THE EFFECT OF TIME PRESSURE DURING COVID-19 ON SOFTWARE QUALITY

Asmaa Hassan[1,2] and Omer Alrwais[1]

[1] Department of Information Systems, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia
[2] Department of Information Systems, College of Sciences and Arts, King Khalid University, Abha 62217, Saudi Arabia

ABSTRACT

Background: as technology develops, governments, organizations, companies, and individuals develop technical solutions for global problems or crises. Currently, the entire world is experiencing the covid-19 pandemic. During this crisis, efforts have been made to find a technical solution to face or even mitigate the crisis. The development of technological solutions under time pressure is one issue being explored by many researchers. We seek to determine the impact of time pressure on the quality of the developed software or applications, especially in the case of the covid-19 crisis in Saudi Arabia. **Method**: this research uses interviewing, survey, and analysis of user comments and ratings for the application (the application was developed under time pressure in the covid-19 crisis) in the app store. **Results**: the research results demonstrated that software quality in this study was positively affected by time pressure. The impact of time pressure on software quality likely depends on a few factors, such as the level of time pressure, the experience of the developers, and the type of software being developed. **Conclusions**: the development teams can perform well under time pressure. Similarly, the challenge–hindrance framework maintains that time pressure may be a positive source of pressure (challenge) and translate into good stress or a negative source of pressure (hindrance) that corresponds to bad stress. In other words, outcomes derived under time pressure can be positive and negative; pressure is positive when it increases efficiency and negative when it reduces quality. Project managers can maintain quality while working under time pressures. To put it in another way, under optimal pressure, software development teams can complete projects in less time, with less effort, and with good quality. Since the ABC application was developed to improve people's health and restore normalcy after quarantine, the developers prioritized its development as a mission of national interest. We concluded that the type of time pressure affects app quality, either negatively or positively.

KEYWORDS

Software Development, Software Quality, Time Pressure, Covid-19

1. INTRODUCTION

Software quality refers to the degree to which a system and process are designed to achieve their purpose and meet all expectations of the user [1]. Typically, the software is described as having a high quality if its functionality fulfills the purpose for which it was developed [1]. However, achieving functional quality is only sufficient if a software's interface is usable, and it cannot be considered high quality. Attaining quality is one of the main goals of software development. It is also one of the most challenging goals to meet and is affected by many factors, including time pressure. Software development quality under time pressure has become an interesting topic, especially during crises such as the outbreak of Covid-19.

All governments, in cooperation with software developers, try to create software to overcome crises within a short amount of time. Time pressure means accomplishing a difficult task within this time; it considerably affects software development, particularly regarding product quality. Furthermore, time pressure has negative psychological effects on both the development team and the stakeholders; it sometimes forces software developers to resort to taking shortcuts, which may, in turn, affect the quality. That is why it has many negative consequences.

Software development requires careful planning, design, and testing to obtain a product that matches the user's desire. This is the most important goal for software development. The idea that time pressure can only have a negative effect on software development is prevalent. However, some studies have indicated that time pressure can also positively impact, as it may catalyze increased productivity [2]. Furthermore, professionals maintain that software engineering quality is affected by the quality of the product-building process [3].

During the outbreak covid-19, a global pandemic, the virus has affected all areas, including the field of software development. Software development has confronted the pandemic by arriving at solutions, including those that help with the practice of social distancing. Software development during the COVID-19 crisis has been conducted under time pressure to contribute to finding helpful solutions as quickly as possible. All the world's industries and businesses have resorted to working remotely, including technology companies and software producers; this may be one of the factors affecting the quality of software products. Some studies have revealed that employees working-remotely are more productive [2], but this affects software quality; achieving high quality remains a significant goal, even with the fast pace of software development.

The authors of [4] investigated how time pressure affects quality and found that a loose schedule leads to increased production defects. Therefore, the relationship between time pressure and quality was U-shaped (Figure. 1).

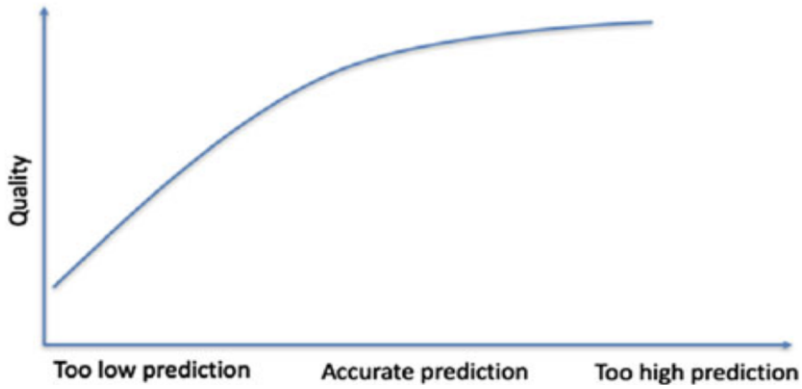

Figure. 1 U-shaped relationship between pressure and quality [4].

The too-low estimate compared to the actual amount of work leads to reduced quality; the too-high estimate does not lead to increased quality over that achievable with a realistic, accurate schedule; and the accurate estimate leads to increased quality [4].

This research aims to study the effect of time pressure during the COVID-19 crisis on the quality of developed software and the measures taken during this pandemic that has increased or decreased such software's quality.

2. LITERATURE REVIEW

In this research, the following areas of research have been examined:

- Software quality;
- Software quality under time pressure.

2.1. Software quality

This section highlights when software can be judged to have the desired quality or to satisfy quality requirements.

