Jurnal Elektronika dan Telekomunikasi (JET), Vol. 24, No. 2, December 2024, pp. 129-136
Accredited by RISTEKDIKTI, Decree No: **158/E/KPT/2021**
doi: 10.55981/jet.674

Development of the Android Application for the Landslide Disaster Mitigation Real-time System Based on Firebase Server and OneSignal

Aif Umar Nawawi[a], Arjuni Budi Pantjawati[b,*], Aip Saripudin[a], Nurul Fahmi Arief Hakim[c], Nurhidayatulloh[d], Novia Karostiani[b], Bagaskara Anandayutya[a], Alvin Dzaki Pratama Darmawan[b]

[a]Electrical Engineering Study Program
Universitas Pendidikan Indonesia
Jalan Dr Setiabudhi No 229
Bandung, Indonesia
[b]Electrical Engineering Education Study Program
Universitas Pendidikan Indonesia
Jalan Dr Setiabudhi No 229
Bandung, Indonesia
[c]Industrial Automation and Robotics Engineering Education Study Program
Universitas Pendidikan Indonesia
Jalan Dr Setiabudhi No 229
Bandung, Indonesia
[d]Multimedia Education Study Program
Universitas Pendidikan Indonesia
Jl. Pendidikan No.15, Cibiru Wetan
Bandung, Indonesia

Abstract

This article presents the development of SIDASIBELO (Landslide Information and Mitigation System), an Android-based application for real-time landslide monitoring and early warning. This system integrates data from several sensors that measure soil moisture, rainfall, slope angle, and ground vibrations at the monitored locations. The Firebase Real-time Database stores and transmits sensor data, while Firebase Cloud Messaging and OneSignal enable push notifications to alert users about potential landslide risks. There are four output status levels: SAFE, ALERT, CAUTION, and WARNING, which are determined on the basis of a comprehensive analysis of the monitored parameters. Mitigation strategies include monitoring of parameters in real time, early warnings, evacuation guidance, and user education on landslide preparedness. Testing on several Android devices demonstrated high compatibility, with an average successful server connection rate of 92.86%. The latency tests indicated an average response time of 894 ms from the input device to the database, 1.29 ms from the database to the Android device, and 2725 ms for push notifications. The total daily bandwidth usage of the system for real-time data transmission is 27,648 MB, indicating high efficiency without overburdening the server's performance. While the app operates efficiently under ideal conditions, unstable network connections can lead to data retrieval failures or, in the worst cases, to app malfunctions. This remains a key challenge for our system. This system aims to increase awareness of landslide risks and allow rapid evacuation, which could potentially reduce the negative impacts of landslides in vulnerable areas.

Keywords: Landslide early warning system, Firebase, mobile application, OneSignal.

I. INTRODUCTION

Landslides are a frequent disaster in Asia, particularly in Indonesia [1]. Numerous efforts have been made by local governments, private sectors and humanitarian organizations to mitigate the adverse effects of landslides. In West Java, landslides are among the most common natural disasters. Data from the official Jabar Open Data website indicate that 609 landslide incidents were recorded in 2019. This high incidence is largely due to suboptimal systems, particularly the reactive rather than preventive approach to disaster management [2], [3].

Advancements in smartphone technology, offering real-time connectivity, are leveraged in this project to mitigate landslides [4], [5]. Consequently, an innovative Android application, SIDASIBELO (Sistem Informasi dan Mitigasi Bencana Longsor), which translates to the Information and Landslide Disaster Mitigation System, is being developed.

This study offers a solution to transform the previously reactive system into a preventive one, which holds significant benefits for the surrounding community. Previous studies have explored different approaches and systems for landslide mitigation. Examples include using XAMPP as an application

* Corresponding Author.
Email: arjunib@upi.edu
Received: September 1, 2024 ; Revised: November 8, 2024
Accepted: December 10, 2024 ; Published: December 31, 2024

server, issuing alerts via SMS at three danger levels through GSM, deploying SQL databases on Java-based web application servers, and using microservices to deliver outputs across Android, web browsers, and iOS applications [6]–[9].

Our system will provide real-time monitoring services through sensors that measure humidity, ground vibration, ground slope, and rainfall sent from the monitoring location through LoRa (long-range) radio transmission up to 15 km so that it can cover areas with minimal cellular signal. LoRa data is sent to the server to provide warnings to the surrounding community through the Landslide Early Warning System (LEWS) [10]. The most crucial feature of our app is push notifications, which will be particularly beneficial to users, providing them with early warnings to prepare for potential landslides. These notifications will be sent as broadcasts, including situations where the app is not open or used [11] - [13].

