**Boardify A Comprehensive Approach to Academic Task Management with Push Notifications and Scheduling Optimization**

**Faiz Syukri Arta**1**, Muhammad Fikry\***2**, Rini Meiyanti** 3

1 Department of Informatics, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, faiz.210170045@mhs.unimal.ac.id

2 Department of Informatics, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, muh.fikry@unimal.ac.id

3 Department of Informatics, Universitas Malikussaleh, Bukit Indah, Lhokseumawe, 24353, Indonesia, rinimeiyanti@unimal.ac.id

**\*** Correspondence: faiz.210170045@mhs.unimal.ac.id

**Abstract.** This study explores the development and implementation of Boardify, an integrated academic task management system designed to enhance the thesis submission process for Informatics students. The system incorporates features such as task management, status tracking, file submission, and seamless communication between students and supervisors, aiming to streamline the entire workflow. By leveraging Firebase Cloud Messaging, Boardify enables real-time push notifications, ensuring timely updates and reducing delays. Furthermore, the implementation of scheduling algorithms optimizes notification timing based on probabilistic factors, enhancing the efficiency of the system. A comparative analysis was conducted between Boardify and similar task management platforms, focusing on aspects such as website load speed, feature functionality, and user experience. The system's performance was further evaluated by measuring the average time taken by supervisors to review student submissions. Results indicate that Boardify significantly improves the efficiency of the thesis submission process, enhancing transparency and facilitating effective communication. The findings underscore Boardify's potential as a powerful tool for academic institutions, offering a promising approach to optimizing educational task management and promoting the advancement of educational technology.

**Keywords:** Boardify. Probability Scheduling, Push Notifications, Informatic, Task Management System

**1. Introduction**

In the rapidly evolving landscape of higher education, effective task management between students and supervisors is essential for academic success. One of the most critical tasks in academic institutions is the timely submission and review of thesis projects, a process often marked by significant communication gaps, delays, and inefficiencies. The traditional approach to thesis submission involves manual interactions between students and their supervisors, typically through emails or in-person meetings. While these methods are familiar, they often result in missed deadlines, lack of transparency in the submission status, and delayed feedback, all of which can hinder the academic progress of students [1]. Therefore, there is an increasing need for systems that automate and streamline academic workflows, facilitating real-time communication and ensuring smooth, timely task execution.

Task management systems have gained prominence in a variety of sectors, including education, by providing a structured approach to handling tasks and responsibilities. These systems can aid students in managing their academic workloads, enabling better prioritization and monitoring of tasks. However, current task management platforms still face limitations when it comes to addressing the unique needs of academic institutions, particularly in the context of thesis submission. Challenges such as the lack of real-time notifications, inefficient scheduling mechanisms [2], and ineffective communication between students and supervisors remain prevalent. To overcome these limitations, an integrated approach that incorporates advanced notification systems and optimized scheduling algorithms [3][4] is essential for improving task management in academia.

The proposed system, Boardify, is an academic task management platform designed to address these challenges specifically within the context of thesis submission for Informatics students. Boardify integrates several core functionalities, including task management, status tracking, file submission, and communication between students and supervisors. Central to the design of Boardify is the use of Firebase Cloud Messaging (FCM) to send real-time push notifications. These notifications are crucial for keeping both students and supervisors updated on important milestones, submission deadlines, and feedback, ensuring that communication flows seamlessly and timely between the two parties.

An innovative aspect of Boardify lies in its use of scheduling algorithms that optimize the timing of notifications based on probabilistic factors [5], ensuring that students receive reminders and updates at the most effective times. This system is designed to reduce delays in communication and enhance task management efficiency by delivering notifications based on the urgency and importance of each task. By incorporating machine learning techniques into the scheduling process [5], Boardify adapts to the unique needs of each user, improving the overall user experience and ensuring that critical updates are not overlooked.

This study aims to evaluate the effectiveness of Boardify in improving the thesis submission process by comparing its performance to other existing task management platforms. Key factors such as website load speed, feature functionality, and user experience are assessed to determine Boardify's advantages and limitations. Additionally, the system's performance is evaluated through the analysis of average review times for thesis submissions by supervisors. The findings of this study contribute to the growing body of research on the application of technology in education, highlighting the potential of integrated systems like Boardify to transform task management and communication within academic settings.