The authors of [5] conducted a study to ascertain how quality is seen and measured. The authors found that one of the main problems facing project managers and software professionals is getting the process right for the first time. This finding indicates that a rigorous, defined, measurable quality regime must be followed [5]. Furthermore, the quality of the processes used to create a software system influences the quality of the software system itself [5]. Moreover, quality is affected not only by procedures, tools, or systems but also by people and their expectations [5].

Thus, guaranteeing that software is of high quality necessitates an understanding that quality is a multifaceted issue that changes with context, sometimes even for the same individual, at different points in time. For example, some individuals need a highly secure application without worrying about the possibility that the product may be unattractive. By contrast, a neophyte user may prefer software that is easy to use over one that is secure. This simple example shows the need to consider the user population when judging quality [5].

2.2. Software quality under time pressure

This [6] study aims to present an overview of studies on time pressure in software engineering, its effects on software development, and the relationship between time pressure and software quality and productivity. Most high-quality studies report increased productivity but decreased quality when working under time pressure

The timely development of software has been a significant issue and challenge for information systems research and the software industry [2]. Time pressure from schedule constraints or crises is a problem confronting software developers. Quality issues occur when developers take shortcuts to cope with unanticipated problems and meet task deadlines. These shortcuts are private decisions that may not serve the interests of a project [7]. An example of an occurrence that causes time pressure is the crisis that the world is currently facing—the COVID-19 pandemic.

The authors of [2] studied the impact of scheduling-induced pressure on performance in software projects. The authors reported that some psychologists typically consider high pressure to result in adverse outcomes; conversely, Parkinson (1957) believed that tremendous pressure engenders greater and better productivity and performance. The U-shaped relationship proposed by other researchers, as mentioned in the introduction, reconciled these two contradictory views.

Research published in January 2020 [6] aimed to provide an overview of studies related to time pressure in software engineering. This review indicated that teams can perform well under time pressure. Similarly, the challenge–hindrance framework maintains that time pressure may be a positive source of pressure (challenge) and translate into good stress or a negative source of

pressure (hindrance) that corresponds to bad stress. In other words, outcomes derived under time pressure can be positive and negative; pressure is positive when it increases efficiency and negative when it reduces quality [8].

A literature review conducted by the researcher [9] reveals a diversity of time-pressure conceptualizations and their impact on software projects.The reported findings indicated that severe levels of time pressure should be avoided. Otherwise, members of software projects might adapt their behavior in ways that negatively affect project success, and time pressure should only be used as a short-term measure. This is because working under high time pressure for long periods lowers employee morale and increases turnover [9].

One study [10] observed a software development project within a large telecom company for ten months. According to preliminary findings, many decisions made under the pressure of specific circumstances had a negative impact on software quality.

The researchers attempted to evaluate the impact of schedule pressure on software quality [11]. The researchers indicated that applying time pressure to software development teams helps reduce development time. Managers can improve their teams' productivity by setting stricter schedules. This beneficial effect of pressure, however, has a limit. When management cuts too much time off a team's expected schedule, developers become burned out and take longer to achieve their tasks. Nevertheless, project managers can maintain quality while working under time pressures. To put it another way, under optimal pressure, software development teams can complete projects in less time, with less effort, and with good quality [11].

In the literature review [12], researchers noted that scientific findings have indicated that when software development is under time pressure, efficiency increases but quality decreases. The positive side of the time pressure is that the development team focuses on the prominent features of the product, while the negative side is that they prioritize delivery according to the deadline over the quality of the product. They also indicated that the type of task affects efficiency and quality. For example, for complex tasks such as tasks with high algorithmic natures (e.g., database queries), completing them under time pressure may improve efficiency less and decrease quality more [12].

Table 1 Studies on the effect of time pressure on software quality

Title	Author(s)	Year	Outcomes
Time pressure in software engineering: A systematic review	M. Kuutila, M. Mäntylä, U. Farooq, and M. Claes	2020	The majority of high-quality studies report increased productivity but decreased quality when working under time pressure.
What Do We Know About Time Pressure in Software Development?	M. Kuutila, M. Mantyla, U. Farooq and M. Claes	2020	When software development is under time pressure, efficiency increases but quality decreases.
The Role of Time Pressure in Software Projects: A Literature Review and Research Agenda	Basten, Dirk	2017	The reported findings indicated that severe levels of time-pressure should be avoided. Otherwise, members of software projects might adapt their behavior in ways that

			negatively affect project success, and time pressure should only be used as a short-term measure.
Why good developers write bad code: An observational case study of the impacts of organizational factors on software quality	Lavallée, Mathieu, and Pierre N. Robillard	2015	According to preliminary findings, many decisions made under the pressure of specific circumstances had a negative impact on software quality.
Software quality factors and software quality metrics to enhance software quality assurance	M. Lee	2014	Challenge–hindrance framework maintains that time pressure may be a positive source of pressure (challenge) and translate into good stress or a negative source of pressure (hindrance) that corresponds to bad stress. In other words, outcomes derived under time pressure can be both positive and negative;
Impact of budget and schedule pressure on software development cycle time and effort	N. Nan and D. Harter,	2009	Achieving the potential positive effect of schedule pressure requires cooperation between clients and software development teams.
The impact of schedule pressure	N. Nan, D. Harter, and T. Thomas	2003	Under the optimal level of pressure, software development teams can complete projects in less time, with less effort, and with good quality.