The platforms used as servers for this project include Firebase Cloud Messaging, Firebase Real-Time Database, and OneSignal. Firebase Real-time Database, developed by Google, is an online platform that offers free data storage in JSON format, with rapid response times [14]. Similarly, Firebase Cloud Messaging, also a Google product, focuses on sending real-time push notifications to Android applications on users' devices at no cost [15], [16]. Additionally, OneSignal is integrated as the primary intermediary for push notifications, to support Firebase Cloud Messaging, due to its easy integration with Android applications [17]. This system is expected to set a new standard in disaster mitigation, specifically designed for landslide-prone areas, to effectively minimize negative impacts and losses. By shifting from reactive to preventive methodology, the system empowers communities to prepare and respond more effectively to potential hazards from landslides.

II. METHODS

In general, to ensure that this system runs effectively, at least three main components are required: the data input section, the server or database, and the Android application as the output.

A. Data Input

The system is the transmitter of all the parameters of the landslide disaster, which will be displayed in the SIDASIBELO application. The flow of these input data is illustrated in Figure 1. Data transmission is managed through two distinct protocols: a real-time data protocol and a trigger protocol that initiates push notifications upon detection of an emergency. The system uses Python scripts to process and transmit data in JSON format.

1) Real-time data protocol

In this protocol, the transmitted data includes soil moisture sensor readings, rainfall amounts, land slope angles, and vibration magnitudes. Additional parameters consist of the monitoring location's name, details of the surrounding environment, coordinates (latitude and longitude), evacuation points, and disaster levels. These data are transmitted every 5 seconds to ensure that users

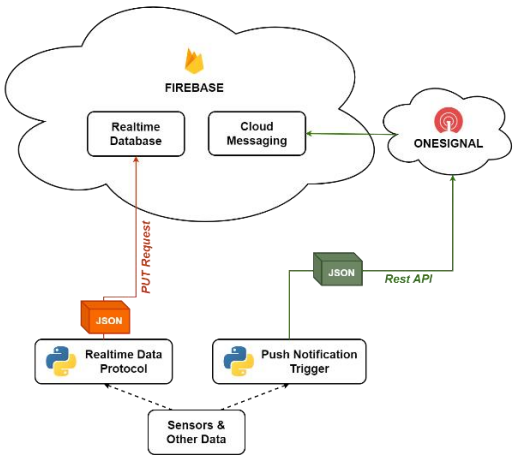

Figure 1. Data input flow chart.

receive up-to-date information via the PUT Request protocol in Python to the Firebase real-time database. The system is optimized for rapid data transmission, requiring a low-latency Internet connection to ensure real-time processing. To minimize bandwidth usage, the data size is kept as small as possible.

2) Protocol Trigger Push Notification

In this system design, there are four output status levels: SAFE, ALERT, CAUTION, and WARNING. This protocol is activated only when sensor parameters reach critical thresholds. Upon determining this status, the system triggers a signal to the Firebase Cloud Messaging server, facilitated by the OneSignal platform, to obtain the REST API (Representational State Transfer Application Programming Interface) and APP ID from the connected application server. This setup ensures that users receive real-time notifications on their devices without having to request data from the server [18], [19]. This protocol is crucial for disaster mitigation systems, particularly for landslides, as it allows users to receive early warnings and prepare for evacuation before a disaster strikes. This approach aims to reduce the negative impact, minimize losses, and decrease the number of casualties resulting from landslides.

B. Database

A high-performance server or database with extremely fast response times is essential for managing this system. We use Firebase Database and Firebase Cloud Messaging because they not only offer rapid response times, but are also easy to use and flexible enough to integrate with various systems or application designs [20]. With Firebase Database, we can focus on system and application development without having to manage complex server-side data structures [21]. Firebase Cloud Messaging acts as the conduit for quickly delivering notifications to users.

The data transfer process is illustrated in Figure 2. The OneSignal platform is also used to send triggers to Firebase Cloud Messaging via the REST API, a protocol to transmit and receive web-based data using the HTTP protocol and application ID in a straightforward manner. This setup simplifies the data transmission pathway from the monitoring points to the user's mobile application [22], [23]. All data sent and

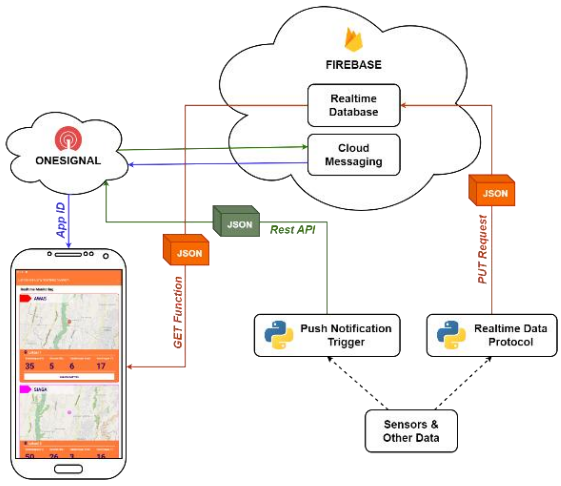

Figure 2. Flow diagram when the user launches the application.

received is packaged in JSON format, which conserves bandwidth and facilitates the data extraction process. The server or database operates in two primary functions: temporarily storing all real-time parameters and sending push notifications.

