Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 4 (2023) 982 - 987 ISSN Media Electronic: 2580-0760

Implementation of System Development Life Cycle (SDLC) on IoT-Based Lending Locker Application

Melinda Melinda[1], Shaquille Rizki Ramadhan Na[2], Yudha Nurdin[3], Yunidar Yunidar[4]
[1,2,3,4]Department of Electrical and Computer Engineering, Faculty of Engineering, Universitas Syiah Kuala, Banda Aceh, Indonesia
[1]melinda@usk.ac.id, [2]shaquille@mahasiswa.unsyiah.ac.id, [3]yudha.nurdin@usk.ac.id, [4]yunidar@usk.ac.id

Abstract

Libraries are social institutions that provide information services that can be accessed publicly to meet the information needs of librarians. Based on the results of a survey conducted on 58 students who have visited the library of the Electrical and Computer Engineering Department of Universitas Syiah Kuala, an information system was needed that provided research book information related to the author's name, year of writing, field concentration, and abstract of the research book and there was a division of categories based on field concentration, and there was an online borrowing feature. Based on these problems, this study aims to implement an Android application system with IoT-based lending lockers using the SDLC (system development life cycle) Prototyping method. This study produces an application with a locker-based online lending feature, several other features as the user desires, and 1 prototype lending locker. The locker-based online lending system integrated with ESP32-WROOM-32 can connect with Firebase storage and send locker key codes to Firebase so the application can access them. Through experiments and tests conducted on the application, it is obtained that the application can access the locker key code and display it to the user. The application has also been validated using black-box and white-box testing and can be accepted by users based on System Usability Scale (SUS) average score with a very feasible interpretation category.

Keywords: *libraries; SDLC; locker-based; IoT-based*

1. Introduction

Libraries are social institutions that provide information services that can be accessed publicly to meet the information needs of librarians [1], [2]. The old paradigm describes the library as a collection of books and information sources in printed form with manual access. However, the digital era has changed the paradigm of the library as a medium of information in digital form that can be accessed online [3], [4].

A survey conducted by the author on 58 students majoring in Electrical and Computer Engineering (JTEK) Syiah Kuala University who visited the JTEK library in May 2022 obtained the results that 77.6 percent of respondents have experienced difficulties when looking for research books in the library, 74.1 percent of respondents have experienced incidents where the book they are looking for is not found, 98.3 percent of respondents feel that an information system is needed that provides research book information related to the author's name, year of writing, field concentration, and abstract of the research book. There is a division of categories based on field concentration and an online lending feature.

This study was conducted to solve the problem by creating an Android-based library application. Although there have been studies that create library applications such as in research [5]-[8]. However, this study differs from previous studies by adding a lending system with IoT-based lockers. The Internet of Things (IoT) is a technology that allows information about objects around the environment to be sent and controlled through the Internet network [9]. This technology will connect everything through the internet [10]. IoT has been implemented in smart homes, buildings, farms, smart electronic health, and military IoT [11], [12]. In the same way, IoT can be a solution to transmit data from lockers to servers via the internet [11] so that lockers and applications can connect and communicate online.

The application does not only aim to communicate with lockers online in book lending services. However, the application must be able to meet the criteria approved by the user and be able to solve the problems in the

Accepted: 27-04-2023 | Received in revised: 16-07-2023 | Published: 22-08-2023

previously conducted survey. Therefore, applications must go through a structured procedure to get results that satisfy users [13]. In addition to the addition of IoT systems, this study uses the Software Development Lifecycle (SDLC) method to create applications that meet user requirements. This study is expected to build an application that meets user needs functionally and non-functionally and produce an application tested for feasibility [14], [15].

Related research such as Locker Swarm [16] focuses on the use of Internet of Things (IoT) and full stack software development to offer an alternative smart locker solution. Instead of a physical key, each locker can be unlocked by scanning a Quick Response code (QR code), and its access can be shared among selected individuals. Users can search for the closest lockers, see the locations of all lockers, and check locker availability.

2. Research Methods

2.1 Software Development Life Cycle (SDLC) Model Prototyping

The software lifecycle is a descriptive and prescriptive characteristic of software development. It deals with various phases and parts, from planning to testing and deploying the application. All activities are performed in different ways according to the requirements. These activities are called the Software Development Lifecycle (SDLC). SDLC is a general methodology used in developing systems or software. The phases in the SDLC start from planning, analysis, design, and implementation to system maintenance [16].

