International Journal of Computing Sciences Research (ISSN print: 2546-0552; ISSN online: 2546-115X)
Vol. 9, pp. 3925-3952
doi: 10.25147/ijcsr.2017.001.1.253
https://stepacademic.net

Short Paper

Leveraging the Synergistic Connectivity of Firebase Database for a Windows Desktop Clinic Management System and Android-based Mobile Incident Reporting

Norris Alexis P. Amora
Graduate School, Institute of Computer Science, College of Arts and Sciences
University of the Philippines Los Baños, Philippines
npamora@up.edu.ph, sirronsixela@gmail.com
(corresponding author)

Concepcion L. Khan
Institute of Computer Science, College of Arts and Sciences
University of the Philippines Los Baños, Philippines
clkhan@up.edu.ph

Date received: March 30, 2025
Date received in revised form: April 30, 2025
Date accepted: September 6, 2025

Recommended citation:

> Amora, N. P., & Khan, C. L. (2025). Leveraging the synergistic connectivity of the Firebase database for a Windows desktop Clinic Management System and Android-based mobile incident reporting. *International Journal of Computing Sciences Research*, 9, 3925-3952. https://doi.org/10.25147/ijcsr.2017.001.1.253

Abstract

Purpose – This study aims to conduct case study testing and build synergistic connectivity of the Firebase database for a Windows desktop-based clinic management system and an Android-based mobile incident reporting system using the Firebase database.

Method – The research design utilized was case study testing, which will be used to establish the connectivity of the Firebase database in the Windows desktop application and Android-based application. It is also a developmental and descriptive study aimed at determining whether the problem has been addressed. The researchers used ISO 25010 to evaluate both applications.

Results – All case study testing delivered the functionality of modules and experiments in both applications when connected to the Firebase database. The case study testing revealed that it was feasible to build a clinic management system that integrates a Firebase database in a Windows desktop application, providing new insights into the study. Both applications were evaluated against ISO 25010, covering the characteristics of functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. In the Windows-based clinic management application, clinic staff scored an average of 4.37, while registered nurses scored 4.78. In the Android-based incident reporting application, a score of 4.90 was received.

Conclusion – The research was able to explore the connectivity of the Windows-based and Android-based applications using the Firebase database as a backend service. Based on the feedback received, both applications came out as 'Excellent' in the overall ISO 25010 results.

Recommendations – It is recommended that future development be conducted and studied in a multiplatform desktop and mobile application connected to a Firebase database, utilizing various technological tools for development. Another recommended improvement for the Windows desktop application is the integration of machine learning, given that ML.NET is available in C#, allowing programmers to use it in their .NET applications without leaving the ecosystem.

Research Implications – This study enabled the connectivity of Windows and Android-based applications using the Firebase database. The system was implemented in a school clinic, and the software's quality was continuously improved.

Keywords – Firebase database, clinic management system, incident reporting, Windows application, Android application

INTRODUCTION

Firebase database is known for empowering the digital world through web and mobile applications. The purpose of this study is to test the capabilities of Firebase in a Windows desktop application and extend the testing. In the field of application development, selecting the appropriate backend technology can have a huge impact on the success of an application. This study will explore Firebase as a Google backend-as-a-service for a Windows desktop application and demonstrate whether it will be the better option for a specific project, such as a clinic management system. In addition, an Android-based mobile application for incident reporting in the clinic will be connected to Firebase and subjected to testing.

The use of databases in our daily lives is common, though we aren't always aware of it. There is a lot of data in the world that is related, so we need a system to handle it, as discussed by Ohyver et al. (2019). Applications for smartphones and desktop/laptop computers are frequently used by a variety of users to improve the efficiency of managing various transactions.

The research problem is to identify and conduct case study testing in a Windows application utilizing a Firebase database, along with exploring the Firebase database's connectivity in Windows and Android-based applications. The researchers also interviewed staff at a school clinic where the application's development would benefit them and highlighted present problems that need automation and solutions. According to the researchers' interview, the school clinic used a spreadsheet to manage and collect data about medical record submissions by students, instructors, and staff. The use of an Excel file resulted in the generation of various worksheets for processing different information sets. Filtering is difficult to perform and provides minimal protection for the data in Excel. Searching for specific records is difficult because they are separated into worksheets, and data cannot be quickly obtained. Redundant data collection to monitor students' medical records requires them to fill out a Google form to confirm that they have submitted the necessary medical requirements.

Reporting of student injuries, illnesses, and accidents on campus is only documented once the student arrives at the clinic; there is no filing of incident reports that can notify the nurse promptly on campus. The nurse will only be reached after the patient has been brought into the clinic. According to the survey conducted by the researcher, out of 44 respondents, who are college students, 97.1 percent believe that it would be useful to have an application on campus for reporting incidents such as injuries or illnesses to the clinic. The study aims to test and develop the synergistic connectivity of the Firebase database for a Windows desktop-based clinic management system with data visualization and an Android-based mobile incident reporting system.

LITERATURE REVIEW

Segun-Falade et al. (2024) examined the role of cloud integration in desktop and mobile operating systems in their study "*Evaluating the Role of Cloud Integration in Mobile and Desktop Operating Systems*", noting that it enhanced usability, efficiency, and functionality. They emphasized that scalable storage and data synchronization were key factors driving the success of cloud integration.

Ho et al. (2024) discussed how digitization transformed traditional paper-based tasks into digital data, enabling them to be processed more efficiently and with less waste in their study "*Streamlining Dental Clinic Management for Effective Digitisation Productivity and Usability*." They proposed a web-based dental clinic management system that provided an excellent user experience for both patients and clinic staff. The authors

concluded that such a system provided significant convenience and improvement for dental clinics.

Omran (2024) discussed the use of cloud computing in mobile healthcare applications, which helped a variety of users, including physicians, patients, and medical personnel, in the study "Mobile Healthcare Application on Android OS Using Cloud Computing." The study also addressed the implementation and security architecture to ensure secure communication.

Alshehhi et al. (2023) discussed that bar and pie charts were the most popular formats for displaying health data in their study "*Understanding User Perspectives on Data Visualization in mHealth Apps: A Survey Study*." They found that end users did not want to browse complex charts. Additionally, users preferred a combination of summary texts and charts to ensure accurate chart interpretation. The study also highlighted challenges faced by users, such as incorrect information presentation, inaccurate touch interfaces, and overwhelming data.