The system relies on Firebase Realtime Database, Firebase Cloud Messaging, and OneSignal for data management and notification delivery. Firebase Realtime Database acts as the main database to store and retrieve the overall data of sensor parameters from each monitoring point; Firebase Cloud Messaging is in charge of providing broadcast notification messages to application users; and OneSignal is an intermediary bridge between the main system (in this case the monitoring gateway) and Firebase Cloud Messaging.

1) Real-time Data Storage

In this section, the server functions as a temporary data repository, facilitating the connection between the sender and receiver of the data. Data are transmitted via the PUT request protocol in JSON format directly from the data sending device. To retrieve the latest data for display in the SIDASIBELO application, a simple GET request can be made using the URL provided by Firebase. Since the Firebase Real-Time Database server is used, the read and write rules must be set to "true" to ensure that the data sent are fully received and stored on the server. The Firebase real-time database has a storage limit of 1 GB, supports a maximum of 100 connections, and has a daily bandwidth limit of 350 MB. These limits are considered sufficient for the project's development phase, and if additional resources are needed, a subscription upgrade through Google is available.

2) Push Notification Section

As previously explained, the platforms used are Firebase Cloud Messaging and OneSignal. These platforms enable seamless notification delivery between the server and users. Additionally, OneSignal offers user management features, allowing the monitoring of notification functionality and the identification of any issues.

This section operates only upon receiving a trigger command from the push notification protocol via the REST API bridge provided by OneSignal. The data received is in JSON format and includes the APP ID, user segments, notification message, title, header image, and buttons with their corresponding commands. After validation, the command is sent to Firebase Cloud Messaging to quickly dispatch push notifications to each user segment and its associated APP ID. Although these events are processed rapidly, delivery can be influenced by factors such as signal quality and the latency of each user's smartphone.

C. SIDASIBELO Application

This section is the core and the most emphasized part of this article, where it contains the back-end and front-end that interact directly with users or the research target. All technical aspects of creating this application were carried out through the Kodular platform, which is a web-based service provider for Android applications, due to its ease of use and attractive features. Similarly to the previous sections, this application generally serves two important functions: the real-time monitoring of landslide parameters and the handling of received push notifications.

1) Real-time Monitoring

In the development of this application, parameters from three monitoring points will be displayed, each point having data consisting of main parameters and other parameters as previously explained in the real-time data protocol section. All the necessary parameters are obtained from the application's requests to the Firebase Real-Time Database server using the GET function, which is executed every 5 seconds and then displayed on the application's GUI. Not only that, when the application is first opened after installation, it will register with OneSignal using the available APP ID so that each user receives his own unique ID [24]. With this user ID, users can receive push notifications from the broadcast provided by the server when there is a point that is experiencing an emergency situation. Further explained in Figure 3.

2) Push Notification Handling

When push notifications are sent from the server, several conditions may occur: the push notification does not reach the user, the notification arrives but is ignored,

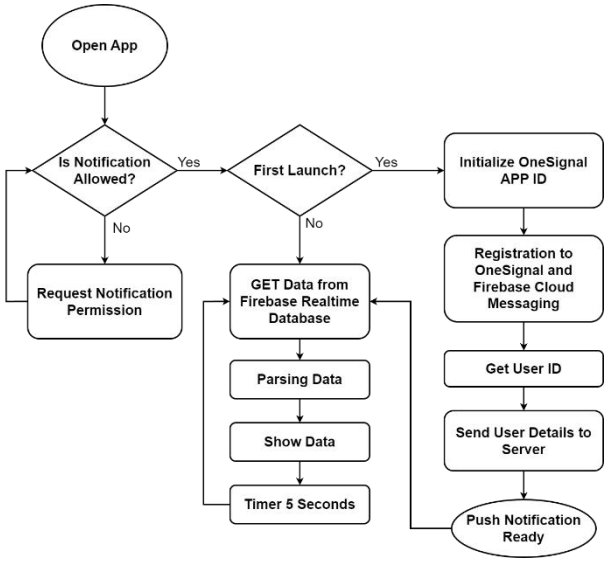

Figure 3. Real-time data and push notification register scheme.

and the notification arrives and receives a response from the user. Users can respond to the notification by accessing the detail view or the evacuation route via the provided shortcut. This push notification handling diagram is illustrated in Figure 4.

