Piguillem, F., Shi, L., 2020. Optimal COVID-19 Quarantine and Testing Policies. Centre for Economic Policy Research.

Ralph, P., Baltes, S., Adisaputri, G., Torkar, R., Kovalenko, V., Kalinowski, M., Novielli, N., Yoo, S., Devroey, X., Tan, X., Zhou, M., Turhan, B., Hoda, R., Hata, H., Robles, G., Milani Fard, A., Alkadhi, R., 2020. Pandemic programming: How COVID-19 affects software developers and how their organizations can help. Empir. Softw. Eng. 25, 1–35. http://dx.doi.org/10.1007/s10664-020-09875-y.

Reelfs, J.H., Hohlfeld, O., Poese, I., 2020. Corona-warn-app: Tracing the start of the official COVID-19 exposure notification app for Germany. Ratio 10 (5), 10–12. http://dx.doi.org/10.1145/3405837.3411378.

Rowe, F., 2020. Contact tracing apps and values dilemmas: A privacy paradox in a neo-liberal world. Int. J. Inf. Manag. 55, 102178–102191. http://dx.doi.org/10.1016/j.ijinfomgt.2020.102178.

Russo, D., Hanel, P., Altnickel, S., van Berkel, N., 2021a. The daily life of software engineers during the Covid-19 pandemic. In: International Conference on Software Engineering. ICSE, IEEE/ACM, pp. 364–373. http://dx.doi.org/10.1109/ICSE-SEIP52600.2021.00048.

Russo, D., Hanel, P., Altnickel, S., van Berkel, N., 2021b. Predictors of well-being and productivity among software professionals during the COVID-19 pandemic – A longitudinal study. Empir. Softw. Eng. 26 (4), 1–63. http://dx.doi.org/10.1007/s10664-021-09945-9.

Salamzadeh, A., Kawamorita Kesim, H., 2015. Startup companies: Life cycle and challenges. In: International Conference on Employment, Education and Entrepreneurship. EEE, SSRN, http://dx.doi.org/10.2139/ssrn.2628861.

Seto, E., Challa, P., Ware, P., 2021. Adoption of COVID-19 contact tracing apps: A balance between privacy and effectiveness. J. Med. Internet Res. 23 (3), http://dx.doi.org/10.2196/25726.

Siegmund, J., Kästner, C., Liebig, J., Apel, S., Hanenberg, S., 2014. Measuring and modeling programming experience. Empir. Softw. Eng. 19 (5), 1299–1334. http://dx.doi.org/10.1007/s10664-013-9286-4.

Silveira, P., Mannan, U.A., Almeida, E.S., Nagappan, N., Lo, D., Kochhar, P.S., Gao, C., Ahmed, I., 2021. A deep dive into the impact of COVID-19 on software development. IEEE Trans. Softw. Eng. http://dx.doi.org/10.1109/TSE.2021.3088759.

Sun, R., Wang, W., Xue, M., Tyson, G., Camtepe, S., Ranasinghe, D.C., 2021. An empirical assessment of global COVID-19 contact tracing applications. In: International Conference on Software Engineering. ICSE, IEEE/ACM, pp. 1085–1097. http://dx.doi.org/10.1109/ICSE43902.2021.00101.

Sutcliffe, A., Sawyer, P., Liu, W., Bencomo, N., 2021. Investigating the potential impact of values on requirements and software engineering. In: International Conference on Software Engineering. ICSE, IEEE/ACM, pp. 39–47. http://dx.doi.org/10.1109/ICSE-SEIS52602.2021.00013.

van Selm, M., Jankowski, N.W., 2006. Conducting online surveys. Qual. Quant. 40 (3), 435–456. http://dx.doi.org/10.1007/s11135-005-8081-8.

Velavan, T., Meyer, C., 2020. The COVID-19 epidemic. Trop. Med. Int. Health 25 (3), 278. http://dx.doi.org/10.1111/tmi.13383.

Wahsheh, H., Al-Zahrani, M.S., 2021. Secure and usable QR codes for healthcare systems: The case of COVID-19 pandemic. In: International Conference on Information and Communication Systems. ICICS, IEEE, pp. 324–329. http://dx.doi.org/10.1109/ICICS52457.2021.9464565.

Wang, T., Jia, F., 2021. The impact of health QR code system on older people in China during the COVID-19 outbreak. Age Ageing 50 (1), 55–56. http://dx.doi.org/10.1093/ageing/afaa222.

Ward, J., Gauna, F., Gagneux-Brunon, A., Botelho-Nevers, E., Cracowski, J.-L., Khouri, C., Launay, O., Verger, P., Peretti-Watel, P., 2022. The French health pass holds lessons for mandatory COVID-19 vaccination. Nat. Med. 1–3. http://dx.doi.org/10.1038/s41591-021-01661-7.

Wen, H., Zhao, Q., Lin, Z., Xuan, D., Shroff, N., 2020. A study of the privacy of COVID-19 contact tracing apps. In: International Conference on Security and Privacy in Communication Systems. SecureComm, Springer, pp. 297–317. http://dx.doi.org/10.1007/978-3-030-63086-7_17.

White, L., van Basshuysen, P., 2021. Without a trace: Why did corona apps fail? J. Med. Ethics 1–4. http://dx.doi.org/10.1136/medethics-2020-107061.

Wymant, C., Ferretti, L., Tsallis, D., Charalambides, M., Abeler-Dörner, L., Bonsall, D., Hinch, R., Kendall, M., Milsom, L., Ayres, M., et al., 2021. The epidemiological impact of the NHS COVID-19 app. Nature 594 (7863), 408–412. http://dx.doi.org/10.1038/s41586-021-03606-z.

Zimmermann, T., 2016. Card-Sorting: From Text to Themes. Elsevier, pp. 137–141.

Richard May is a startup founder and Ph.D. student at Harz University of Applied Sciences Wernigerode, Germany. In 2020 he received his M.Eng. degree in Technology and Innovation Management at Harz University of Applied Sciences. Currently, his research topic are scalable cyber-security concepts in the context of highly-configurable systems.

Niklas Baron is a startup founder from Wernigerode, Germany. In the context of his studies in Business Consulting (M.A.) at Harz University of Applied Sciences Wernigerode, he founded his first startup in 2020. During the last years, he was part of various workshops as well as keynote speaker at startup events, sharing his experiences of founding and leading startups.

Jacob Krüger is Assistant Professor in the Software Engineering and Technology cluster of Eindhoven University of Technology, The Netherlands. He obtained is Ph.D. degree in 2021 at Otto-von-Guericke University Magdeburg, Germany. He has also worked as associated researcher at Ruhr-University Bochum, Germany, as well as Harz University of Applied Sciences, and as visiting researcher at University of Toronto in Canada as well as Chalmers | University of Gothenburg in Sweden. His research focuses on software evolution, software re-engineering, human factors, product lines, and maintenance.

Thomas Leich is Professor at the chair of Business Information Systems at Harz University of Applied Sciences, Wernigerode, Germany. He received his diploma in Business Information Systems and his Ph.D. from Otto-von-Guericke University Magdeburg, Germany, in 2004 and 2012, respectively. Since 2013 he is executive director of the METOP GmbH, an affiliate institute to the Otto-von-Guericke University Magdeburg. His research interests include requirements and software product-line engineering as well as measurement of program comprehension and data management.