Zelensky (2023) discussed the method of employing the Microsoft Visual Studio environment to develop desktop and mobile applications using Java programming languages in the study "*Construction Surface Using Desktop and Mobile Applications*", The research also covered the use of relational database management systems like Access and SQL Server, alongside cloud databases such as Firebase Real-Time Database, which allowed developers to store and synchronize data across multiple clients.

Marbella et al. (2023) discussed that clinics and hospitals were examples of health facilities typically found in the community in their study "*Design and Development of a Web-Based Patient Management Information System*." The authors developed the information system after identifying the problem.

Bakar et al. (2023) discussed the advantages and use of a clinic management system (CMS) in medical environments in their study "Clinic Management System." The research focused on the goals of the CMS, including eliminating form-writing mistakes, creating consistent procedures and data formats, and producing reports for patient illness data storage and doctor scheduling. The authors concluded that the digital platform reduced the possibility of errors occurring when forms were written by hand, thereby improving accuracy and quality.

Zala et al. (2022) discussed in their study "*PRMS: Design and Development of Patients'* E-Healthcare Records Management System for Privacy Preservation in Third Party Cloud Platforms" that personal data stored on a public platform was viewed as a major source of concern in terms of security and privacy. They emphasized that patient data and data management in e-health should have adhered to set standards.

Lwin et al. (2021) discussed in their study "*Firebase as a Backend Service in Mobile and Web Application Development*" the usage of the Firebase database in the development of both web and mobile applications, utilizing its services to provide the essential functionality. The authors concluded that their system received positive feedback from end users and that Firebase's services simplified development.

Demissie and Ranise (2021) discussed in their study "*Assessing the Effectiveness of the Shared Responsibility Model for Cloud Databases: The Case of Google Firebase*" that web and mobile apps were shifting their databases to the cloud. They mentioned the popularity of the Firebase database among application developers and noted that, since Firebase could be used on a variety of platforms, developers should improve its security.

Ignacio (2021) conducted research entitled "Mobile Application for Incident Reporting", which focused on three types of incidents: public disturbance, ordinance violation, and crime. The study found that the application was rigorously tested and received an excellent rating based on ISO 25010, as assessed by the respondents. This research provided valuable insights into the effectiveness of mobile applications in enhancing incident reporting processes.

Ouda (2020) conducted research entitled "Cloud Computing Service Providers: A Comparative Study", which discussed the new advent of technology in managing resources and sharing data across global networks, focusing on cloud computing. The study compared Google, Microsoft, and Amazon in terms of various aspects. Based on their results, all three platforms had good documentation and provided a variety of services in the cloud.

Oliveira et al. (2020) conducted research entitled "On the Adoption of Kotlin on Android Development: A Triangulation Study", discussing how programmers adopted Kotlin as a programming language for Android development. Their findings indicated that programmers found Kotlin easy to understand and implement, and they believed that it could increase productivity, readability, and code quality.

Lubis et al. (2019) conducted research entitled "Clinic Management System: Business Process Re-engineering based on User Experience (UX)", which discussed how the clinic management system was designed to manage and organize procedures in healthcare facilities to ensure they met patient demand. The study concluded that patient data was considered the property of the patient, which staff only borrowed and were required to safely return in its original form.

METHODOLOGY

System Architecture

Figure 1 shows the system architecture of the developed system; it illustrates the structure of the software and how it is organized. The interconnection of the system components can be traced through their data links, demonstrating the various functions performed by each component.

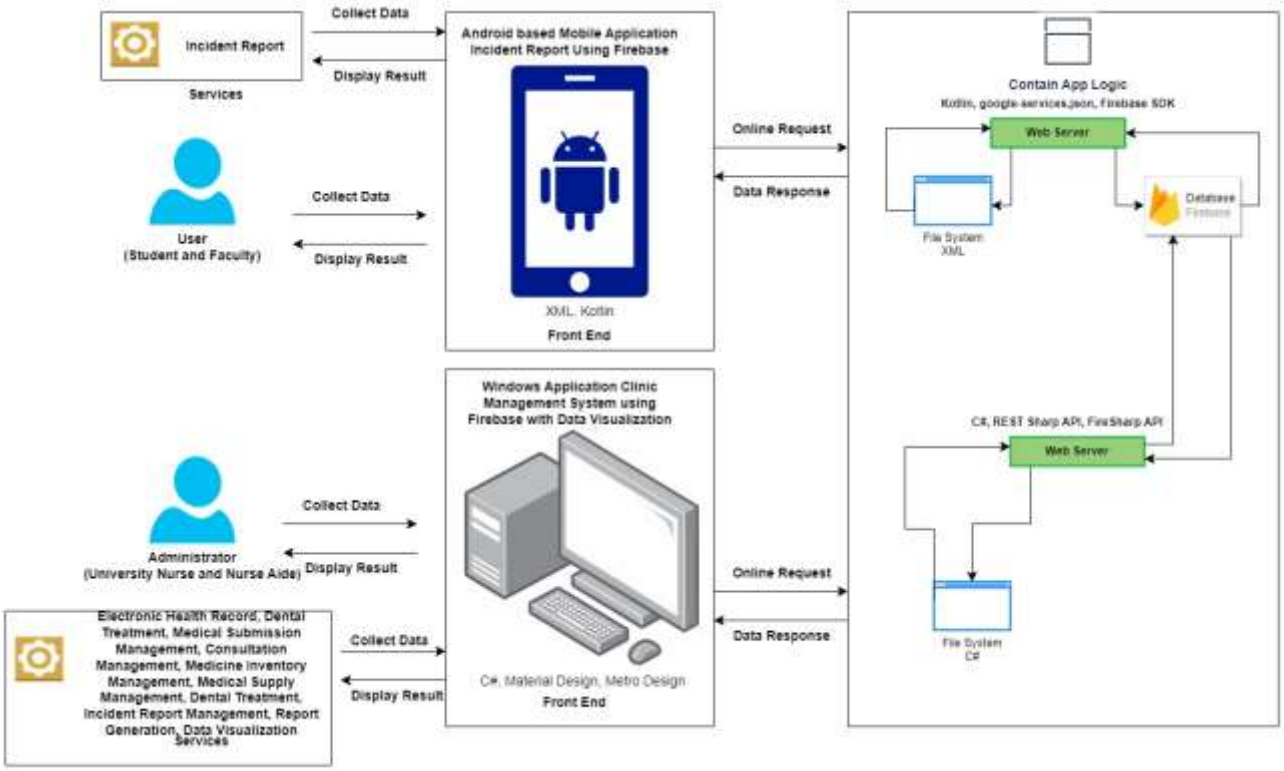

Figure 1. System Architecture of the developed system

Use Case Diagram

Figure 2 shows the use case diagram of the clinic management system. The diagram illustrates the administrator actor, which consists of all the modules for the system that pertain to school clinic transactions.