III. Result and Implementation

A. SIDASIBELO Application

The SIDASIBELO application offers features that are easily accessible to users. These features include the display of disaster parameters from four sensors along with the latest status, a GUI (Graphical User Interface) in the form of a map showing the distribution of each monitoring point, guidance routes to evacuation points, explanations of each status or level, education on landslide evacuation stages, and mitigation in the form of danger warnings through push notifications. The front-end display consisting of UI (User Interface) and UX (User Experience), along with the back-end and other features, are explained in the following points.

1) *Application Icon*

An important consideration in creating an Android application is, of course, the selection of the icon, as this greatly impacts the users. An attractive icon is highly prioritized, which makes users eager to try installing it. The icon we designed is a combination of several components as follows:

a) *Mountain*: It is described as a large dark green triangular shape. The mountain represents a mountainous area that is prone to landslides due to its slope and soil conditions.
b) *Raindrop*s: Graphs are shown as vertical lines in orange, black, and white. Heavy rain is a key trigger for landslides, as it can cause erosion and soil saturation.
c) *Slope*: Depicted as an orange-slanted area. Steep slopes increase the risk of landslides due to gravity and soil instability.
d) *Damaged houses*: This is indicated by several small houses leaning on the orange slope, illustrating the impact of landslides on settlements and infrastructure.
e) *Black cloud*: The shape appears as a gray shape at the top of the image. Dark clouds indicate bad weather that can trigger rainfall.

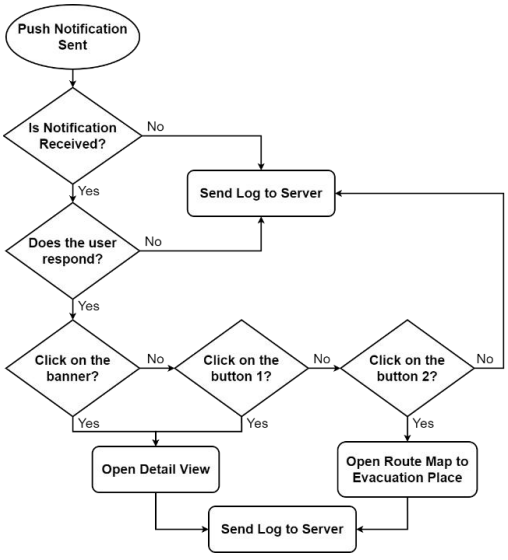

Figure 4. Possible events during push notification.

2) *Monitoring Tab*

The first part that appears when the application is opened is the loading screen, which ensures that the connection to the server is running efficiently. After that, the application will display the next view, shown in the left part of Figure 5, which is the monitoring tab. This tab includes the condition level of each point, the four disaster parameters, and a small map showing the monitoring location.

3) *Evacuation Route Page*

In the section that directs users to the gathering or evacuation point, the application will analyze the best route from the user's location to the predetermined evacuation point. The page switches directly through Google Maps, as shown in the right part of Figure 5. This feature aims to provide accurate guidance, ensuring that users are directed to personnel at the evacuation site and can receive the necessary assistance.

4) *Guidance Page*

Another feature of the app is the counseling section, where users, who are members of the general public, can learn what actions to take when symptoms or critical events of landslides occur. This counseling tab explains the steps to follow in case of a dangerous situation. Information is presented in the form of concise bullet points for easy reading, as shown in the left part of Figure 6. Additionally, there is an explanation of each parameter and disaster level, ensuring that users understand the meaning of these elements.

5) *Push Notification*

To fulfill the application design in the form of this mitigation point, we use push notifications as an early warning for landslide emergencies before any undesirable events occur. Users can prepare for any possible situations that may occur and inform others in

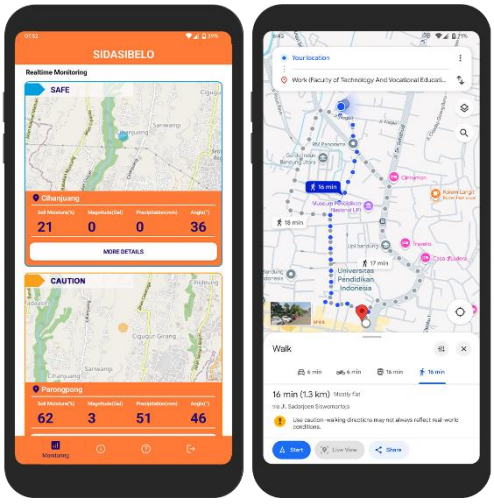

Figure 5. The Monitoring tab (left) and the evacuation route (right).