**2. Related Works**

In this study, we use a probability scheduling method to send reminder messages to the lecturer if they have not read it after a certain period. To apply this methodology, there are several steps to calculate the probability, including determining the formula, setting parameters, finding the average reading time of the lecturer, determining the lecturer's reading time, and determining the time to send notifications to their email using this probability method.

*2.1 Firebase Cloud Messaging*

Task management systems have become increasingly popular in both academic and professional settings, providing structured frameworks for organizing and monitoring tasks. In the educational context, several platforms have been developed to assist students with managing academic workloads, including project deadlines, thesis submissions, and communication with supervisors. A study by explored the use of digital platforms for task management in university settings, highlighting the benefits of integrating task tracking, file submission, and real-time communication within a unified system [6]. However, many existing systems still face challenges, such as poor user experience, delayed notifications, and lack of optimization in communication flow, which can hinder their effectiveness.

Firebase Cloud Messaging (FCM), a tool developed by Google, has been explored in various applications for real-time notifications and efficient communication. FCM offers several advantages, including low battery consumption, ease of integration, and support for both web and mobile devices, making it an ideal solution for task management systems like Boardify. The positive impact of FCM in applications where timely alerts and push notifications are crucial, such as in healthcare, e-commerce, and educational platforms [7]. These studies emphasize the importance of delivering notifications in real time, ensuring users stay updated without unnecessary delays, which is a key feature of the Boardify system.

Several studies have focused on improving scheduling algorithms to optimize task management, particularly in environments that require coordination between multiple parties, such as students and supervisors. A dynamic scheduling algorithm was proposed to enhance the timing of notifications based on user preferences and contextual factors [8]. The algorithm was designed to prioritize tasks and adjust notification timing based on the importance of the message, ensuring that urgent messages were sent immediately while less critical ones could be scheduled for later delivery. This approach to scheduling aligns with the design of Boardify, where scheduling algorithms play a crucial role in optimizing notification delivery to users.

Another area of research relevant to Boardify’s development is the evaluation of user experience (UX) in academic platforms. Various academic task management systems, focusing on the effectiveness of their features, ease of use, and overall user satisfaction [9][10]. The study concluded that platforms that integrated real-time notifications and intuitive interfaces were significantly more effective in promoting student engagement and task completion. These findings underscore the importance of user-centered design in the development of academic task management systems and align with Boardify’s emphasis on seamless communication and user-friendly interfaces.

*2.2 Priority Scheduling*

 In the context of academic task management, systems like Boardify have the potential to benefit from probabilistic scheduling algorithms to enhance notification delivery and task prioritization. The effectiveness of probabilistic scheduling in academic platforms, particularly in managing deadlines and notifications [11][12]. By considering variability in task execution times, these systems can predict and optimize when notifications should be sent to users, thus reducing the likelihood of delays and ensuring timely updates. This research supports the implementation of similar scheduling strategies in Boardify, ensuring that students and supervisors receive notifications at the most appropriate times based on probabilistic factors.

Furthermore, Firebase Cloud Messaging (FCM) has been widely studied as an efficient tool for delivering push notifications in various applications. FCM allows for seamless communication between systems and users, ensuring that notifications are delivered reliably and in real time. The benefits of integrating FCM into task management systems, particularly in terms of reducing notification delays and improving user engagement [13]. The study found that real-time push notifications significantly improved the responsiveness of users, making FCM an ideal choice for systems like Boardify, which aims to enhance the thesis submission process by providing timely updates to students and supervisors.

*2.3 Push Notification*

 Push notifications have become an essential feature for improving real-time communication in modern applications, particularly in systems that require timely updates, such as task management platforms. Push notifications enable a direct and efficient communication channel between the application server and the user’s device, eliminating the need for manual requests. Pusher, which utilizes WebSocket technology, is one of the key tools in enabling real-time communication by maintaining an open connection between the client and the server. The effectiveness of WebSockets in real-time applications, showing how it outperforms traditional HTTP-based communication by providing faster and more reliable data transfer [14]. This two-way communication allows for instantaneous updates, a crucial element in task management systems like Boardify, where students and supervisors need to stay informed of the status of submissions and updates without delays.