Figure 3 shows the continuation of the use case diagram of the clinic management system on the left side of the figure. On the right side, it presents the use case diagram of the users of the Android mobile application for incident reporting in the clinic.

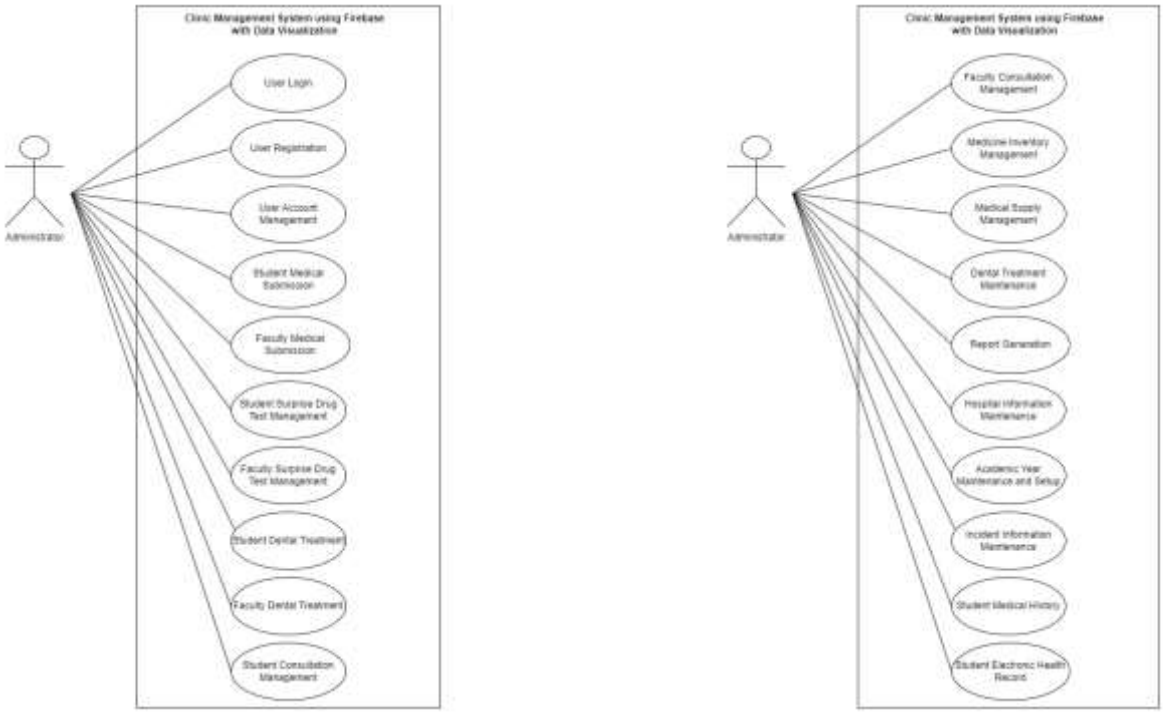

Figure 2. Use Case - Administrator of Clinic Management System Part 1

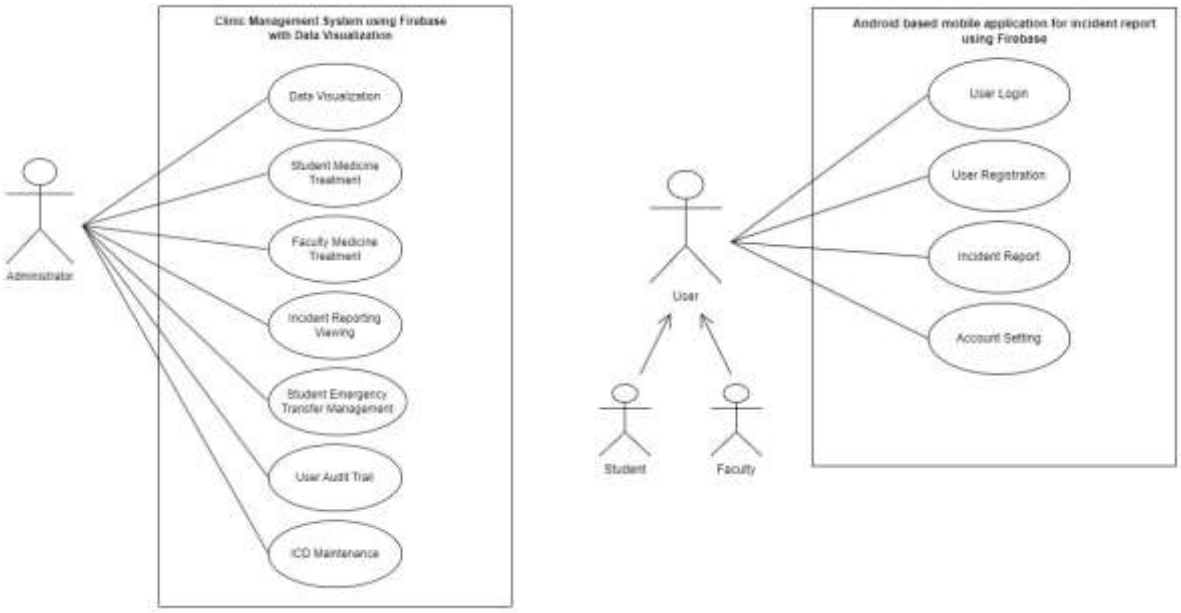

Figure 3. Use Case – Administrator of Clinic Management System Part 2 (left) and Use Case - User of Android Incident Reporting (right)

End-to-End Application Architecture

Figure 4 shows the application software tools, packages, and programming languages used in building the Windows desktop clinic management system, which utilized Visual

Studio Community, C#, .NET, Google Material Design, Metro Modern UI, MS Chart, Microsoft Report Viewer, Firebase REST API, Microsoft .NET Framework, and Firebase Database. The Android-based mobile application for incident reporting in the clinic used Android Studio, Kotlin, XML, Android Virtual Device, and Firebase database.

Figure 4. End-to-End Application Architecture

Statistical Treatment of the Data

The researcher collected data from the respondents, which was organized and compiled. It was then statistically treated in order to justify the questions given to the study respondents.