Overall, the findings of these studies are mixed. For example, some studies have found that time pressure has a negative impact on software quality, while others have found that it can have a positive effect.

3. METHODOLOGY

The purpose of the present study is to conduct qualitative research and build a grounded theory of how software development under time pressure during the COVID-19 pandemic affects software quality.

The methodology focuses on collecting experiences mainly through interviews; the purpose is to identify the most critical factors that positively or negatively affect the quality of applications during the COVID-19 crisis. The case research strategy is ideal for gathering practitioner knowledge and generating theories. We follow the grounded theory approach because it enables the identification of new theories and concepts. The primary purpose of this study is to determine and represent any relationships between time pressure and software quality. We intend to collect information through interviews and questionnaires and observe the application launched during the COVID-19 crisis [25].

The company selected for the study from amongst those that released applications during the COVID-19 pandemic in Saudi Arabia. The data will be collected from interviews with and observations of technical staff working in the quality departments of this software company in Saudi Arabia.

Our study focuses on software developed urgently during the Covid-19 crisis, either for community service reasons (such as checking physical distancing or sorting out infected cases) or to allow an institution to continue work under crisis conditions or during quarantine. We conducted interviews and surveys and studied user comments about the developed application under the pressure of the Covid-19 crisis. We will supplement this with theoretical explanations of how time pressure during COVID-19 affects quality and analyze the interviewees' perceptions of the current state of applications' quality in their organizations. The results of this study may contribute to determining the effects of time pressure on software quality and the factors that increase and decrease this quality [13][14].

ISO 9126 defines a set of attributes and metrics that describe software quality's internal and external aspects, including reliability, usability, maintainability, efficiency, and portability. These definitions will guide our selection of interview and questionnaire questions of software quality.

Software Quality Attribute	**Sub characteristics**	**Definition**
Functionality		**Meet stated and implied needs of the user**
	Suitability	Provide an appropriate set of functions for specified tasks and user objectives
	Accuracy	Provide the right or agreed results with the needed degree of precision
	Security	Provide information/data access to only authorized persons/systems
	Interoperability	Interact with one or more specified systems
	Compliance	Adhere to standards, conventions, regulations in laws
Reliability		**Maintain a specified level of performance when used under specified conditions**
	Maturity	Avoid failure due to faults in the software
	Fault tolerance	Maintain a specified level of performance in case of software faults
	Recoverability	Re-establish a specified level of performance in case of failure
Usability		**Be understood, learned, used, and attractive to the user**
	Understandability	User knows how the product can be used for particular tasks
	Learnability	User is able to learn the application of the product
	Operability	User can operate and control the product
	Attractiveness	User likes the product
Efficiency		**Provide an appropriate level of performance**
	Time behavior	Provide an appropriate response time
	Resource utilization	Efficient use of resources
Maintainability		**Can be modified to incorporate changes in the functional and technical specifications**
	Analyzability	Can be diagnosed for modification or causes of failure
	Changeability	Enable a specified modification to be implemented
	Stability	Avoid unexpected effects from modifying the software
	Testability	Enable modified software to be validated
Portability		**Can be transferred from one environment to another**
	Adaptability	Can be adapted for different environments
	Installability	Can be installed in a specified environment
	Co-existence	Can co-exist with other software in an environment that shares common resources
	Replaceability	Can be used instead of another product for the same purpose in the same setting

Figure. 2 Definitions of Software Quality Attributes [15]

This research will study the ABC application (a fictitious name that will be used through this article instead of the application's real name) developed during the COVID-19 pandemic by a leading government software development company in the Kingdom of Saudi Arabia. The application will be referred to as ABC due to an agreement with the developer company to preserve privacy. The goal of the application was to maintain the health and safety of the citizens and residents of the Kingdom of Saudi Arabia—27 million users. The app development was carried out in cooperation with the Ministry of Health and several government agencies during the imposed ban to help limit the spread of COVID-19 in the Kingdom of Saudi Arabia. As a result, the application launched several critical services that contributed to achieving a safe return to normality.

The research data collection and Analysis involved the following steps:

A. Analysis of 506 user comments and 130.000 ratings for the ABC application in the App Store.
B. Interviews with nine technical staff at the software development companies in the Kingdom of Saudi Arabia developed for the target application in this article. In addition, we contacted some senior QA engineers and employees and obtained their consent to be interviewed.
C. Survey to track user satisfaction with the ABC application.
    a. The survey questionnaire was prepared in English and Arabic to reach the targeted users.
    b. We targeted students at King Saud University because they belong to the category of people who need to use the application studied in this work.
    c. The survey was sent to 3,000, and 116 valid responses were received.
D. Analysis of the data collected to understand the impact of time pressure on software quality.

We followed the five steps of grounded theory: describing the research question, conducting a literature review, describing the methodology, performing data analysis to explain the theory, and discussing the implications [16]. As explained in [14], 'the choice of grounded theory as the research method is justified because the need for qualitative approaches in the areas related to human behavior is recognized widely also in software engineering research.'

5. POPULATION AND SAMPLING

The ABC application was chosen for this research because it was one of the officially approved apps in Saudi Arabia developed to limit the spread of COVID-19, promote health, facilitate the movement of its residents, and regulate the scheduling of vaccine doses. The app targeted all the citizens and residents of the Kingdom of Saudi Arabia. The application was available in several languages, such as Arabic, Urdu, English, Indonesian, Bengali, Filipino, and Hindi. Students from King Saud University made up the study population because they were obligated to use the application being studied in this research and because they belonged to an academic setting with a high level of technological expertise.