Moreover, the integration of WebSockets in push notification systems is particularly beneficial in environments where constant updates are necessary. In educational settings, systems that offer real-time communication between students and faculty can significantly enhance the efficiency of academic workflows. The application of real-time notifications in academic management systems, revealing that platforms incorporating push notifications resulted in better time management, improved student engagement, and more timely feedback [8]. These findings support the integration of WebSocket-based push notifications in Boardify, as it enhances communication between students and supervisors, ensuring that important updates, such as feedback and submission deadlines, are delivered instantly.

In addition to the communication framework, the scheduling algorithms used in academic task management systems have garnered attention in recent years. These algorithms help optimize the timing and delivery of notifications to ensure that users are not overwhelmed with irrelevant or excessive alerts. Research with examined the use of probabilistic scheduling models to improve task management in academic systems [15][16][17]. By incorporating factors such as user behavior and task urgency, these models dynamically adjust notification delivery times [18], enhancing the user experience and ensuring that critical messages are prioritized. This approach has been applied in systems like Boardify, where notifications are scheduled based on the probabilistic likelihood of user interaction, ensuring that students receive timely updates without unnecessary interruptions.

**3. Materials & Methods**

*3.1 Probability and Distribution Formula*

We utilize the exponential distribution as it is commonly employed to model the time between events, such as the waiting time until a lecturer reviews an email. The exponential distribution is characterized by its cumulative distribution function (CDF), which describes the probability that an event will occur within a specified time interval.

F(t) = 1-e­- λ

Description

* t : is the time in hours
* λ : is the rate parameter to defined as the reciprocal of the lecturer's average reading time.

( λ $\frac{1}{average reading time)}$)

The exponential distribution enables us to model the probability that the lecturer has not yet read the email at a particular moment, as well as to determine the optimal time to send a notification to ensure timely reading. By utilizing the rate parameter λ, we can efficiently calculate these probabilities using the appropriate formula.

*3.2 Parameters*

Assuming the lecturer reads emails on average once every 3 hours, the rate parameter (λ) can be determined as the reciprocal of this average time. Thus,

λ$=\frac{1}{3}=0.333$

The rate parameter λ is used to quantify the frequency at which an event, such as the lecturer reading an email, is expected to occur within a specific time frame. Given that the lecturer reads emails on average once every 3 hours, λ is calculated as$\frac{1}{3}$, which equals approximately 0.333. This indicates that there is roughly a 0.333 probability per hour that the lecturer will read the email.

3.3 Students *Send Message Data*

**Table 1.** Initial Data for Message Sending

|  |  |  |
| --- | --- | --- |
| **No** | **Students** | **Message send time** |
| 1 | Students 1 | 08:00 |
| 2 | Students 2 | 09:00 |
| 3 | Students 3 | 10:00 |
| 4 | Students 4 | 11:00 |
| 5 | Students 5 | 12:00 |
| 6 | Students 6 | 13:00 |
| 7 | Students 7 | 14:00 |
| 8 | Students 8 | 15:00 |
| 9 | Students 9 | 16:00 |
| 10 | Students 10 | 17:00 |

Table 1 presents the data on message sending times by students. The first student sent their message at 08:00, the second at 09:00, and the third at 10:00, with each subsequent student sending their message at one-hour intervals. These varying times reflect the sequence in which the messages were sent. The next step is to calculate the time it takes for each message to be delivered to the lecturer’s email, which is assumed to occur one hour after the student's message sending time.

3.4 *System Design*

To implement the message delivery calculation program, a structured approach to planning is crucial. The overall system design diagram outlines the key components of the program, which uses probability-based scheduling to manage the lecturer's response time following a student's message. As depicted in Figure 1, the process begins when a student submits a message via the website. If the lecturer does not reply within a specified time frame, the system will use probabilistic calculations to determine the optimal time for sending a reminder message to the lecturer’s email. However, if the lecturer reads the message promptly, the probability calculation is not required. The program will automatically send a reminder if the message remains unread within the designated period. The scheduling method applied in this system is Priority Scheduling, which ensures that student messages are prioritized according to their urgency, and reminders are sent in a timely manner to improve communication efficiency.