1. The weighted mean of each item in the user evaluation of both applications was calculated using the following formula:

$$WM = \frac{5(fsa) + 4(fa) + 3(ffa) + 2(fd) + 1(fsd)}{N}$$

2. The Ordinal Scale used data that was categorized into groups that could be rated. This scale of measurement was applied to calculate the weighted mean of each item in the five-variable set.

Table 1. Statistical Limits of Numerical Response Ratings

Numerical Descriptor	Range Scale	Level of Agreement	Interpretation
5	4.21-5.00	Strongly Agree (sa)	Excellent
4	3.41–4.20	Agree (a)	Average
3	2.61-3.40	Fairly Agree (fa)	Below Average
2	1.81-2.60	Disagree (d)	Poor
1	1.00-1.80	Strongly Disagree (sd)	Very Poor

ISO 25010

The quality model was the foundation of the product quality evaluation system. The researcher utilized ISO/IEC 25010 and adopted quality characteristics such as functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability, along with their respective sub-characteristics. It was selected because it directly aligned with identifying the study's objectives, particularly in determining where testing should be conducted. It was used in the study because it provided a standard way of evaluating software quality and was widely accepted in research. These characteristics highlighted key quality aspects of the system, while others, such as safety, were not included because the software testing did not involve life-threatening risks. The quality of a system was determined by how well it met the stated and implicit needs of all of its users, ultimately resulting in value.

Table 2. ISO 25010

Characteristics	***Sub Characteristics***
Functional Suitability *- This characteristic indicates the degree to which a system or product fulfills explicit and implicit requirements when utilized in accordance with established guidelines.*	**Functional Completeness** - The degree to which the set of functions addresses all of the defined tasks and objectives of the user. **Functional Correctness** - The degree to which a product or system produces accurate outcomes with the necessary level of precision. **Functional Appropriateness** - The degree to which the functions support the completion of specific activities and objectives.
Performance efficiency *- This characteristic indicates performance in relation to the amount of resources consumed under the conditions given.*	**Time Behavior** - The degree to which a product or system satisfies specifications in terms of response, processing, and throughput rates while operating. **Resource Utilization** - The degree to which the amounts and types of resources used by a product or system to accomplish its operations meet criteria. **Capacity -** The degree to which the permitted limits of a product or system component satisfy specifications.
Compatibility - *The degree to which a system, product, or component may share hardware or software environments with other systems, products, or components to exchange information and/or carry out necessary operations.*	**Co-existence -** The degree to which a product can carry out its necessary tasks effectively while coexisting with other items in the same environment and with the same resources, all without negatively affecting the other products. **Interoperability -** The degree to which information may be shared and used between two or more products, systems, or components.

Table 2. ISO 25010 (cont.)

Characteristics	*Sub Characteristics*
Usability **-** *The degree to which a system or product may be utilized by specific users to fulfill specific objectives in a specific context of usage with efficacy, efficiency, and satisfaction.*	**Appropriateness Recognizability** - The degree to which users can determine if a product or system is suitable for their requirements. **Learnability** - Degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk, and satisfaction in a specified context of use. **Operability -** The degree to which a product or system contains features that make it simple to use and control. **User Error Protection -** The degree to which a system safeguards users from making errors. **User Interaction Aesthetics** - The degree to which a user interface allows for pleasant and satisfying engagement for the user. **Accessibility -** The degree to which a product or system can be utilized by individuals with diverse features and abilities to achieve a specific objective in a specific context of usage.
Reliability **-** *The degree to which a system, product, or component performs defined functions under given conditions for a set amount of time.*	**Maturity** - The degree to which a system, product, or component satisfies dependability requirements during regular operation. **Availability -** The degree of functionality and accessibility of a product, system, or component when needed for usage**.** **Fault Tolerance** - The degree to which a product, system, or component continues to function as planned even in the face of hardware or software flaws. **Recoverability -** The degree to which a product or system can restore the immediately affected data and return the system to its intended condition in the case of an interruption or failure.

Table 2. ISO 25010 (cont.)

Characteristics	*Sub Characteristics*
Security *- The degree to which a system or product secures information and data so that users or other systems or products can access it to the right extent, depending on the kinds and degrees of permission.*	**Confidentiality** - The degree to which a system or product ensures that only those with authorization may access data. Integrity **Non-repudiation** - The degree to which actions or occurrences may be validated beyond a reasonable doubt, preventing their subsequent repudiation. **Authenticity** - The extent to which the identity of a subject or resource can be proven to be the one claimed. **Accountancy** - The degree to which an entity's activities may be uniquely linked to it.
Maintainability *- This characteristic shows how well and efficiently a system or product may be adjusted to make corrections, enhance its functionality, or adjust to changing needs and environmental conditions*	**Modularity** - The degree to which a computer program or system is made up of distinct components so that changing one part has little impact on other parts. **Reusability** - The degree to which a resource may be built upon or utilized in many systems. **Analysability** - The degree of efficacy and efficiency with which a product or system may be assessed for the impact of a planned change to one or more of its parts, diagnosed for defects or reasons of failure, or identified for modification. **Modifiability** - The degree to which a product or system may be successfully and efficiently updated without introducing flaws or reducing existing product quality. **Testability** - The degree of efficacy and efficiency with which test criteria for a system, product, or component may be defined and tested to see if they are satisfied.
Portability *- The degree to which a product, system, or component may be effectively and efficiently moved from one operating or usage environment or piece of hardware to another using software or other tools.*	**Adaptability** - The degree to which a product or system may be successfully and efficiently adapted to new or changing hardware, software, or other operational or usage settings. **Installability** - The degree of efficiency and efficacy with which a system or product may be installed and/or removed effectively in a given setting. **Replaceability -** The extent to which a product may function in the same context and replace another designated software product for the same purpose.

Software Case Study Testing and Evaluation

The researcher conducted a software case study testing and evaluation on both a Windows-based clinic management system and an Android-based incident reporting application, which share the same Firebase database.

RESULTS

Table 3 presents the case study results for the clinic management system, detailing various implementation scenarios, scopes, and experimental outcomes. The case study testing covers Firebase database connectivity within the Windows application, login security, menu navigation, and access to all system modules, including transaction processing within the clinic. The table presenting the case study testing of the clinic management system is provided in Appendix A.

Table 4 presents the case study results for the incident reporting system, detailing various implementation scenarios, scope, and experimental outcomes. The case study testing on the Android-based incident reporting application covers Firebase database connectivity, user login with credentials, account registration, and incident submission.