Concerning mobile applications, information is needed about all application needs from users. Information about the needs of mobile applications needs to be collected first so that the application can fulfill the user's wishes [17]. Therefore, information is collected using the prototyping model SDLC method [18]. With this method, information can be collected quickly according to user needs. Errors can be detected quickly, and user feedback is more quickly available to produce the best solution [19].

Prototyping techniques focus on presenting aspects of the software that users will see. Users will evaluate prototypes to filter the needs of the software until they find the software specifications according to the user's wishes [19]. The stages of this development model are shown in Figure 1.

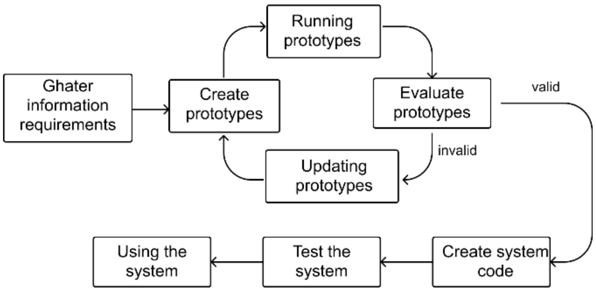


Figure 1. Prototyping Model Development Stages

Figure 1 shows the prototyping model SDLC method procedure, starting with collecting information on the needs required by the software. A prototype is made based on the collected information. Furthermore, the prototype will be evaluated by prospective users to get feedback. The feedback will be used as a reference to improve and update the prototype. When the user has validated the prototype, then the prototype will enter the coding process into a system. The next process after finishing coding the prototype is testing the system. Systems that successfully pass the testing process will be published for use by users.

2.2 IoT Systems

The lockers implemented in the book lending system use the ESP32-WROOM-32 module. This module allows applications and lockers to communicate and share data through the Firebase data storage service [20]. The IoT concept is applied to the lending locker utilizing the ESP32-WROOM-32 module to communicate and share data with the application through the Firebase storage service intermediary.

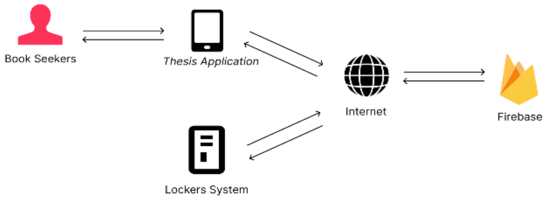


Figure 2. Workflow of IoT Systems

The ESP32-WROOM-32 has various specifications, ranging from individually controllable CPU cores to clock frequencies that can be adjusted from 80 to 240 MHz. In addition, ESP32-WROOM-32 also integrates a rich set of peripherals ranging from capacitive touch sensors, SD card interface, and ethernet to I2C [21]. From Figure 2, it is shown that the app and lockers require the Internet to connect with Firebase [22]. That way, the app, and lockers can communicate and share necessary data. The user will receive a locker key code that the locker sends to Firebase. Thus, the user can access the locker and retrieve the book that has been borrowed from the locker. The IoT concept for the lending locker implements a simple algorithm shown in Figure 3.

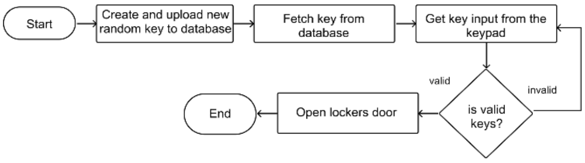


Figure 3. Workflow of the IoT Loan Locker Algorithm

The system will generate a new random 4-digit key. The generated key will be stored in the Firestore database so the application can access it. Then after the key is successfully generated and stored, the system will make another request to the database so that the key obtained by the application user has similarities with the key in the locker. When the user enters the key given by the electronic thesis application on the locker, the system will check the key similarity. When the given key is not the same, the system will give an error message on the LCD and ask the user to re-enter the key. When the key is the same, the system will open the locker door then the system will create a new key.