**Figure 1.** System Design

The diagram above depicts the process flow for sending messages from students to lecturers. The sequence begins with the student sending a message, followed by the system checking the lecturer’s message reading status. If the message remains unread, the system calculates the probability to determine the optimal time to send an automatic reminder to the lecturer’s email. The process concludes either when the lecturer reads the message or when the reminder is sent, ensuring timely attention to the message.

In summary, this system is designed to facilitate efficient communication between students and lecturers. By automating the reminder process, students no longer need to worry about their messages being ignored, while lecturers can more effectively manage their time and promptly respond to incoming messages.

**4. Result and Discussion**

The implementation of the Priority Scheduling method for calculating the delivery of message reminders has proven highly effective in enhancing communication efficiency. This approach ensures that messages from students, which remain unread by the lecturer, are prioritized based on their urgency. By applying priority scheduling, the system can automatically manage and organize the delivery of reminder messages, ensuring that critical messages are promptly addressed and that lecturers can respond in a timely manner.

As a result, the process of sending and reminding messages becomes more streamlined and efficient, leading to significant improvements in communication within the academic setting. This structured approach not only minimizes the risk of important messages being overlooked but also optimizes the overall communication flow between students and lecturers, ultimately contributing to better academic interaction.



**Figure 2.** Conversation between student and lecturer

 The image above presents an example of a conversation between a student and a lecturer. The process begins when the student sends a message through the system. If the lecturer responds promptly or within a short time frame, there is no need for Priority Scheduling calculations, as the communication is already effective.

However, if the lecturer does not read or reply to the message within the specified period, the system activates Priority Scheduling to calculate the optimal time for sending an automatic reminder to the lecturer’s email. This mechanism ensures that the student’s message is not overlooked, and it receives a response based on its urgency, thereby maintaining efficient communication.



**Figure 3.** Lecturer Reminder

Figure 3 displays the outcome of the reminder system for messages that have not been read by the lecturer. If a student’s message remains unread after a certain duration, the system automatically triggers a reminder notification to the lecturer’s email. This process ensures that messages are not overlooked and that lecturers are reminded to address them in a timely manner.

The reminder system is powered by the Priority Scheduling method, which ensures that messages requiring attention are prioritized for follow-up. This method evaluates the urgency of each message and determines the optimal timing for sending reminders, allowing for more efficient communication. By using this approach, the system ensures that unread messages are flagged for immediate attention.

When a message is not read within the predetermined period, the system acts proactively, sending out automatic reminders to the lecturer. This minimizes delays in response times and enhances the overall communication flow between students and lecturers. The priority system helps avoid potential bottlenecks, ensuring that critical messages are promptly addressed.

Overall, the reminder system’s integration with Priority Scheduling has proven effective in ensuring timely communication. By automating reminder notifications, it reduces the likelihood of overlooked messages and helps maintain an organized, responsive academic communication system.

The following is the calculation of the reminder time for messages that were not read for a certain period by the lecturer, which will be sent to the email.

F(t) = 1-e­- λ

Description

* t : is the time in hours
* λ : is the rate parameter, defined as the perifocal of the average reading time of the lecturer

( λ $\frac{1}{average reading time)}$)

To calculate the probability that the lecturer has not read the email by 8 AM, we assume that the time frame begins at midnight (00:00). This allows us to determine the likelihood of the email remaining unread until the specified time.

P(T>8) =e -0.333:8  = e -2.664 = 0.070

Determine Notification Delivery Time

P(T > t) = e-M  ≤ Pthresehold = 0.05 (5%)

Solving Equations for t

 −0.333t ≤ In (0.05)

t ≥ $\frac{in(0.05)}{-0.333}$

calculate value t:

 t ≥ $\frac{-2.9957}{-0.333}$ = 8.99 ==> Result

The value t≈8.99 means that the lecturer has not read the email for approximately 8 hours and 59 minutes after midnight, which is around 08:59 AM. At this time, the system will send a notification to the lecturer's email.