Table 4. Software Case Study Testing (Incident Reporting)

Case Implementation	Scope	Results	Status
Firebase Database Connectivity in Android Mobile Application	Test the connectivity of the Firebase database in the Android application by adding Firebase to the app.	The mobile application was able to connect to the Firebase database by adding the google-services.json file to the program with the proper configuration in the application.	Passed
Log in with valid credentials	Test login with valid credentials, such as correct accounts	The mobile application successfully authenticated the valid account	Passed
Login with invalid credentials	Test login with invalid credentials, such as an incorrect account	The mobile application provided an error message for incorrect accounts.	Passed
Account Registration	Test the Account Registration Feature	The mobile application successfully registered a new account.	Passed

Table 4. Software Case Study Testing (Incident Reporting) (cont.)

Case Implementation	Scope	Results	Status
Incident Reporting	Test the Incident Reporting Feature	The mobile application allowed users to submit incident reports in the clinic. The clinic management system was able to retrieve the submitted data.	Passed

The evaluation summary results of the Windows desktop clinic management system, using Firebase with data visualization, are presented. In terms of functional suitability, the clinic staff respondents left a rating of 4.37, which was interpreted as excellent. The registered nurse respondents left a rating of 4.78, which was also interpreted as excellent. It was part of the objective of the study to build a clinic management system integrated with a Firebase database that was functional. It functioned and operated according to the behavior that was expected to be integrated into the system.

Table 5. Evaluation Summary Results of Windows Desktop Clinic Management System using Firebase with Data Visualization

	CLINIC STAFF		REGISTERED NURSES	
Criteria	**Mean**	**Interpretation**	**Mean**	**Interpretation**
Functionality Suitability	4.33	Excellent	4.80	Excellent
Performance Efficiency	4.33	Excellent	4.90	Excellent
Compatibility	4.17	Average	4.80	Excellent
Usability	4.28	Excellent	4.80	Excellent
Reliability	4.42	Excellent	4.70	Excellent
Security	4.53	Excellent	4.80	Excellent
Maintainability	4.47	Excellent	4.70	Excellent
Portability	4.44	Excellent	4.77	Excellent
Overall Mean	**4.37**	**Excellent**	**4.78**	**Excellent**

According to the evaluation results stated above, the functional suitability had a higher score of 4.35 and was the best criterion passed by the Android-based mobile application, while portability had a score of 4.24, which was the inverse. However, both were still excellent in interpretation. Overall, the mobile application scored a total mean of 4.29, which was deemed excellent based on the ISO 25010 software quality model used with the respondents. In short, the application fulfilled its stated purpose of allowing users to report incidents in the clinic management system. This ensured that the staff members were aware of the present state of events, and the evaluation results guaranteed the system's quality.

Table 6. Evaluation Summary Results of Android-Based Mobile Application Incident Reporting using Firebase Database

Criteria	Mean	Interpretation
Functionality Suitability	4.35	Excellent
Performance Efficiency	4.27	Excellent
Compatibility	4.26	Excellent
Usability	4.31	Excellent
Reliability	4.26	Excellent
Security	4.34	Excellent
Maintainability	4.28	Excellent
Portability	4.24	Excellent
Overall Mean	**4.29**	**Excellent**

Table 7 outlines the criteria for the Windows desktop clinic management system integrated with the Firebase Database. It covers key areas such as database connection setup, CRUD operations and API calls, application and database integration, real-time functionality, data import, data structure, querying, cost, and backup and recovery. The table also presents the results for each of these areas, along with discussions based on observations from the experiment and development process.

Table 7. Criteria of the Windows desktop clinic management system integrated with Firebase

Area	Result	Discussion
Database Connection Setup	The application was successfully integrated with the Firebase database using the Firebase API.	This approach was crucial for ensuring that the application maintained a consistent connection to the Firebase database. It used FireSharp, FireSharp.Serialization.Json.Net, Newtonsoft.Json, and RestSharp to integrate the Windows application with the Firebase database. It configured Firebase in the application through authentication secrets and base paths. It was handled by the Firebase client for connecting the application to the database.
CRUD Operations and API Calls	The application executed CRUD operations.	Inserting data from the application was done using the push method. Retrieving data from the application using the get method proved effective. Updating records in the application was done using the update method. Removing records was also accomplished using the delete method.

Table 7. Criteria of Windows desktop clinic management system integrated with Firebase (cont.)

Area	Result	Discussion
Application and database integration	Firebase databases were used with the application. It functioned effectively when used in different clinic transaction modules.	The application's database integration functioned smoothly, enabling the system's transactions and operations to run efficiently.
Real time	The async method was used to set up the real-time system and refresh the data retrieval.	The database was real-time, but it had to be set up in a Windows-based application. It used an approach like the async method for tasks that were handled, such as auto-refreshing the table. It was properly handled in the code, or else it would slow down. The table refreshed only when there was a change in the data in the Firebase database to maintain good performance.
Importing Data	The application was able to import data through an Excel file containing a data set.	It took a minute if there were hundreds of records in the spreadsheet. It required loading and waiting time before uploading a set of data from the application to the Firebase database. It involved network latency and depended on the internet speed.
Data Structure	The application was able to handle the document structure stored in JSON format found in the Firebase database.	The application used the C# programming language to handle data in the Firebase database using get and set. The Firebase response was used to retrieve the tree structure values that represented keys and values in the database through the Firebase client. The results were stored in a dictionary and transferred into a data table format. Data View in C# created filters and custom views for complex data in the database.

Table 7. Criteria of Windows desktop clinic management system integrated with Firebase (cont.)

Area	Result	Discussion
Querying	The application was able to handle specific conditions when retrieving data.	The C# programming language and Firebase database were used to manage the data in the Firebase database. It performed basic queries in the Firebase database and extended features through C# by using data view filters and method approaches to handle complex transactions in the clinic management system. This applied to a variety of transactions, including content management, data visualization, report generation, and security.
Cost	The application was able to use the free tier, such as the Spark plan, which included 1 GB of free data storage.	The application took advantage of Firebase's free tier, which was ideal for developing the system's prototype. To improve storage, it was recommended that the developer use the Blaze plan, which was pay-as-you-go. Despite the free offer, the application ran smoothly and performed effectively in various transactions; however, an internet connection was required.
Backup and recovery	It automatically backed up the data, as it was stored in the cloud.	The data was stored in the Firebase database. Given that the data was stored in the cloud, the application did not require manual backups.

The figure below represents some modules that came from the clinic management system and incident reporting.