2.3 Mobile Application

Application requirements contain features or services that must be provided by the application, including the system's reaction to certain inputs and the application's behavior in certain situations. Application requirements depend on the application's user and where the application will be used [17], [23]. In this study, application requirements have been obtained based on the analysis of the survey results that have been carried out. The application needs in this study are divided into 2 based on user identity, namely guests and research book seekers. The application requirements in this study are shown in Table 1 and Table 2.

Table 1. Guest Requirements Information

Function Code	Function Name	Description
T-F-001	Registration	Guest can register on the system as a research book seeker.
T-F-002	Login	Guest can log into the system as a research book seeker.

Table 2. Research Book Seekers Requirements Information

Function Code	Function Name	Description
T-F-003	View the list of research books	Research book seekers can see a list of books according to the field concentration category according to the system
T-F-004	View research book details	Research book seekers can view detailed research book information on the research book detail page.
T-F-005	Borrow a research book	Research book seekers can borrow research books on the research book details page.
T-F-006	View borrowed books	Research book seekers can see a list of borrowed research books.
T-F-007	View loan details	Research book seekers can view detailed loan information on the loan details page.
T-F-008	See locker code	Research book seekers can see the storage locker code on the loan details page.
T-F009	Delete loan history	Research book seekers can delete the book borrowing history on the research book loan list page
T-F-010	Adding bookmarks	Research book seekers can add bookmark on research book details page.
T-F-011	View a list of bookmarks	Research book seekers can see a list of research books that have been added from the previous research book details page.
T-F-012	Removing bookmarks	Research book seekers can remove bookmark on research book detail page
T-F-013	Search for research books	Research book seekers can search for research books in the system by using the title of the research book as a search keyword.
T-F-014	Change username	Research book seekers can change username
T-F-015	Change password	Research book seekers can change password
T-F-016	Logout	Research book seekers may exit system.

In addition, a use case diagram is shown in Figure 4 that describes and illustrates the interaction of actors or users with the software system. In addition, use case diagrams are also used as verification requirements, provide an overview of the interface of a system, and provide certainty about system requirements [14]. The use case diagram is created by linking each functional requirement to each actor.

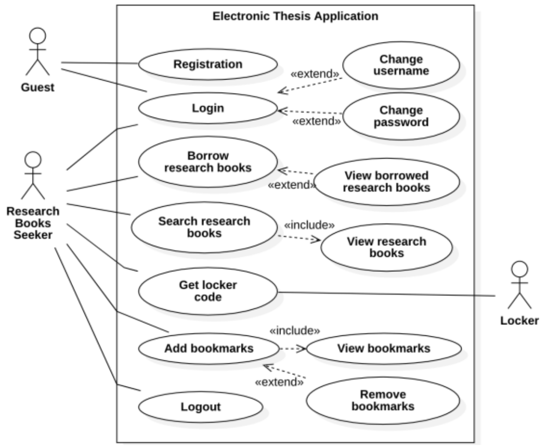


Figure 4. Diagram Use Case

2.4 Lockers System

The locker system consists of several components, namely ESP32-WROOM-32 as a microcontroller and interactor in the database [20], PCF8574 as a helper module that connects the LCD and Keypad with ESP32, 16x2 LCD as an interface to the locker [24], 4x4

DOI: https://doi.org/10.29207/resti.v7i4.5047

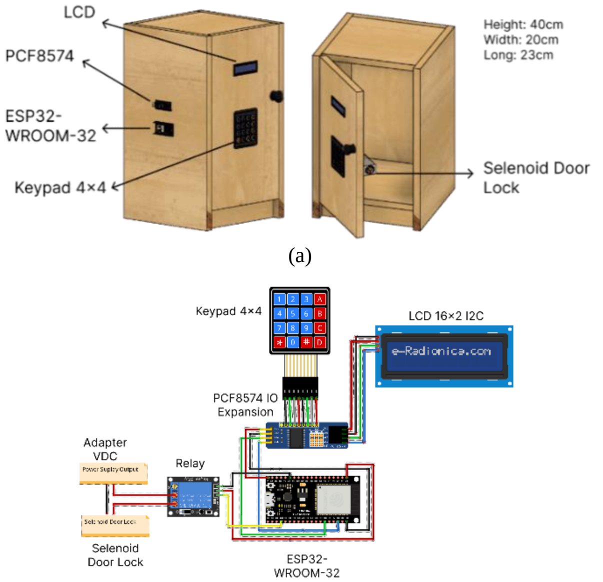
Keypad as input to the locker, Relay as a current breaker on the solenoid lock door, Selenoid Lock Door as a locker door lock. The architecture and circuitry of the locker system are shown in Figure 5, and locker specification are shown in Table 3.