**Table 2.** Data Result Send Message

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Students** | **Message send time** | **Lecturer** | **Time Reminder To Email** |
| 1 | Students 1 | 08:00 | Lecturer 1 | 08:59 |
| 2 | Students 2 | 09:00 | Lecturer 2 | 09:59 |
| 3 | Students 3 | 10:00 | Lecturer 3 | 10:59 |
| 4 | Students 4 | 11:00 | Lecturer 4 | 11:59 |
| 5 | Students 5 | 12:00 | Lecturer 5 | 12:59 |
| 6 | Students 6 | 13:00 | Lecturer 6 | 13:59 |
| 7 | Students 7 | 14:00 | Lecturer 7 | 14:59 |
| 8 | Students 8 | 15:00 | Lecturer 8 | 15:59 |
| 9 | Students 9 | 16:00 | Lecturer 9 | 16:59 |
| 10 | Students 10 | 17:00 | Lecturer 10 | 17:59 |

The data presented in Table 2 illustrates the automatic reminder system implemented in Boardify, specifically designed to ensure timely communication between students and lecturers. Each student's message is sent at a specific time, and if the lecturer does not respond within one hour, the system triggers an automatic reminder. This approach utilizes Priority Scheduling to prioritize unread messages, ensuring that they receive prompt attention based on their urgency. The reminder time is set to one hour after the initial message is sent, maintaining an efficient communication flow and ensuring that messages are not overlooked by lecturers.

The implementation of Priority Scheduling within the system optimizes the reminder process, ensuring that unread messages are prioritized over others. This scheduling method helps to differentiate between urgent and non-urgent messages, offering the lecturer an effective way to manage communication based on the importance of the task. For instance, a message related to an impending thesis submission might be more time-sensitive than a general query, and the system ensures that such messages are flagged for quicker responses. This reduces the likelihood of critical communications being delayed, promoting timely interactions between students and lecturers.

The system’s performance is highly dependent on the accurate calculation of the time intervals between the message sent by the student and the lecturer’s response. As shown in the data, the time to send a reminder is consistent across the board, with all reminders set to be sent one hour after the initial message. This consistent delay ensures that reminders are timely and that no important message is missed. The one-hour reminder interval was chosen based on practical considerations, as it provides enough time for the lecturer to respond while also ensuring that messages requiring attention are promptly followed up on.

Comparing Boardify to other task management systems highlights its advantage in prioritizing communications based on urgency. While many systems simply notify users of new messages, Boardify’s integration of scheduling algorithms and real-time push notifications ensures that critical messages are addressed before others. This capability is essential in an academic environment, where timely feedback and communication are crucial for the success of students. The priority-based approach also contributes to reducing the administrative burden on lecturers, allowing them to focus on high-priority tasks while ensuring that less urgent matters are still addressed within a reasonable time frame.

The findings from this study emphasize the potential of Boardify to improve academic communication. By automating the reminder process and utilizing Priority Scheduling, the system enhances the overall efficiency of communication between students and lecturers. This approach ensures that messages are promptly addressed, reducing the chances of miscommunication or overlooked tasks. Additionally, the use of real-time push notifications ensures that both students and lecturers are kept informed, fostering a more responsive and transparent communication environment. As a result, Boardify demonstrates considerable potential as an effective tool for optimizing academic task management and supporting better educational outcomes.

**5. Conclusions**

 This study successfully demonstrates the effectiveness of the Priority Scheduling method in an automated message reminder system for communication between students and lecturers. By incorporating probabilistic calculations and priority management, the system ensures that student messages are not overlooked and are responded to based on their urgency. The time-based reminder system has proven to be highly efficient in maintaining smooth communication within the academic environment, enhancing the responsiveness of both students and lecturers.

The results indicate that the system performs effectively in the scenarios tested, ensuring that unread messages receive timely attention. However, external factors such as unscheduled lecturer responses and competing commitments may impact the reminder system’s overall effectiveness. To further optimize the system, future developments should aim to incorporate adaptive features that respond to these external variables, ensuring that important communications are not missed, even in dynamic academic settings.

The integration of Priority Scheduling into the message delivery system provides a more intelligent solution for managing communications between students and lecturers, making the process more efficient and structured. This advancement contributes to the academic workflow by reducing the likelihood of important messages being overlooked and enhancing the overall communication experience. The system’s ability to automatically manage reminders based on message urgency streamlines interactions and supports a more organized approach to academic communication.