Figure 5 depicts the main form that the user encounters once they are granted access by the system. It includes a list of menu items available to the user. The user selects the module they intend to access in the system. When the user clicks on a menu item, it takes them to that module.

Figure 5. Main Menu

The Student Electronic Health Record, as shown in Figure 6, serves to manage electronic health records, which comprise a student's information, vitals, symptoms, diagnosis, reason for the visit, and prescriptions. It consists of multiple linked tables, as they all belong to a specific student. An auto-generated EHR code uniquely identifies each student's electronic health record. The system stores these records in a tree structure within a Firebase database. This structure serves to enable smooth data loading into the application and ensures consistent execution throughout each transaction.

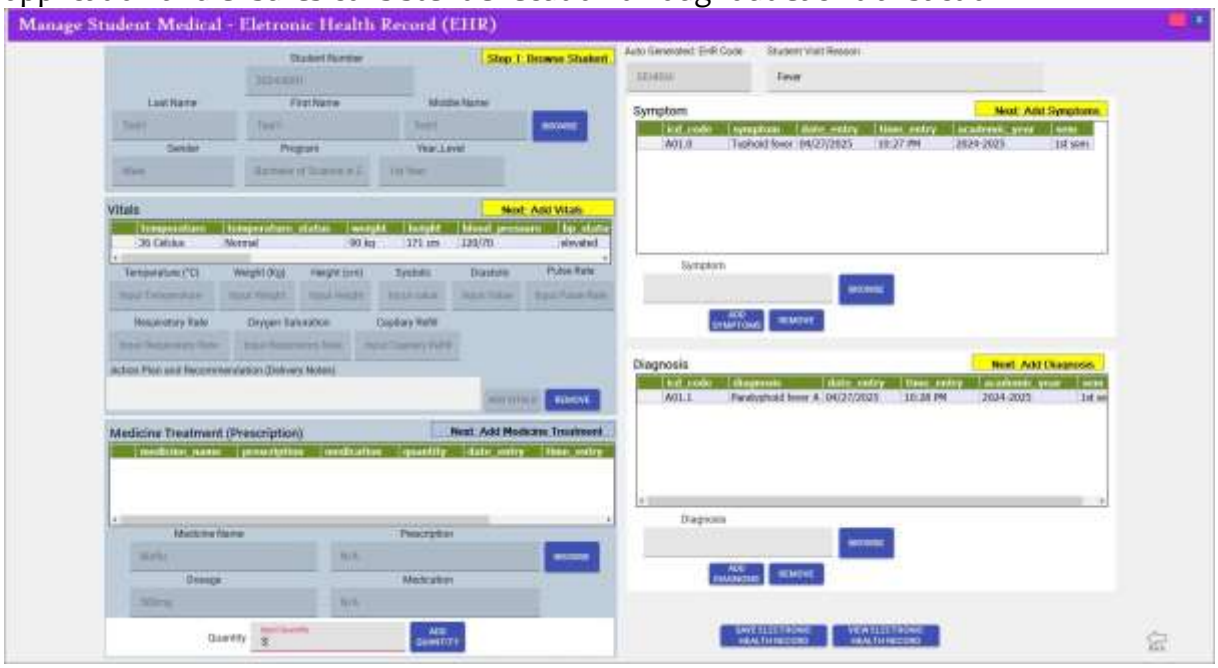

Figure 6. Student Electronic Health Record (EHR)

Figure 7 depicts the student consultation data visualization, which generates monitoring results based on the student's vitals, such as temperature, capillary refill, oxygen saturation, respiratory rate, pulse rate, and blood pressure. It also includes the number of student consultations categorized by gender and course. Additionally, it shows the frequency of consultation concerns. The system supports filtering through a date-time picker based on specified criteria. The software gathers crucial data from student consultations and displays them as charts. Despite a high volume of filtering requests in the database, the application loads the data visualizations smoothly. The technique used involves retrieving all data and then applying filters based on the monitoring criteria in student consultations.

Figure 7. Student Consultation Data Visualization

The Student Dental Treatment Report, as shown in Figure 8, covers the generation of reports that can be previewed and printed within the application. It allows users to view the report and print it based on a selected date range and course. Additionally, it demonstrates how data from the Firebase database can be retrieved and populated in the report viewer.

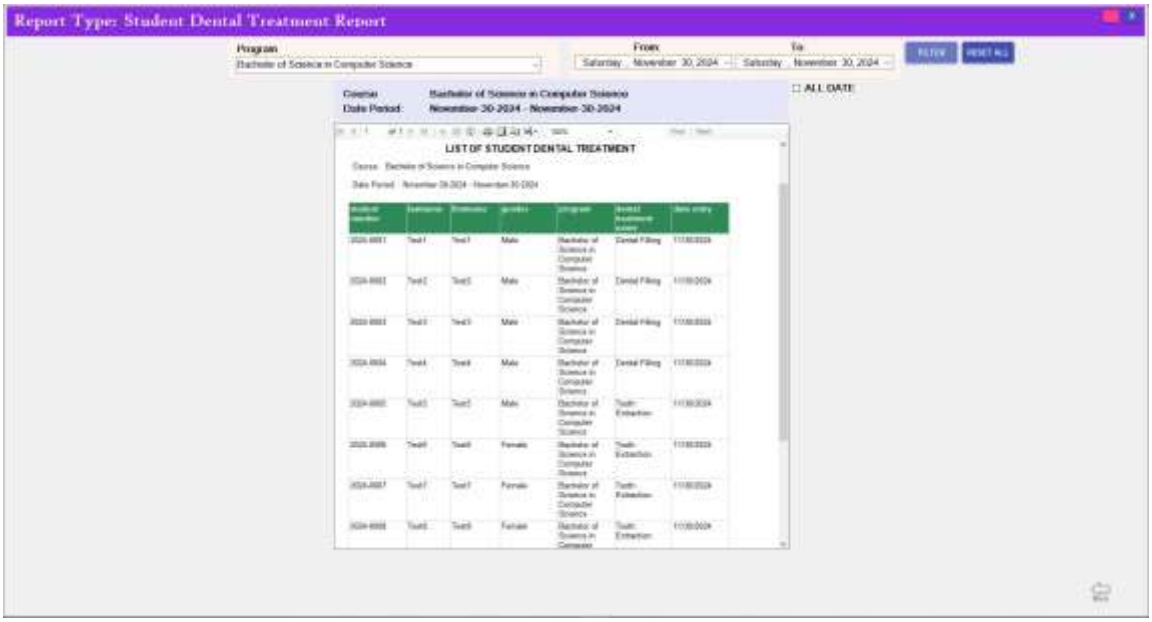

Figure 8. Report Generation: Student Dental Treatment Report

Figure 9 depicts the Android-based mobile incident reporting application, which is designed to submit incident report data to the clinic software, where clinic staff are immediately notified. The table displayed in the Windows desktop clinic management application instantly refreshes to reflect any modifications or additions. This setup demonstrates how the desktop and mobile applications communicate with each other by sharing the same Firebase database.