Table 3. Locker Specification

Components	Total	Specifications
Wood Plank	3 meters	2 cm
Door Handle	1 unit	As required
Door Hinges	1 unit	As required

Table 4 Locker System Specification

Components	Total	Specifications
Selenoid mini small lock door	1 unit	Voltage 6V/12V/24V, Ampere 350mA/600A
PCF8574	1 unit	I2C Interface
Arduino keypad 4x4	1 unit	24Vdc, 30mA
ESP32 WROOM-32	1 unit	Power 3,3 VDC



(a)

(b)

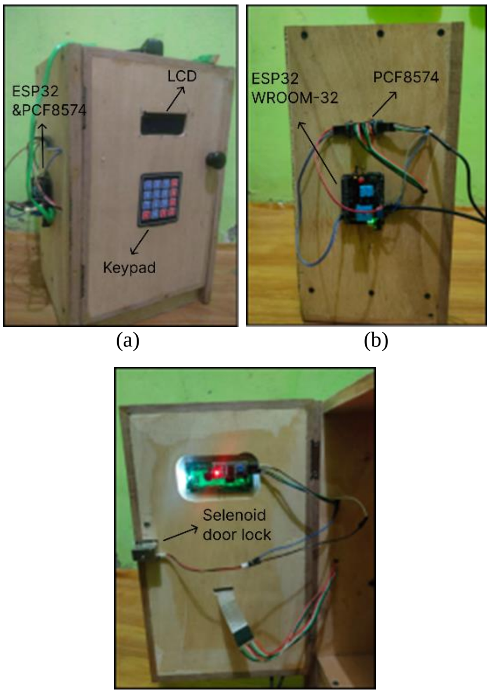
Figure 5. (a) Lockers Design, (b) Lockers System Circuit

The lockers are made using wooden blocks with a height of 40cm, width of 20cm, and length of 23cm. LCD and Keypad are placed on the front door. PCF8574 and ESP32 are placed on the outer right side. And selenoid door lock is placed on the inside door of the locker.

3. Results and Discussions

One of the features contained in the application is the locker-based online research book lending feature which needs to be applied in previous studies [5]-[8]. When run, the locker which has accessed to internet, then will send a key code to Firebase (Database in the cloud that will store key code) so the application can access it. The locker key code is sent according to the algorithm in Figure 3. The results of the circuit design implementation are shown in Figure 6.

After the locker successfully sends the locker key code to Firebase, the application can access the key code that stored in the Cloud and to be displayed to the user. The application provides a borrowing form that the user must fill in to fulfill the requirements for borrowing research books. After the form is filled in, the application will provide detailed borrowing information to the user where there is a locker key code. The locker key code provided will match the locker key code sent by the locker system on Firebase. By implementing this IoT system, it produces a locker system that is different from [25], which also implements a locker-based digital library, where the locker system in this study is not used as a secret storage place but only a place to store books that have been borrowed by librarians and can be accessed by anyone who has a locker key code. The following is the borrowing details page and locker key code page in the application shown in Figure 7.

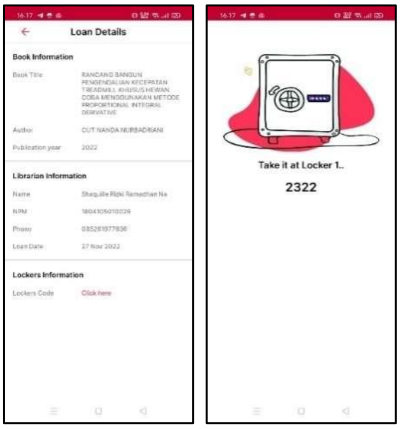


(a)

(b)

(c)

Figure 6. Result of Locker Implementation. (a) Entire Locker Section, (b) Side of the Locker, (c) Inside Locker



Loan Details

Book Information

Book Title: RANCANG BANGUN PENGENDALIAN KECEPATAN TREADMILL KHUSUS HEWAN COBA MENGGUNAKAN METODE PROPORTIONAL INTEGRAL DERIVATIVE

Author: CUT NANDA NURBADRIANI

Publication year: 2022

Librarian Information

Name: Shaquille Rizki Ramadhan Na

NPM: 1804105010026

Phone: 085261977836

Loan Date: 27 Nov 2022

Lockers Information

Lockers Code: Click here

Take it at Locker 1..