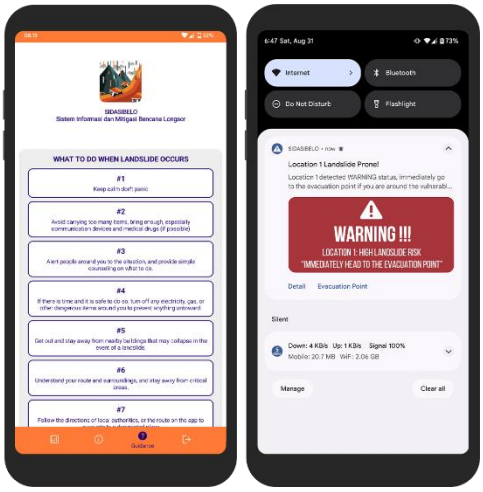

Figure 6. Guidance page (left) and notification GUI (right).

their environment so that they receive the same information. This push notification depends on the network connection of the user's smartphone connected to the OneSignal and Firebase Cloud Messaging servers. Notifications are sent to users' devices using registered IDs [25].

This push notification displays five sections: the title section, message, image, detail button, and gathering or evacuation button. The detail button will open the map along with the details of the readable parameters, while the evacuation point button will create a route for the user to a predetermined location on the full-screen map display. These buttons are designed to facilitate user navigation without the need to manually open the application; the view can be seen in Figure 6 on the right side.

B. Application Compatibility

Based on tests involving the installation of application packages on three different types of Android devices, everything ran successfully without any issues, as shown in Table 1. This success is due to the devices meeting the minimum SDK level requirement, which is Android 5.0 (Lollipop), released in early November 2014. This minimum criterion is sufficient to cover many Android devices currently available on the market [26]. Application compatibility is crucial to ensure that all users can use the application along with its features.

C. User Registration to OneSignal

The results of the experiment show that all users can successfully connect to the OneSignal server using the APP ID provided. This testing is necessary to ensure that all users can receive push notifications, as notifications will only reach registered users. The registration process can be tracked through the OneSignal management dashboard, which records details such as the smartphone model, subscription ID, OneSignal ID, access session, and other user information.

TABLE 1
APP COMPATIBILITY

Phone Series	Android Version	Compatibility	
		Executable Function	Responsive UI
Google Pixel 3A	12	✓	✓
Samsung Galaxy M22	13	✓	✓
Xiaomi Redmi 9C	10	✓	✓

D. Connection Test to Firebase Real-Time Database

Differences in connection types, device specifications, and other factors can cause connection errors. The undesirable consequence of such errors is that users may not receive the latest data, or in the worst-case scenario, the application could not function. Table 2 displays the test data for three smartphones that handle 100 connections to the Firebase real-time database every 5 seconds continuously.

E. Latency Test

In this latency test, the process is divided into several parts: the latency from the input device to the Firebase Real-time Database, the latency of data retrieval from the Firebase Real-Time Database to the Android device, and the latency from the input device to the Android push notification. The test results are displayed in Table 3, with each part performed over 100 repetitions.

F. Bandwidth Calculation

Based on measurements using a Python script, the average size of the generated JSON file for real-time data is 1.6 kB. With this size, the estimated daily data bandwidth usage to send real-time data can be calculated using formulas (1)-(3). Meanwhile, for a single push notification data transmission, the size of the data sent is 555 B. This relatively small data bandwidth usage is very suitable for real-time transmission and does not burden server performance, allowing efficient implementation without disrupting system performance.

$$Total\ BW = Size \times \frac{24\ h}{5\ s} \quad (1)$$

$$Total\ BW = 1{,}6 \times \frac{86400\ s}{5\ s} \quad (2)$$

$$Total\ BW\ =\ 27648\ kB\ =\ 27{,}648\ MB \quad (3)$$

G. System Comparison

This study shows significant advantages over previous studies, especially in terms of latency and

TABLE 2
SERVER CONNECTION TEST

Phone Series	Execution		Percentage
	Success	Fail	
Google Pixel 3A	86	14	86%
Samsung Galaxy M22	98	2	98%
Xiaomi Redmi 9C	96	4	96%
Total	**280**	**20**	**92.86%**

TABLE 3
LATENCY TEST

Test Type		Latency (ms)		
From	To	Min	Max	Avg
Input Device	Firebase Real-time Database	808	1279	894
Firebase Real-Time Database	Android device	1	2	1.29
Input device (Trigger Push Notification)	Android Device (Push Notification)	2325	4646	2725

bandwidth consumption. For example, the results of this study are better than Kabakus [27], who recorded a latency of 3.8 ms; Venica et al. [28], with a latency of 1.5 s; and Sofwan et al. [29], with a latency of 2 s. In terms of bandwidth consumption, the system developed in this study shows high efficiency with an average data requirement of only 1.6 kB for each transmission. This value is lower compared to a similar test conducted by Younis et al. [30], where they stored data from four sensor parameters on 15 smartphones in 10 repetitions, resulting in total data of 640.43 kB or an average of 4.27 kB per transmission. In addition, compared to similar studies that also focus on landslide mitigation, such as those conducted by Kapoor et al. [6], Sejera et al. [7], Silva [8], and Bai et al. [9], the results of this study provide a clear comparison, as summarized in Table 4.