Figure 9. Incident Reporting

CONCLUSIONS AND RECOMMENDATIONS

Based on the findings, all the case study testing was able to deliver the functionality of modules and experiments in both applications while connected to the Firebase database. Both applications were evaluated based on ISO 25010, with the

characteristics of functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. In the Windows-based clinic management application, clinic staff scored an average of 4.37, while registered nurses scored 4.78, signifying that the respondents were satisfied with the overall operational state of the application. In the Android-based incident reporting application, a score of 4.90 was received, signifying that it performed its operation in the submission of incident reports in the clinic. With the successful development of both applications that used the Firebase database, it was feasible to conclude that the research met its aims and purpose of studying the Firebase database in both applications. Indeed, the Firebase database was an option to use as a backend service in the desktop clinic management system and mobile application for clinic incident reporting.

It is recommended to explore the Firebase database in a multiplatform environment using different development tools for desktop and mobile applications. Additionally, integrating machine learning with ML.NET to generate predictions is recommended as a further improvement to the clinic management system.

IMPLICATIONS

The study enabled the Windows desktop clinic management system and Android-based incident reporting application to connect to the Firebase database. The integration of the Firebase database in both applications provided new insights. The system was implemented in a school clinic, and the software's quality was continuously improved.

ACKNOWLEDGEMENT

First and foremost, the researchers would like to express deepest appreciation and gratitude to the Almighty God, from the bottom of our hearts, for all of his blessings and spiritual lessons that help us to fulfill the project. The researchers extend their thanks to the University of the Philippines Los Baños, the Institute of Computer Science, and the Graduate School.

FUNDING

This study did not receive any funding from any organization.

DECLARATIONS

Conflict of Interest

The researcher declares no conflict of interest in this study.

Informed Consent

Individual consent in accordance with Philippine data privacy laws was stated in the study respondent's software evaluation.

Ethics Approval

Not applicable

REFERENCES

Alshehhi, Y. A., Abdelrazek, M., Philip, B. J., & Bonti, A. (2023). Understanding User Perspectives on Data Visualization in mHealth Apps: A Survey Study. *IEEE Access*,1, 84200-84213. https://doi.org/10.1109/ACCESS.2023.3302325

Bakar, K., Nurpathama, M., & Hariri, H. (2023). Clinic Management System. *Jurnal Inovasi Global*. 1(2), 179-191. https://doi.org/10.58344/jig.v1i2.27

Demissie, B. F., & Ranise, S. (2021). Assessing the effectiveness of the shared responsibility model for cloud databases: The case of Google's Firebase. In *Proceedings of the 2021 IEEE International Conference on Smart Data Services (SMDS)* (pp. 121–131). IEEE. https://doi.org/10.1109/SMDS53860.2021.00026

Firebase. (2024). *Firebase Realtime Database*. Retrieved from https://firebase.google.com/docs/database

Ho, S.-B., Chew, E.-Y., & Tan, C.-H. (2024). Streamlining Dental Clinic Management for Effective Digitisation Productivity and Usability. *Journal of Informatics and Web Engineering*, 3(2), 70-85. https://doi.org/10.33093/jiwe.2024.3.2.5

Ignaco, M. A. E. (2021). Mobile Application for Incident Reporting. *JOIV: International Journal on Informatics Visualization*, 5(4), 388–394. https://doi.org/10.30630/joiv.5.4.741

ISO 25000. (2024). ISO/IEC 25010. Retrieved from https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

Lubis, M., Sutoyo, E., Handayani, D., & Azuddin, M. (2019). Clinic Management System: Business Process Re-engineering based on User Experience (UX). *Journal of Physics: Conference Series*. 1361(1), 012031. https://doi.org/10.1088/1742-6596/1361/1/012031

Lwin, C. S., Dim, N. K., & Aye, S. M. M. (2021). Firebase as a backend service [sic] in mobile and web application development. *J. Myanmar Acad. Arts Sci.*, 19(2), 309-317.

Marbella, H. N., Akbar, I. A., & Setiawan, B. (2023). Design and development of a web-based patient management information system. *Procedia Computer Science*, 234, 1799–1806. https://doi.org/10.1016/j.procs.2024.03.188

Mateus, B., & Martinez, M. (2020). On the adoption, usage, and evolution of Kotlin features in Android development. In *ESEM '20: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (pp. 1–12). ACM. https://doi.org/10.1145/3382494.3410676

Ohyver, M., Moniaga, J. V., Sungkawa, I., Subagyo, B. E., & Chandra, I. A. (2019). The comparison of Firebase Realtime Database and MySQL Database Performance using the Wilcoxon Signed-Rank Test. *Procedia Computer Science*, 157, 396–405. https://doi.org/10.1016/j.procs.2019.08.231

Ouda, G. K. (2020). Cloud computing service providers: A comparative study. *Samarra Journal of Pure and Applied Sciences*, 2(1), 76–89. https://uosamarra.edu.iq/wp-content/uploads/2021/09/280f00e221c1a458.pdf

Segun-Falade, O. D., Osundare, O. S., Kedi, W. E., Okeleke, P. A., Ijomah, T. I., & Abdul-Azeez, O. Y. (2024). Evaluating the role of cloud integration in mobile and desktop operating systems. *International Journal of Scholarly Research in Engineering and Technology*, 04(01), 019–031. https://doi.org/10.56781/ijsret.2024.4.1.0019

Zala, K., Thakkar, H. K., Jadeja, R., Singh, P., Kotecha, K., & Shukla, M. (2022). PRMS: Design and Development of Patients' E-Healthcare Records Management System for Privacy Preservation in Third-Party Cloud Platforms. *IEEE Access*. 10, 85777–85791. https://doi.org/10.1109/ACCESS.2022.3198094

Zelensky, O. (2023). Construction surface using desktop and mobile applications. Scientific and practical journal. *Economics and Technical Engineering*, 1(1), 87–102. https://doi.org/10.62911/ete.2023.01.01.07

Author's Biography

Norris Alexis P. Amora is affiliated with the University of the Philippines Los Baños Graduate School. His research interests include information technology, information systems, database systems, and computer programming. He enjoys hands-on tasks, such as writing programs, running them on a computer, and utilizing them effectively. With a strong passion for studying, he focuses particularly on topics related to computers and technology. He is also dedicated to sharing his knowledge and inspiring others to learn.