2322

(a) (b)

Figure 7. (a) Loan Details Page, (b) Locker Key Code Page

To find out if the application made has been running as planned and in accordance with the wishes of the user, 3 stages of testing of the application are carried out, namely white-box [26], black-box [27], and usability [28]. White-box testing tests the suitability of the input flow that enters the ViewModel module before being displayed on the application screen by analyzing the code structure in the ViewModel. Then, black-box testing is continued by observing the suitability of input and output on each application feature without knowing the code structure. The results of white-box and black-box testing are listed in Table 4 and Table 5.

Table 4. White-box Testing Results

Name of Controller	Total Test Case	Status
RegisterViewModel	1	Test Passed ✓
LoginViewModel	2	Test Passed ✓
CupBoardViewModel	1	Test Passed ✓
DetailViewModel	7	Test Passed ✓
FavoriteViewModel	1	Test Passed ✓
LoanFormViewModel	2	Test Passed ✓
HomeViewModel	2	Test Passed ✓
SearchViewModel	1	Test Passed ✓
SettingsViewModel	2	Test Passed ✓

Table 5. Black-box Testing Results

No	Function Code	Total Test Case	Status
1	Registration/T-F-001	3	Test Passed ✓
2	Login/T-F-002	3	Test Passed ✓
3	View the list of research books /T-F-003	1	Test Passed ✓
4	View research book details /T-F-004	1	Test Passed ✓
5	Borrow a research book / T-F-005	2	Test Passed ✓
6	View borrowed books /T-F-006	1	Test Passed ✓
7	View loan details /T-F-007	1	Test Passed ✓
8	See locker code/T-F-008	1	Test Passed ✓
9	Delete loan history /T-F-009	2	Test Passed ✓
10	Adding bookmark/T-F-010	1	Test Passed ✓
11	View a list of bookmarks /T-F-011	1	Test Passed ✓
12	Removing bookmarks /T-F-012	1	Test Passed ✓
13	Search for research books/T-F-013	2	Test Passed ✓
14	Change usename/T-F-014	1	Test Passed ✓
15	Change password/T-F-015	1	Test Passed ✓
16	Logout/T-F-016	2	Test Passed ✓

The white-box and black-box testing results show that the application has been validated and can run as planned in this study. Furthermore, usability testing is carried out using the System Usability Scale (SUS) [29], where 10 users who have used the application are given several questions related to the satisfaction of using the application that has been made, as shown in Figure 8.

Based on the Likert scale [30], the respondents' scores obtained from filling out 10 System Usability Scale questions will be processed to determine the calculated value of each respondent. The total score of the respondents obtained is 940, and the average score on SUS is 94. Based on the interpretation of SUS values, the average score on SUS shows that the software that has been made is included in the very feasible interpretation category.

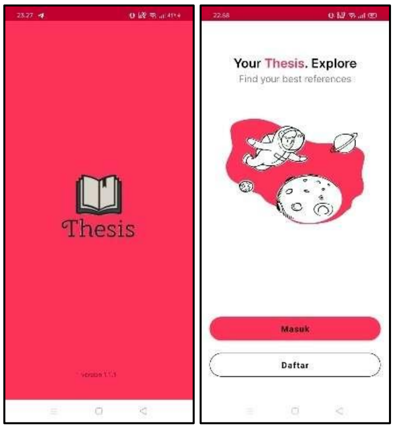


Figure 8. Result of Application Interface Display

4. Conclusion

This study is based on creating a mobile application with a locker-based online lending feature in the library. As a result, the application has a locker-based online lending feature and several other features as the user desires. The locker-based online lending system integrated with ESP32-WROOM-32 can connect with Firebase storage and send locker key codes to Firebase so the application can access them. Through experiments and tests conducted on the application, it is seen that the application can access the locker key code and display it to the user. The application has also been validated and can be accepted by users with a feasible interpretation category. One prototype locker was used in this study, and the application was only created for research book search users. In reality, the application also needs to be created for library staff to update the latest research book data and provide information on the location of the borrowed research book in the locker if there is more than one locker in use. Therefore, the whole system became more complex, resulting in a better system.