We contacted the company that developed the application and received approval to communicate and conduct interviews with nine employees. It was agreed upon with the developer company not to mention the application's name in the paper to preserve privacy. We had the cooperation of the nine employees of the company and those in charge of the ABC application, and they are listed below.

- Senior expert in the company
- Manager
- Project Manager
- Quality Manager
- Software Quality Team Leader
- Quality Assurance
- Quality engineer
- Quality tester
- Business systems analyst

The interview details are provided in Appendix A. The interviews were conducted over the phone, and the employees were given a choice between two languages, either Arabic or English, for communication. The duration of the interview was 20–25 minutes. We directed the questions

listed in Appendix A to the employees working on the ABC app during the COVID-19 crisis. The questions comprised two parts. The first part included personal questions and questions about the differences in the app development process before and after the COVID-19 crisis. The second part was prepared following ISO 9126 and focused on the characteristics of the ABC application. We targeted a set of attributes and metrics that describe the internal and external aspects of the software quality of the ABC application.

We analyzed user reviews and ratings for the ABC app in the App Store. The user comments on the app was extracted using Python code, and an app-store-scraper tool was used. We also used the App Follow tool, making it easy to track an application and know what consumers think about it through ratings and reviews. The app had more than 103,000 ratings and 500 comments.

The study survey population comprised students from King Saud University since they were obligated to use the application being studied in this research and because they belonged to an educational community with good technical knowledge. The questionnaire provided in Appendix B was published and distributed to the students at King Saud University through their unified groups via Telegram. The target group comprised 36,788 bachelor's students and 420 master's and doctoral students. Around 116 questionnaires were received from the participants. The questionnaire was designed using Microsoft Forms in both Arabic and English, and the participants could select the language of their choice. The questionnaire began with a cover letter that stated the purpose of the study and assured the protection of participants' privacy. This helped ensure that participants understood the context of the questions, which encouraged them to provide thoughtful and accurate responses. Following that, multiple-choice questions were included about the characteristics related to the quality of the targeted application that the user can measure, such as functionality, reliability, usability, efficiency, and portability. The questions were designed based on ISO 9126. The questionnaire consisted of 18 items and used a Likert scale [17] to gather participant data.

Likert scales are a standard survey rating format. Respondents score quality from excellent to poor or from best to worst using five or seven levels. This format is a way to measure attitudes, opinions, and perceptions [17]. This study asked the respondents to rate their level of agreement or acceptance using a Likert scale ranging from 1 to 5, where 5 represents the highest acceptance degree, and 1 represents the lowest acceptance degree, as described in Table 1.[18]

Table 2 Respondent Likert scale

Level	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
Scale	1	2	3	4	5
Mean Range	1.0-1.8	1.8-2.6	2.6-3.4	3.4-4.2	4.2-5.0
Weight means	20%-36%	36%-52%	52%-68%	68%-84%	84%-100%

For all the questionnaire items, the degree of responses for each item will be determined based on the five-point division (Likert scale). Values from 1 to less than 1.8, 1.8 to less than 2.6, 2.6 to less than 3.4, 3.4 to less than 4.2, and 4.2 to 5.0 represent 'strongly disagree,' 'disagree,' 'medium,' 'agree,' and 'strongly agree,' respectively.

Statistical Validity of the Questionnaire: To ensure the validity of the questionnaire:

3.1.1 Internal consistency

Internal consistency [9, 20] of the questionnaire was measured using a scouting sample, which consisted of thirty questionnaires, by measuring the correlation between each item in the construct and the whole construct. Table (2) below shows the correlation coefficients for each construct item. All correlations were significant, indicating that the items in all the constructs were consistent and valid for measuring what we intended to measure.

** Correlation is significant at 0.01 level

Table 3. The correlation coefficient between each item in the field and the whole field

Functionality		**Reliability**		**Usability**		**Portability**		***Efficiency***	
1	.875**	1	.753**	1	.823**	1	.817**	1	*.856***
2	.823**	2	.806**	2	.897**	2	.751**	2	*.741***
3	.872**	3	.837**	3	.808**			3	*.818***
4	.826**			4	.831**			4	*.853***
5	*.810***								

6. ANALYSIS

6.1. Interview Questions:

The first part included personal questions and questions about the differences in the app development process before and after the COVID-19 crisis. The interviews showed that seven responses indicated that the company developed applications independently. In comparison, two responses indicated that the software was developed in partnership and cooperation with leading companies in the Kingdom of Saudi Arabia. All responses indicated a difference in the development process and quality assurance before and after the COVID-19 crisis, and this difference was also evident in the roles and nature of the work.

Remote work from home during this crisis ensured better assurance of quality. Some of the employees said that at the beginning, working from home reduced the quality of the products and affected their quality of life and timekeeping, but it was for a short period. After continuing to work from home, the employees' working times became organized, and they got used to the new method. Although there was attention to quality before the crisis, afterward, there was a greater focus on quality because even minor errors could affect the course of life in Saudi Arabia. Thus, each amendment is approved only after various quality tests.
Language auditing has become more critical; previously, linguistic errors were dealt with as simple matters.

Regarding the strategy for measuring and reviewing quality that they had to follow after the COVID-19 crisis, all employees agreed that the project's plan and work methods are the same as before the crisis. The differences are that they work from home, and their working hours have increased. Some strategies have been followed for a better way to handle the epidemic, such as performance testing (a process of testing an application using a load-generating tool to simulate real users to identify system bottlenecks [21]), security testing (looks for vulnerabilities and malicious applications [22]), automation testing (automate the entire process of testing [21]) and unit testing and regression testing (practice to find and fix issues caused on by defects injected a posteriori [23]).