Future research should focus on refining the algorithm’s performance, including large-scale field trials to validate the system's effectiveness across diverse academic scenarios. Additionally, exploring the potential for expanding the system to include other forms of academic communication, such as assignment submissions and grading feedback, would further enhance its utility. By optimizing the communication processes between students and lecturers, Boardify has the potential to significantly contribute to improved academic interactions and foster more effective educational environments.

# 6. References

[1] T. Liu, W. Tian, and W. Ning, “Sequential probability ratio test for zero inflation in counting data,” *Commun Stat Simul Comput*, vol. 52, no. 4, pp. 1344–1360, 2023,

[2] G. Xiao, M. Dong, J. Li, and L. Sun, “Scheduling routine and call-in clinical appointments with revisits,” *Int J Prod Res*, vol. 55, no. 6, pp. 1767–1779, Mar. 2017,

[3] U. Klanšek and M. Pšunder, “Troškovna optimizacija terminskih planova za vodenje projekata,” *Ekonomska Istrazivanja*, vol. 23, no. 4, pp. 22–36, 2010, doi: 10.1080/1331677X.2010.11517431.

[4] L. Hu, L. Lin, J. Zhao, X. Che, and X. Wei, “Optimisation to the execution performance of grid job based on distributed file system,” *International Journal of Parallel, Emergent and Distributed Systems*, vol. 27, no. 2, pp. 109–121, Apr. 2012, doi: 10.1080/17445760.2011.574629.

[5] D. Carreras-García, D. Delgado-Gómez, E. Baca-García, and A. Artés-Rodriguez, “A Probabilistic Patient Scheduling Model with Time Variable Slots,” *Comput Math Methods Med*, vol. 2020, 2020, doi: 10.1155/2020/9727096.

 [6] Williams, Bryan, et al. "2018 ESC/ESH Guidelines for the management of arterial hypertension: The Task Force for the management of arterial hypertension of the European Society of Cardiology (ESC) and the European Society of Hypertension (ESH)." European heart journal 39.33 (2018): 3021-3104.

[7] Gray, S. R. J., et al. "Caught by the fuzz: using FCM to prevent coastal adaptation stakeholders from fleeing the scene." Marine Policy 109 (2019): 103688.

[8] Zhang, Fangfang, et al. "Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling." ieee transactions on cybernetics 51.4 (2020): 1797-1811.

[9] Maslov, Ilia, Shahrokh Nikou, and Preben Hansen. "Exploring user experience of learning management system." The International Journal of Information and Learning Technology 38.4 (2021): 344-363.

[10] Anderson, Christoph, et al. "A survey of attention management systems in ubiquitous computing environments." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2.2 (2018): 1-27.

[11] Liu, Weifeng, et al. "Optimal stochastic scheduling of hydropower-based compensation for combined wind and photovoltaic power outputs." Applied Energy 276 (2020): 115501.

[12] Fikry, Muhammad, and Sozo Inoue. "Optimizing Forecasted Activity Notifications with Reinforcement Learning." Sensors 23.14 (2023): 6510.

[13] Shahvi, S., et al. "A Fuzzy Cognitive Map method for integrated and participatory water governance and indicators affecting drinking water supplies." Science of the Total Environment 750 (2021): 142193.

[14] Murley, Paul, et al. "Websocket adoption and the landscape of the real-time web." Proceedings of the Web Conference 2021. 2021.

[15] Fikry, Muhammad, Nattaya Mairittha, and Sozo Inoue. "Modelling Reminder System for Dementia by Reinforcement Learning." Sensor-and Video-Based Activity and Behavior Computing: Proceedings of 3rd International Conference on Activity and Behavior Computing (ABC 2021). Singapore: Springer Nature Singapore, 2022.

[16] Fikry, Muhammad. "Requirements analysis for reminder system in daily activity recognition dementia: Phd forum abstract." Proceedings of the 18th Conference on Embedded Networked Sensor Systems. 2020.

[17] Anderson, David R., et al. "Rationing scarce healthcare capacity: A study of the ventilator allocation guidelines during the COVID‐19 pandemic." Production and Operations Management (2023).

[18] Fikry, Muhammad, and Muhammad Iqbal. "Optimizing Multi-Time Notifications Using Q-Learning." Proceedings of Malikussaleh International Conference on Multidisciplinary Studies (MICoMS). Vol. 3. 2022.