H. Analysis and Discussion

1) Firebase

On the Firebase Real-time Database dashboard, the administrator can view the data stored in real time, as shown in Figure 7. The number of connections, storage usage, server bandwidth, server load, and the number of requests and transmissions are also reported in other dashboard sections.

2) OneSignal

OneSignal records the history of all ongoing, sent, and failed notifications and allows manual notifications to be sent through the provided dashboard. In the OneSignal dashboard, administrators can view user interaction activities, such as the CTR (Click Through Rate) metric, as shown in Figure 8. In addition, user data registered on the server, including session time, ID, device model, Android version, and time zone, is reported on the dashboard.

TABLE 4
SYSTEM COMPARISON

Reference	Reference System	Our System
Kapoor et al. [6]	Local XAMPP Server	Global Firebase Server
Sejera et al. [7]	SMS/GSM Alerts	FCM Alerts
Silva [8]	Web-App-based	Android App-Based
Bai et al. [9]	Microservices	Simple Database
Kabakus [27]	3.8 ms latency	2725 ms Latency
Venica et al. [28]	1.5 s latency	2725 ms Latency
Sofwan et al. [29]	2 s latency	2725 ms latency
Younis et al. [30]	4.27 kB bandwidth	1.6 kB Bandwidth

Figure 7. Firebase real-time database dashboard view.

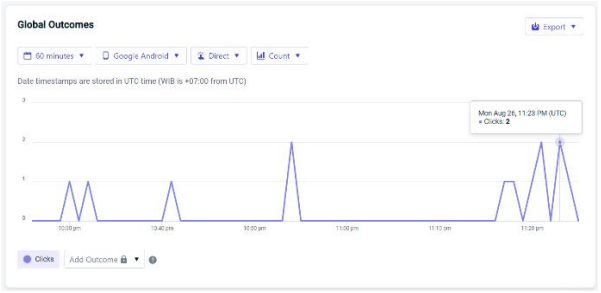

Figure 8. CTR on OneSignal dashboard.

The development and testing of the SIDASIBELO application reveal the implementation of a real-time landslide early warning system on a mobile platform. High compatibility across various Android devices (an average success rate of 92.86%) indicates that this application can reach a wide user base, which is crucial for the effective dissemination of landslide alerts. Your latency shows relatively fast data transmission from the sensor to the Firebase Real-Time Database (an average of 894 ms), quick data retrieval by Android devices (an average of 1.29 ms) and 2725 ms for push notifications. This low latency is crucial to providing users with up-to-date information about landslide risks.

The integration of various data sources (soil moisture, rainfall, slope angle, and ground vibrations) provides a comprehensive approach to landslide risk assessment. The four-tier monitoring system improves alert precision by classifying risks based on sensor readings. The addition of educational content and evacuation guidelines to the application addresses a crucial aspect of user preparation.

The use of the Firebase and OneSignal platforms, with low bandwidth usage (around 27,648 MB per day), facilitates scalable and real-time data management. However, one of the limitations of the current system is its reliance on cellular network connectivity for data transmission and push notifications. In landslide-prone areas with poor network coverage, this can limit the effectiveness of the system. Based on the test results, the technology-based system developed in this research shows optimal performance in supporting the mitigation of landslides. The system is expected to be an effective solution in tackling various problems related to landslides, especially by providing early warning and real-time information that can help the community take preventive measures before a disaster occurs. Therefore, the mitigation approach proposed in this research not only relies on spontaneous response after a disaster, but also encourages planned preventive efforts to reduce the negative impact and risk of casualties due to landslides.

IV. CONCLUSIONS

The Android application SIDASIBELO represents a significant step forward in leveraging mobile technology for landslide risk mitigation. By providing real-time monitoring, early warnings, and educational content, this system has the potential to enhance community awareness and preparedness for landslide hazards. The high compatibility across various devices and the low latency in data transmission indicate the feasibility of using smartphone applications to reduce

disaster risk. Challenges remain in maintaining consistent performance across devices and network conditions. High latency or poor connectivity can hinder data transmission and, in some cases, lead to application malfunctions. Another limitation is the installation process on Android devices with lower specifications; the app may lag because the application is too heavy to load the map display.

Future work should focus on optimizing the delivery of push notifications, expanding data transfer capabilities, validating the accuracy of the system's predictions against real-world landslide events, and improving the reliability of the application to prevent errors, especially on low-specification devices. As climate change potentially increases the frequency and severity of landslides in many areas, systems such as SIDASIBELO will play an increasingly important role in protecting vulnerable communities. The continuous development and implementation of mobile-based early warning systems such as this, combined with education and community involvement, offer a promising way to reduce casualties and economic losses due to landslides.

DECLARATIONS

Conflict of Interest

The authors have declared that there are no competing interests.