When we asked the employees about the strategy while measuring and reviewing the quality they had to dispense with after the COVID-19 crisis, they answered that there needed to be a clear new strategy. However, there was interest in the time factor, a quality review was in progress, and the company's network was opened so employees could access it from home. There was also a shortcut to some points to save time, such as designing the test life cycle as a plan that does not exist in writing, and there was knowledge of every step, even if it was written or documented.
We then asked the employees about procedures that positively affected the software quality produced during the COVID-19 crisis. Their answers included remote work, automation testing, unit testing, regression, zero blockers, and access to the network from home, as the request for approval to complete the work within the company's network required a long time to be approved. For example, the quality auditor had remote access during and after the crisis, and there were additional working hours. As for procedures that negatively affected software quality, most employees confirmed no negative impact on the quality because the team was carefully selected, and the tests were intensified to avoid errors.

We concluded the first part of the interview by asking about the number of errors discovered during audits before and after the COVID-19 crisis. They answered that the team working on the application was effective, and the number of errors was almost the same. However, more time was needed to discover errors before the crisis; during the crisis, errors were discovered more quickly and resolved faster. Errors appeared because employees accomplished more than one task simultaneously. The errors differed from one programmer to another, but the user did not see them.

The second part was prepared following ISO 9126 and focused on the characteristics of the ABC application.

All employees confirmed that the application provides appropriate functions for specific tasks and user goals. They also agreed that the work is done around the clock, and several test procedures were used to assess the work, issue the application with the required accuracy, and ensure that the results of the ABC application were correct and more accurate than other applications during the crisis.

The consensus was that ABC provides access to information only to authorized persons and systems, which was confirmed by a two-stage audit. The application interacts with many systems, such as the National Information Centre, to provide information to users, ministries, citizens, and residents. It also acts as a collection system. The application adheres to standards, agreements, and legal regulations, and a special team explained any legal dimensions or questions to the employees before any part was issued. This included state regulations and device systems, the most important reason for obtaining international awards. (Functionality).

The results showed software errors in which the server falls or hangs, but it returns within minutes or hours, and most of the errors are at the server. The consensus is that the ABC application maintains a specified level of performance in the event of a software failure and has a recovery plan. One participant mentioned that sometimes, failures bring down an entire system. The employees agreed that the ABC application re-establishes a certain level of performance in case of failure, during which the recovery plan would be activated. (Reliability)

Also, the users of the ABC application know how to use it for specific tasks. The consensus is that users of the ABC application can learn the program because each step is clear to the next, and each new service that is launched shows how to use it, and users of the ABC application can run and control the software as permitted. (Usability)

The consensus is that ABC provides a very good response time at the application level. The ABC application is efficient in its use of resources, and this was one of the goals. (Efficiency)

Most employees agreed that the ABC application could be diagnosed for modification at a very high level. (Maintainability)

The ABC application can adapt to different environments and is tested on different operating systems (iOS, Windows, Android, and the HarmonyOS) and services (Wi-Fi, 2G, or 3G). (Portability)

6.2. Questionnaire Questions:

The questionnaire aims to measure the impact of time pressure during Covid-19 on software quality. And from the analysis of all questionnaire domains, the ABC application meets all the quality characteristics, and the opinions of the questioners ranged between strongly agreed and agreed. Thus, according to the participants, the effect of time pressure on the ABC application was not negative since it meets all the characteristics.

Figure. 3 All domains of time pressure during covid-19

As shown in Figure.8, Portability and Reliability were the lowest based on the weighted mean. Through our analysis of user reviews of the ABC application, most of them mentioned that the application does not work for those who jailbreak their phone, which is the process of breaking the restrictions imposed by the phone manufacturer. This could be one of the reasons that affected Portability. As for reliability, the employees indicated that the failure of the entire system that the application was exposed to and appeared to the user only was due to the service provider. It was resolved in a short time, less than 24 hours.

6.3. Analyzing user comments for ABC application in the App Store:

The application was launched in May 2020. The overall app rating on the App Store was 3.7 out of 5. The total number of ratings for the application since its inception was 103,267, and the total number of comments was 506. The total number of ratings that rated the application with five stars since its release was 63,837, while there were 19,526 1-star ratings (Table (4), Figure. (4), and Figure. (5)).

Table 4 The number and percentage of ratings

Year	Category	Frequency	Percent
Since Its Release Until today	5	63,837	61.8
	4	9,269	9
	3	5,837	5.7
	2	4,798	4.6
	1	19,526	18.9
	Total	103,267	100.0

Figure. 4 Total ratings counts

Figure. 5 Reviews

After extracting the comments using Python and displaying them using the AppFollow tool, it became clear to us that the reviews associated with one-star ratings were the most common, as shown in the pie chart in Fig. 10. People are more likely to leave a negative review than a positive review as Pete Blackshaw said: “satisfied customers tell three friends angry customers tell 3,000” [24].

7. RESULTS

The research results demonstrated that software quality was positively affected by time pressure. In contrast, the application has become one of the most prominent and influential solutions for promoting health in the Kingdom of Saudi Arabia. It also won two awards for institutional resilience and innovative responses to the COVID-19 pandemic and for the efficiency of a system for crisis management. The study indicated that the nature of the crisis, the cause of time pressure, the application developer's experience, and the target group's size might affect the severity of the impact on the quality. Since the ABC application was developed to help people by promoting their health and safety during COVID-19, a major health crisis, the concern for its quality was high because app errors may significantly impact people's health. Since the application targets millions of citizens and residents of the Kingdom of Saudi Arabia, correcting app errors after its launch has a significant negative impact due to the large number of users receiving the service.