References

[1] D. Begum and Md. H. Elahi, “Digital library services to support online learning amid COVID-19: a study of a private

DOI: https://doi.org/10.29207/resti.v7i4.5047

university library in Bangladesh," *DLP*, vol. 38, no. 3, pp. 332–345, Jun. 2022, doi: 10.1108/DLP-03-2021-0025.

[2] S. Hussaini, M. Haruna, and A. Muhammad, "Application and Use of ICT in University Library Services: A Panacea for Conventional Library Services," *JOITS*, vol. 7, no. 1, pp. 1–8, Dec. 2020, doi: 10.46610/JOITS.2021.v07i01.001.

[3] "The next step in librarianship: Is the traditional library dead? The next step in librarianship: Is the traditional library dead?" https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1638&context=libphilprac (accessed Nov. 22, 2022).

[4] A. Rahimi, M. R. Soleymani, A. Hashemian, M. R. Hashemian, and A. Daei, "Evaluating digital libraries: a systematised review," *Health Info Libr J*, vol. 35, no. 3, pp. 180–191, Sep. 2018, doi: 10.1111/hir.12231.

[5] M. M. Aristov, J. W. Moore, and J. F. Berry, "Library of 3D Visual Teaching Tools for the Chemistry Classroom Accessible via Sketchfab and Viewable in Augmented Reality," *J. Chem. Educ.*, vol. 98, no. 9, pp. 3032–3037, Sep. 2021, doi: 10.1021/acs.jchemed.1c00460.

[6] T. Iqbal and M. Wali, "IDOL: Retrofit-Kotlin Service-Based Online Digital Library Application and College Open Data Repository," *ijsecs*, vol. 2, no. 1, pp. 1–8, May 2022, doi: 10.35870/ijsecs.v2i1.760.

[7] A. R. Dinazzah and R. Rahmi, "Mobile Library Application in Indonesia's Digital Libraries," *j.of education technology*, vol. 6, no. 1, p. 149, Mar. 2022, doi: 10.23887/jet.v6i1.43575.

[8] K. Rivo and M. Žumer, "Academic Libraries and Use of Mobile Devices: Case Study of Slovenia," *The Journal of Academic Librarianship*, vol. 48, no. 3, p. 102507, May 2022, doi: 10.1016/j.acalib.2022.102507.

[9] L. Suryani *et al.*, "Implementation of Maggot Cage Temperature and Humidity Control Using ESP8266 Based On the Internet of Things," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 6, no. 5, pp. 877–882, Nov. 2022, doi: 10.29207/resti.v6i5.4502.

[10] Izhangghani, I. Hikmah, and Slamet Indriyanto, "Prototype of Body Temperature and Oxygen Saturation Monitoring System Using DS18B20 and MAX30100 Sensors based on IOT," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 6, no. 5, pp. 810–817, Oct. 2022, doi: 10.29207/resti.v6i5.4385.

[11] M. M and Finki Dona Marleny, "Platform Digital and Content Innovation to Increase Youth Interest in the Agricultural Sector," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 6, no. 2, pp. 281–288, Apr. 2022, doi: 10.29207/resti.v6i2.3924.

[12] M. Abdurohman, A. Herutomo, V. Suryani, A. Elmangoush, and T. Magedanz, "Mobile tracking system using OpenMTC platform based on event driven method," in *38th Annual IEEE Conference on Local Computer Networks - Workshops*, Sydney, Australia: IEEE, Oct. 2013, pp. 856–860. doi: 10.1109/LCNW.2013.6758523.

[13] M. I. S. B. Khairat, Y. Priyadi, and M. Adrian, "Usability Measurement in User Interface Design Using Heuristic Evaluation & Severity Rating (Case Study: Mobile TA Application based on MVVM)," in *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA: IEEE, Jan. 2022, pp. 0974–0979. doi: 10.1109/CCWC54503.2022.9720876.

[14] Dony Novaliendry and S. Andriani, "English Edugame Application for Childhood base on Android," *RESTI*, vol. 4, no. 1, pp. 187–192, Feb. 2020, doi: 10.29207/resti.v4i1.1622.