CRediT Authorship Contribution

Arjuni Budi Pantjawati and Aip Saripudin: Conceptualization, Nurhidayatulloh and Novia Karostiani: Methodology, Aif Umar Nawawi: Software, Validation, and Writing-Original draft preparation; Bagaskara Anandayutya and Alvin Dzaki Pratama Darmawan: Data curation; Nurul Fahmi Arief Hakim: Writing-Reviewing and Editing.

Funding

This work was supported by the UPI research program (Penelitian Penguatan Kelompok Bidang Keilmuan) under contract number 483/UN40.LP/PT.01.03/2024.

Acknowledgment

The authors thank the research group of Universitas Pendidikan Indonesia in the Landslide Disaster Mitigation project for supporting this research.

REFERENCES

[1] R. B. Bhardwaj, “Landslide Detection System: Based on IOT,” *Int. J. Sci. Res. Dev.*, 2021, vol. 9, no. 1, pp. 54–59.

[2] M. Herviany, S. P. Delima, T. Nurhidayah, and K. Kasini, “Perbandingan Algoritma k-Means dan k-Medoids Untuk Pengelompokkan Daerah Rawan Tanah Longsor pada Provinsi Jawa Barat: Comparison of k-Means and k-Medoids Algorithms For Grouping Landslide Prone Areas In West Java Province,” *MALCOM: Indones. J. Mach Learn. Comput. Sci.*, 2021, vol. 1, no. 1, pp. 34–40.

[3] M. R. Dzulkarnain, A. Fariza and A. Basofi, "Mobile Based Of Mitigation and Emergency System For Landslide in Ponorogo, East Java, Indonesia," 2016 International Electronics Symposium (IES), Denpasar, Indonesia, 2016, pp. 154-159, doi: 10.1109/ELECSYM.2016.7860993.

[4] M. Sarwar and T. R. Soomro, “Impact of Smartphone’s on Society,” *Europ. J. Sci. Res*, 2013, vol. 98, no. 2, pp. 216–226.

[5] R. K. Swamy, “Mobiles Have Changed the Way We Communicate,” *Int. J. Engl. Res.*, 2020, vol. 6, no. 6, pp. 40–43.

[6] S. Kapoor, H. Pahuja and B. Singh, "Real time monitoring & alert system for landslide," 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Greater Noida, India, 2016, pp. 584-589, doi: 10.1109/IC3I.2016.7918030.

[7] M. M. Sejera, A. H. Ballado, B. N. H. Fernando, M. F. I. A. Montemayor and A. V. D. Niebres, "Mobile App-Based Early Warning System for Landslides Using Land Monitoring Through GSM," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 90-93, doi: 10.1109/TENSYMP50017.2020.9230936.

[8] C. R. Silva, “Addressing Landslide Issue in Sri Lanka Using a Web-Based Mobile Application,” M.S. thesis, Dept. Comput. Sci., Auckland Univ. Technol., Auckland, New Zealand, 2020.

[9] D. Bai, J. Tang, G. Lu, Z. Zhu, T. Liu, and J. Fang, “The Design and Application of Landslide Monitoring and Early Warning System Based on Microservice Architecture,” Geomatics, Natural Hazards and Risk, vol. 11, pp. 928–948, Jan. 2020, doi: 10.1080/19475705.2020.1766580.

[10] A. Wicki, P. Lehmann, C. Hauck, S. I. Seneviratne, P. Waldner, and M. Stähli, “Assessing The Potential of Soil Moisture Measurements for Regional Landslide Early Warning,” Landslides, vol. 17, no. 8, pp. 1881–1896, Aug. 2020, doi: 10.1007/s10346-020-01400-y.

[11] R. Maulidi, B. Kristanto, and Y. Listio, “Earthquake Information Push Notification System in Android Application using Google Firebase,” *Int. J. Inf. Syst. Comp. Sci*, vol. 4, pp. 98–105, Aug. 2020, doi: 10.56327/ijiscs.v4i2.898.

[12] S. Maryam and A. Purwono, “Android Application Development for Push Notification Feature for Indonesian Space Weather Service Based on Google Cloud Messaging,” in *J. Phys. Conf. Ser.*, 2022, vol. 2214, no. 1, p. 012031.

[13] R. el Stohy, N. Ghetany, and H. El-Ghareeb, “A Proposed System for Push Messaging on Android,” *Int. J. Interact. Mob. Technol.*, vol. 10, no. 3, pp. 29–35, Jul. 2016, doi: 10.3991/ijim.v10i3.5567

[14] M. Ohyver, J. V. Moniaga, I. Sungkawa, B. E. Subagyo, and I. A. Chandra, “The Comparison Firebase Realtime Database and MySQL Database Performance using Wilcoxon Signed-Rank Test,” *Procedia Comput. Sci.*, vol. 157, pp. 396–405, Jan. 2019, doi: 10.1016/j.procs.2019.08.231.