Additionally, the application developer was a government company under the supervision and follow-up of the government of the Kingdom of Saudi Arabia. These may be the reasons behind the no negative impact on app quality and high efficiency, which resulted in the system winning first place for its crisis management efficiency. However, the results might not generalize because of the app's sensitivity and the team's size. Nevertheless, by comparing the evaluations of the previous applications of the developing company, it became clear that the application was at a good rate, as the ratings of the developed company's applications on the App Store were between 4.8 and 2.5 out of 5. This indicates that the quality of the application by this company was not affected due to time pressure during the Covid-19 crisis.

8. DISCUSSION

The main research question was as follows: How did time pressure affect software quality during the COVID-19 crisis?

Through our analysis of the interviews, the questionnaire, and the user comments and ratings for the ABC application, we concluded that time pressure did not negatively affect the quality of the application. Rather, the developers of the ABC application agreed that time pressure motivated them to release the application with high quality.

Most study participants agreed that no new quality assurance strategies were implemented during the COVID-19 crisis. Instead, the company followed the same strategies used during the pre-COVID-19 era. However, employees' sense of responsibility towards quality was high because the application aimed to enhance health and significantly impacted people's quality of life during the COVID-19 crisis. They agreed that employees' remote work positively impacted the application's quality because the working hours were greater, with an increased ability to access the application 24 hours a day compared to the limited access before the COVID-19 crisis. Pre-COVID-19 era, the developers or the quality auditor primarily accessed the applications during working hours. Some participants mentioned that before the COVID-19 crisis, when an error was discovered in the application at the end of a workday, employees were required to postpone working on a solution to address the error the next day, which may have affected the quality. They also indicated that abbreviated writing and documentation had a positive effect on speeding up work without negatively affecting quality.

The development team of the ABC application was carefully selected and highly skilled, which positively influenced app quality. Since the app targeted all citizens and residents of the Kingdom of Saudi Arabia, adapting and increasing the number of performance tests improved performance quality. Increasing the quality audit tests for the entire system also enhanced app quality. Some study participants indicated that before the COVID-19 crisis, only one test for the entire system was conducted before the customer acceptance test. In contrast, the developers were keen on ensuring the entire application passed two testing stages before the customer acceptance test. They indicated that manual testing was used in the pre-COVID-19 era. Still, after increased time pressure due to the pandemic, automation testing was implemented, accelerating the testing process and enhancing app quality.

Substantial attention was given to linguistic proofreading, and a legal language check was also implemented because any linguistic error may affect people’s lives, negatively affecting app quality. The developers mentioned that the quality review was conducted continuously daily. Before the COVID-19 crisis, the frequency of quality reviews was once a month. The developers focused on a few features at a time to be reviewed, then released them, and then moved on to develop and release other features.

Working from home had a negative impact on the employees’ quality of life but not on the quality of the application. Another negative impact was attributed to the fact that employees were required to be present most of the time, sometimes 24 hours a day. This was especially applicable when a technical problem was discovered because employees were required to solve the problem immediately due to added time pressure. However, all study participants avoid that the greater the time pressure, the lower the quality. Therefore, work was performed continuously and cooperatively among all employees because they viewed the application as a mission of national interest and humanitarian work.

9. CONCLUSIONS

Overall, the findings of previous studies are mixed. Some studies have found that time pressure has a negative impact on software quality, while others have found that it can have a positive effect. The impact of time pressure on software quality likely depends on several factors, such as

the level of time pressure, the experience of the developers, and the type of software being developed.

We agreed with research published in January 2020 [18] aimed to provide an overview of studies related to time pressure in software engineering. This review indicated that teams can perform well under time pressure. Similarly, the challenge–hindrance framework maintains that time pressure may be a positive source of pressure (challenge) and translate into good stress or a negative source of pressure (hindrance) that corresponds to bad stress. In other words, outcomes derived under time pressure can be positive and negative; pressure is positive when it increases efficiency and negative when it reduces quality [19].

One of the studies that are consistent with the results of our research indicated that applying time pressure to software development teams helps reduce development time. Managers can improve their teams' productivity by setting stricter schedules. This beneficial effect of pressure, however, has a limit. When management cuts too much time off a team's expected schedule, developers become burned out and take longer to achieve their tasks. Project managers can maintain quality while working under time pressures. To put it another way, under optimal pressure, software development teams can complete projects in less time, with less effort, and with good quality [20]. This we agree with.

Since the ABC application was developed to improve people’s health and restore normalcy after quarantine, the developers prioritized its development as a mission of national interest. We concluded that the type of time pressure affects app quality, either negatively or positively.

The developers of the ABC application, whom we interviewed, indicated that time pressure due to the COVID-19 health crisis motivated implementing the application at its best quality to preserve people’s health and help them return to their everyday lifestyles. Future research should examine whether the difference in the cause of time pressure affects the degree of impact on the quality of the application, either positively or negatively. Research on the effects of time pressure is extensive. Time pressure is one of the factors affecting quality.

Future studies may consider expanding the evaluation to include more applications developed by various parties in the government or private sector during the COVID-19 crisis.

10. LIMITATIONS OR CHALLENGES:

We faced many difficulties, including communicating with developers and obtaining permission from application development companies to study their applications.