[15] B. Weder, J. Barzen, F. Leymann, and D. Vietz, "Quantum Software Development Lifecycle," in *Quantum Software Engineering*, M. A. Serrano, R. Pérez-Castillo, and M. Piattini, Eds., Cham: Springer International Publishing, 2022, pp. 61–83. doi: 10.1007/978-3-031-05324-5_4.

[16] J. Sa-ngiampak *et al.*, "LockerSwarm: An IoT-based Smart Locker System with Access Sharing," in *2019 IEEE International Smart Cities Conference (ISC2)*, Casablanca, Morocco: IEEE, Oct. 2019, pp. 587–592. doi: 10.1109/ISC246665.2019.9071664.

[17] Y. Bassil, "A Simulation Model for the Waterfall Software Development Life Cycle," *International Journal of Engineering*, vol. 2, no. 5, 2012.

[18] J. Dąbrowski, E. Letier, A. Perini, and A. Susi, "Analysing app reviews for software engineering: a systematic literature review," *Empir Software Eng*, vol. 27, no. 2, p. 43, Mar. 2022, doi: 10.1007/s10664-021-10065-7.

[19] P. Ragunath, S. Velmourougan, P. Davachelvan, S. Kayalvizhi, and R. Ravimohan, "Evolving A New Model (SDLC Model-2010) For Software Development Life Cycle (SDLC)," 2010.

[20] A. Garg, R. Kumar Kaliyar, and A. Goswami, "PDRSD-A systematic review on plan-driven SDLC models for software development," in *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India: IEEE, Mar. 2022, pp. 739–744. doi: 10.1109/ICACCS54159.2022.9785261.

[21] "ESP32-WROOM-32 Datasheet." https://www.alldatasheet.com/datasheet-pdf/pdf/1148026/ESPRESSIF/ESP32-WROOM-32.html

[22] A. C. Gheorghe, E. Stan, and I. Udroiu, "Electricity Consumption Measurement System Using ESP32," *The Scientific Bulletin of Electrical Engineering Faculty*, vol. 21, no. 2, pp. 23–26, Dec. 2021, doi: 10.2478/sbeef-2021-0017.

[23] S. S. R. Emmadi and S. Potluri, "Android Based Instant Messaging Application Using Firebase," vol. 7, no. 5, 2019.

[24] D. Zhu *et al.*, "Building a Three-Level User Experience (UX) Measurement Framework for Mobile Banking Applications in a Chinese Context: An Analytic Hierarchy Process (AHP) Analysis," *MTI*, vol. 6, no. 9, p. 83, Sep. 2022, doi: 10.3390/mti6090083.

[25] "PCF8574 Datasheet."

[26] "Library Rooms And Study Spaces."

[27] J. Singh, S. Bilgaiyan, B. S. P. Mishra, and S. Dehuri, Eds., *A Journey Towards Bio-inspired Techniques in Software Engineering*, vol. 185. in Intelligent Systems Reference Library, vol. 185. Cham: Springer International Publishing, 2020. doi: 10.1007/978-3-030-40928-9.

[28] B. H. Rambe, R. Pane, D. Irmayani, M. Nasution, and I. R. Munthe, "UML Modeling and Black Box Testing Methods in the School Payment Information System," vol. 4, no. 36, 2020.

[29] D. Kamińska, G. Zwoliński, and A. Laska-Leśniewicz, "Usability Testing of Virtual Reality Applications—The Pilot Study," *Sensors*, vol. 22, no. 4, p. 1342, Feb. 2022, doi: 10.3390/s22041342.

[30] D. Demirkol, "Sistem Kullanılabilirlik Ölçeğinin Türkçeye Çevirisi: SUS-TR," *uujss*, vol. 11, no. Eylül-2018, pp. 238–253, Jan. 2018, doi: 10.29217/uujss.495.

[31] L. South, D. Saffo, O. Vitek, C. Dunne, and M. A. Borkin, "Effective Use of Likert Scales in Visualization Evaluations: A Systematic Review," *Computer Graphics Forum*, vol. 41, no. 3, pp. 43–55, Jun. 2022, doi: 10.1111/cgf.14521.

DOI: https://doi.org/10.29207/resti.v7i4.5047