[15] M. A. Mokar, S. O. Fageeri, and S. E. Fattoh, “Using Firebase Cloud Messaging to Control Mobile Applications,” in *2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, Sep. 2019, pp. 1–5. doi: 10.1109/ICCCEEE46830.2019.9071008.

[16] P. Chougale, V. Yadav, A. Gaikwad, and B. Vidyapeeth, “Firebase-Overview and Usage,” *Int. Res. J. Modern. Eng. Technol. Sci.*, 2021, vol. 3, no. 12, pp. 1178–1183.

[17] R. K. Sungkur, Y. Gangabaksh, and N. Rutah, “Cloud-based Cross-Platform Push Notification System for more Informed Learners,” in *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies (EmergiTech)*, 2016, pp. 229–234, doi: 10.1109/EmergiTech.2016.7737344.

[18] A. B. Gunawan, S. Hansun, and M. B. Kristanda, “Nolong. in: an Android Based Incident Notification Application With Push Notification Technology.,” *Int. J. Elect. Comp. Eng. (2088-8708)*, 2019, vol. 9, no. 1, DOI: http://doi.org/10.11591/ijece.v9i1.pp485-495

[19] M. Ahmadi, B. Biggio, S. Arzt, D. Ariu, and G. Giacinto, "Detecting misuse of Google Cloud Messaging in Android badware," in Proceedings of the 6th Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '16), Vienna, Austria, 2016, pp. 103–112, doi: 10.1145/2994459.2994469.

[20] D. Firdaus, B. Priambodo, and Y. Jumaryadi, “Implementation of push notification for business incubator,” *Int. J. Online Biomed. Eng*, Oct. 2019, vol. 15, no. 14, pp. 42–53, doi: 10.3991/ijoe.v15i14.11357.

[21] N. Chatterjee, S. Chakraborty, A. Decosta, and A. Nath, “Real-Time Communication Application Based on Android Using Google Firebase,” *Int. J. Adv. Res. Comput. Sci. Manag. Stud.*, 2018, vol. 6, no. 4.

[22] I. O. Suzanti, N. Fitriani, A. Jauhari, and A. Khozaimi, “REST API Implementation on Android Based Monitoring Application,” *J. Phys. Conf. Ser.*, Jul. 2020, vol. 1569, no. 2, p. 22088, doi: 10.1088/1742-6596/1569/2/022088.

[23] R. T. Fielding, R. N. Taylor, J. R. Erenkrantz, M. M. Gorlick, J. Whitehead, R. Khare, and P. Oreizy, “Reflections on the Rest Architectural Style and ‘Principled Design of the Modern Web

Architecture' (impact paper award)," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, 2017, pp. 4–14. doi: 10.1145/3106237.3121282.

[24] R. Ivanov and V. Velkova, "Tangible and Personalized Smart Museum Application," *Digit. Present. Preserv. Cult. Sci. Herit.*, 2023, vol. 13, pp. 97–106, DOI: https://doi.org/10.55630/dipp.2023.13.9.

[25] A. Sanmorino and R. M. Fajri, "The Design of Notification System on Android Smartphone for Academic Announcement.," *Int. J. Interactive Mob. Technol.*, 2018, vol. 12, no. 3, DOI: 10.3991/ijim.v12i3.8494

[26] J. Liu and J. Yu, "Research on Development of Android Applications," in *2011 4th International Conference on Intelligent Networks and Intelligent Systems*, 2011, pp. 69–72. doi: 10.1109/ICINIS.2011.40.

[27] A. T. Kabakuş, "A Performance Comparison of SQLite and Firebase Databases from a Practical Perspective," *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, 2019, vol. 7, no. 1, pp. 314–325, DOI: 10.29130/dubited.441672

[28] L. Venica, E. N. Irawan, and D. I. H. Putri, "IoT with Firebase: Smart Ring Android App Using Max30100 for Fatigue Detection," *J. Electric. Electron., Inf. Comm. Technol.*, 2024, vol. 6, no. 1, pp. 8–15, DOI: https://dx.doi.org/10.20961/jeeict.6.1.81312

[29] A. Sofwan, F. R. Adhipratama, and K. Budiraharo, "Data Communication Design Based on Internet of Things Architecture For Smart Greenhouse Monitoring And Controlling System," in *2022 5th International Conference on Information and Communications Technology (ICOIACT)*, 2022, pp. 205–209, DOI: 10.1109/ICOIACT55506.2022.9971912.

[30] M. F. Younis and Z. S. Alwan, "Monitoring the Performance of Cloud Real-Time Databases: a Firebase Case Study," in *2023 Al-Sadiq International Conference on Communication and Information Technology (AICCIT)*, 2023, pp. 240–245, DOI: 10.1109/AICCIT57614.2023.10217953.