Concepcion L. Khan is an Associate Professor at the University of the Philippines, Los Baños. Her research interests include Digital Agriculture, Data Mining, Artificial Intelligence, and Information Systems. She consistently collaborates with her students and colleagues on research to contribute to the body of knowledge, provide innovative solutions, and share valuable insights. She is deeply passionate about teaching, research, and inspiring students to pursue their educational goals.

APPENDICES

Appendix A: Table 3 – Case Study Results for the Clinic Management System

Table 3. Software Case Study Testing (Clinic Management System)

Case Implementation	Scope	Results	Status
Connectivity of the Firebase Database in a Windows-based Application	Test the connectivity of the Firebase database in the Windows desktop clinic management system using the Firebase API in .NET	The application was able to connect to the Firebase database using FireSharp, FireSharp.Serialization.Json, Newtonsoft.Json, and RestSharp. These APIs are typically used when interacting with the Firebase database. Without an internet connection, the developer will be unable to test the application's code that interacts with Firebase.	Passed
User Login Password Encryption	Test the password encryption during user login	The application was able to encrypt the password using AES 256-bit encryption and saved the encrypted password in the database to prevent it from revealing the plaintext.	Passed
Log in with valid credentials	Test the login with valid credentials, such as correct account details	The application was able to authenticate the valid account for the clinic management system and enabled reading usernames and passwords from a list of accounts stored in the Firebase database.	Passed
Log in with invalid credentials	Test the login with invalid credentials, such as incorrect account details	The application was able to provide an error message for incorrect accounts.	Passed
Main Menu	Test the options in the main menu	The application was able to navigate through different menu items	Passed
Student and Faculty Entry Management	Test the student entry management for saving, updating,	The application successfully carries out the functionality required for student and	Passed

	retrieving table data, and searching with filters.	faculty entry management. Serialization and deserialization work effectively, as they handle both JSON and C# objects. Push, Get, and Update operations function correctly in the Firebase database. Filtration from C#.NET to the Firebase Database operates smoothly. By combining C# and Firebase Database API features to handle data, the transaction process becomes more efficient.	
Import Data	Test the data import using an Excel file	The application was able to import a set of data from Excel into the Firebase database. It checked each row of the spreadsheet table and converted it to JSON format. It pushed data until it reached the last record in the table. Uploads were dependent on internet speed. It took a few minutes if there were hundreds of rows in the spreadsheet.	Passed
Student and Faculty Medical Submission	Test the student and faculty medical submission management	The application successfully provided the required functionality for medical submission. Saving, updating, and searching with filtration all worked as intended. It displayed green for normal statuses and maroon for those with abnormal findings. Push, get, and update operations had been successful in previous modules; therefore, implementing them in this module was achievable. To detect each status, the system checked the retrieved value to	Passed

		determine whether it was normal or indicated abnormal findings, and then applied C# features to mark the status accordingly.	
Student and Faculty Surprise Drug Test	Test the student and faculty drug test management	The application successfully provided the functionality required for surprise drug testing for both students and faculty. Saving, updating, and searching with filtration all performed as expected. It displayed green for normal statuses and maroon for those with abnormal findings. Since this was already achievable in the previous module, managing data and displaying statuses in the table columns in this module also worked successfully.	Passed
Student and Faculty Dental Treatment	Test the Dental Treatment Management for Students and Faculty	The application enabled the user to search for student and faculty records and select dental treatment information. It saved, updated, retrieved, and searched with filters. This module followed the same coding approach to handle data.	Passed
Student Medical History and Electronic Health Record	Test the Student's Medical History and Electronic Health Record	The application maintained the medical history and electronic health record of the student. These modules included selecting from a variety of tables, such as student, symptoms, diagnosis, vitals, and medication treatments, that were relevant to the student. It used the GET method to retrieve values from different tables, starting with JSON format and converting them to a Data Table. It	Passed

		generated an EHR code to ensure the uniqueness of each submission based on the submitted record in the Firebase database.	
User Account Management	Test the user account management	The application managed the accounts by viewing and changing the status of accounts as either active or inactive. It used the GET method to retrieve a list of accounts. When the table was updated, it automatically refreshed. It was used in conjunction with the async method in C# to retrieve values from the Firebase database.	Passed
Student and Faculty Consultation Management	Test the student and faculty consultation management	The application was able to save, update, retrieve, and search with filters. The retrieval of the table automatically provided the status for each record.	Passed
Medicine and Medical Supply Inventory	Test the medicine and medical supply inventory management	The application maintained the medicine and medical inventory. It enabled users to add new items and update existing quantities or information. It searched for information using filters. It automatically indicated the status of good and critical items. This approach made it achievable to perform, as it handled data using C# and Firebase API features.	Passed
Student and Faculty Medicine Treatment Management	Test the student and faculty medical treatment management	The application was able to allow users to search for student and faculty records and then select medical treatment information. It also supported saving, updating, retrieving, and filtering data. The inventory of medicine and	Passed

		medical supply quantities was updated accordingly. Performing operations on two tables within a single transaction was made possible by following the same approach used in previous modules. The system used GET and UPDATE operations on the medicine inventory table and pushed data into the student medicine treatment table. Although the system retained a document-based structure, working with JSON format in relation to data tables had always been implemented.	
Report Generation	Test the generation of reports	The application was able to generate all reports in the system. Reports were viewed and filtered using the report viewer. It was also capable of saving the reports in PDF, Excel, and Word formats. Since the report viewer did not directly support Firebase databases as data sources, data retrieval was made possible by selecting the object and document tree structure and converting it into a data table.	Passed
Data Visualization	Test the data visualization	The application was able to produce data visualizations in different modules that contained data transactions. It was also capable of drilling through the charts.	Passed
Student Emergency Transfer Management	Test the student emergency transfer management	The application was able to manage the student emergency transfer information.	Passed
Content Maintenance	Test the maintenance modules	The application was able to manage records related to	Passed

		dental treatment, hospital contacts, courses, academic years, incident information, and ICD.	
User Audit Trail	Test the user audit trail	The application was able to record user activities within the system.	Passed