We also point out some limitations: the developers have rated the app they built, and an evaluation of the application by external experts may give a more accurate presentation of the quality of the application. As we indicated, the results might not generalize because of the app's sensitivity and the team's size.

ACKNOWLEDGEMENTS

The authors would like to thank everyone, just everyone!

REFERENCES

[1] Petrasch, "The definition of 'software quality’: A practical approach," 1999. [Accessed 12 October 2020].

[2] N. Nan and D. Harter, "Impact of budget and schedule pressure on software development cycle time and effort", IEEE Transactions on Software Engineering, vol.35, no.5,2009, pp.624 637.Available:10.1109/tse.2009.18. [Accessed 3 May 2021].
[3] J. McManus and T. Wood-Harper, "Software engineering: a quality management perspective," The TQM Magazine, vol.19, no. 4, 2007, pp.315-327.
[4] T. Halkjelsvik and M. Jørgensen, Time Predictions: Understanding and Avoiding Unrealism in Project Planning and Everyday Life. [Accessed 20 May 2021].
[5] J. McManus and T. Wood-Harper, "Software engineering: a quality management perspective", The TQM Magazine, vol. 19, no. 4, 2007, pp. 315-327. Available: 10.1108/09544780710756223. [Accessed 31 December] 2020].
[6] M. Kuutila, M. Mäntylä, U. Farooq, and M. Claes, "Time pressure in software engineering: A systematic review," Information and Software Technology, vol. 121, p.106257, 2020. [Accessed 16 Oct 2021].
[7] Austin, Robert D. "The effects of time pressure on quality in software development: An agency model." Information systems research, 2001 [Accessed 20 May 2021].
[8] M. Lee, "Software quality factors and software quality metrics to enhance software quality assurance", British Journal of Applied Science & Technology, vol. 4, no. 2, 2014, pp.3069-3095.Available: 10.9734/bjast/2014/10548 [Accessed 10 April 2021].
[9] Basten, Dirk, "The Role of Time Pressure in Software Projects: A Literature Review and Research Agenda" (2017). International Research Workshop on IT Project Management 2017. 1. https://aisel.aisnet.org/irwitpm2017/1 [Accessed 14 Feb 2022].
[10] Lavallée, Mathieu, and Pierre N. Robillard, "Why good developers write bad code: An observational case study of the impacts of organizational factors on software quality." 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. Vol. 1. IEEE, 2015. [Accessed 14 Feb 2022].
[11] N. Nan, D. Harter, and T. Thomas, "The impact of schedule pressure on software development: A behavioral perspective." ICIS 2003 Proceedings, vol. 71, 2003. [Accessed 16 Oct 2021].
[12] M. Kuutila, M. Mantyla, U. Farooq and M. Claes, "What Do We Know About Time Pressure in Software Development?", IEEE Software, vol. 38, no. 5, pp. 32-38, 2020. Available: 10.1109/ms.2020.3020784. [Accessed 15 Feb 2022].
[13] N. Parikh, "Strategies for E-Commerce Platform Adoption in the Manufacturing Sector in Western India," Doctoral dissertation, Walden University, 2016. [Accessed 16 May 2021].
[14] O. Taipale, J. Kasurinen, K. Karhu, and K. Smolander, "Trade-off between automated and manual software testing", International Journal of System Assurance Engineering and Management, vol. 2, no. 2, 2011, pp. 114- 125.Available: 10.1007/s13198-011-0065-6 [Accessed 16 May 2021].
[15] P. Padmanbahan, Software quality perceptions of stakeholders involved in the software development process. 2013. [Accessed 22 Feb 2022].
[16] C. Williams, "Research Methods", Journal of Business & Economics Research (JBER), vol. 5, no. 3, 2011. Available: 10.19030/jber. v5i3.2532 [Accessed 20 May 2021].
[17] ALLEN, I. Elaine; SEAMAN, Christopher A. "Likert scales and data analyses". Quality progress, 2007, 40.7: 64-65. [Accessed 8 May 2022].
[18] KADA, Zaoui. Inner Evaluations of Language Attitudes among Algerian EFL Students. [Accessed 10 Apr 2023]
[19] Brains, Willnat, Manheim, Rich 2011. Empirical Political Analysis 8th edition. Boston, MA: Longman p. 105], [Kramer, Geoffrey P., Douglas A. Bernstein, and Vicky Phares. Introduction to clinical psychology. 7th ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2009 [Accessed 14 May 2022].
[20] BROWN, Ted; BONSAKSEN, Tore. An examination of the structural validity of the Physical Self-Description Questionnaire-Short Form (PSDQ–S) using the Rasch Measurement Model. Cogent Education, 2019, 6.1: 1571146. [Accessed 20 Sep 2022].
[21] SAROJADEVI, H. Performance testing: methodologies and tools. Journal of Information Engineering and Applications, 2011, 1.5: 5-13. [Accessed 10 Apr 2023].
[22] WANG, Yong; ALSHBOUL, Yazan. Mobile security testing approaches and challenges. In: 2015 First Conference on Mobile and Secure Services (MOBISECSERV). IEEE, 2015. p. 1-5. [Accessed 10 Apr 2023].

[23] ROSERO, Raúl H.; GÓMEZ, Omar S.; RODRÍGUEZ, Glen. 15 years of software regression testing techniques—a survey. International Journal of Software Engineering and Knowledge Engineering, 2016, 26.05: 675-689. [Accessed 10 Apr 2023].
[24] BLACKSHAW, Pete. Satisfied customers tell three friends, angry customers tell 3,000: running a business in today's consumer-driven world. Currency, 2008. [Accessed 11 Apr 2023].
[25] K.S. Sherif, Actualizing software reuse: a qualitative study of barriers to adoption. Texas A&M University, 1998. [Accessed 20 May 2021].
[26] M. Xenos, "Software metrics and measurements", in Encyclopedia of E- Commerce, E-Government and Mobile Commerce, Mehdi Khosrow-Pour (Ed.), Idea Group Publishing, no. 1-59140-799-0, 1029-1036, 2006. [Accessed 10 November 2021].
[27] The Economic Times. "What is software engineering? Definition of software engineering, software engineering meaning," The Economic Times, 2020 [online] Available :< https://economictimes.indiatimes.com/definition/software-engineering> [Accessed 6 October 2020]
[28] S. Russell, T. Bennett, and D. Ghosh. "Software engineering principles to improve quality and performance of R software," PeerJ Computer Science, 5, 2019, p.e175. [Accessed 20 May 2021].
[29] Default. 2020. Application Quality Defined | |FREE Demo| | Video Explanation. [Online] Available at: <https://www.castsoftware.com/glossary/application- quality> [Accessed 6 October 2020].
[30] A. Adewumi, S. Misra, N. Omoregbe, B. Crawford, and R. Soto, "A systematic literature review of open source software quality assessment models," SpringerPlus, vol. 5, no. 1, 2016. Available: 10.1186/s40064-016- 3612-4. [Accessed 20 Jan 2021].
[31] Arisholm, Erik. "Empirical assessment of the impact of structural properties on the changeability of object-oriented software." Information and software technology ,2006 [Accessed 31 December 2020].
[32] F. Al Obisat, Z. Alhalhouli, T. Alrawashdeh, and T. Alshabatat, "Review of literature on software quality", World of Computer Science and Information Technology Journal (WCSIT), 2018. [Accessed 31December 2020].
[33] Iso.org, 2020. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso- iec:25010:ed-1:v1:en. [Accessed: 31- Dec- 2020].
[34] Estdale, J., & Georgiadou, E. "Applying the ISO/IEC 25010 quality models to software product." In European Conference on Software Process Improvement (pp. 492-503). Springer, Cham ,2018 [Accessed 5 April 2021].
[35] L. Luo, (2001). "Software testing techniques," nstitute for Software Research International Carnegie Mellon University, Pittsburgh, PA, 15232(1-19), 19.I [Accessed 20 Jun 2021].
[36] H. Gamido and M. Gamido, "Comparative review of the features of automated software testing tools", International Journal of Electrical and Computer Engineering (IJECE), vol. 9, no. 5, 2019, p. 4473-4478. Available: 10.11591/ijece. v9i5.pp4473-4478 [Accessed 7 March 2021].
[37] J. Pan, (1999). Software testing. Dependable Embedded Systems, 5, 2006.
[38] A. A. Sawant, P. H. Bari, and P.M. Chawan, "Software testing techniques and strategies," International Journal of Engineering Research and Applications (IJERA), vol. 2, no. 3, 2012, p. 980-986. [Accessed 2 May 2021].
[39] P. Pocatilu, "Automated software testing process," Economy Informatics, 1, 2002, pp. 97-99. [Accessed 20 Sep 2021].
[40] O. Taipale, J. Kasurinen, K. Karhu, and K. Smolander, "Trade-off between automated and manual software testing", International Journal of System Assurance Engineering and Management, vol. 2, no. 2, 2011, pp. 114- 125.Available: 10.1007/s13198-011-0065-6 [Accessed 16 May 2021].
[41] S. Davalbhakta et al., "A systematic review of smartphone applications available for Corona Virus Disease 2019 (COVID19) and the assessment of their quality using the mobile application rating scale (MARS)", Journal of Medical Systems, vol. 44, no. 9, 2020. Available: 10.1007/s10916-020-01633-3. [Accessed 5 Jan 2021]
[42] Ritter, N. (2010). Understanding a widely misunderstood statistic: Cronbach's alpha. Paper presented at Southwestern Educational Research Association (SERA) Conference 2010, New Orleans, LA (ED526237). [Accessed 11 Oct 2022].
[43] Brains, Willnat, Manheim, Rich 2011. Empirical Political Analysis 8th edition. Boston, MA: Longman p. 105. [Accessed 11 Oct 2022].
[44] Ritter, N. (2010). Understanding a widely misunderstood statistic: Cronbach's alpha. Paper presented at the Southwestern Educational Research Association (SERA) Conference 2010, New Orleans, LA (ED526237) [Accessed 11 Oct 2022].

AUTHORS

Asmaa Hassan received a bachelor's degree in Information Systems from King Khalid University in 2018/2019 with an Excellent Category and First Class Honor. I am a teaching assistant at the College of computer science, King Khalid University. I am a master's student at King Saud University, and my research interest includes Software engineering, System analysis, and Software Quality.

Omer Alrwais is an assistant professor at King Saud University in the College of Computer and Information Sciences under the Information Systems department. My entire academic study has been within the information system discipline. Throughout my study I have gained knowledge both in applied information technology solutions as well as core theoretical underpinning of the IS field. My research focuses on how GIS impacts decision-making and how public organizations use GIS to support strategic management. I have developed a new GIS maturity model, which focuses on actual system usage and value gained for local government. Prior to pursuing the academic path, I have worked on SABIC's enterprise system as a system analyst.