Analysis of Automation Testing Using Repeato for Functional Testing of the Yess Nutrition Application Based on Flutter

Muhammad Ari Rifqi [1*], *Sri* Endang Anjarwani [1], and *Ari* Hernawan [1]

[1] University of Mataram, Dept. of Informatics Engineering, Mataram, Indonesia

Abstract. Automated testing has the advantage of executing test cases faster than manual testing and has a higher accuracy rate because it can detect more defects in the application. Automated testing is also effective for regression testing performed when there is a fix or update, to ensure that the fix does not cause new bugs to appear in the system. Therefore, automated testing becomes essential to replace manual testing. Automated testing involves the use of testing tools or frameworks that can reduce the time required in the testing process. This paper reviews Repeato software as an automatic testing tool, where Repeato works based on Computer vision. Experiments were conducted to see the test steps and results of the application display rendering time using the Repeato tool. The advantage of Repeato lies in its ability to automate visualbased testing, which can save the time and effort required in manual testing. However, as is the case with other testing tools, Repeato also has its limitations and drawbacks. Repeatos may not be able to recognize visual elements that are very complex or have arbitrary patterns. Repeato can conduct two rounds of tests on the Yess Nutrition application within 216 seconds, which is equivalent to 3.6 minutes.

1 Introduction

The process of developing software is inseparable from the process of running a series of tests on the software. Testing is a process that aims to analyze software to identify differences between actual conditions and expected conditions, find failures or errors in applications, and evaluate the features in the software [1][2].

Software testing can be done by two methods: manual and automated. Manual testing involves testers performing tests directly without the use of tools [1][2]. The level of accuracy of the tester is a key factor in this method. On the other hand, automated testing uses tools in the form of automation testing tools and relies on test scripts. In automated testing, a test script is executed to compare actual test results with expected results that have been documented in that script. The selection of automated testing tools determines the accuracy of application functional testing [1].

During the testing process, there are many test cases to be carried out, reaching hundreds of cases. If the test is done manually, it will take a long time and the risk of human error is unavoidable. However, this issue can be resolved with automated testing. Automated testing has the advantage of executing test cases faster than manual testing and has a higher degree of accuracy because it can detect more defects in the application. Automated testing is also effective for regression testing performed when there is a fix or update, to ensure that the fix does not cause new bugs to appear in the system [3][4].

Choosing the right automated testing equipment is important in testing. These considerations should be tailored to the testing needs and human resources available within the project team. In addition, it is important to consider the advantages and disadvantages of each existing test equipment [3]. Especially functional testing of the application is very necessary because, with functional testing, it can be known whether the application works according to the identification of its needs and development plan or deviates from the initial plan [5].

In this study, we will analyze how to automate testing on the Yess Nutrition application built using the Flutter framework by utilizing the Repeato testing tool that works based on machine learning and computer vision developed by Stephan Petzl [6].

This research is unique because it examines how the Repeato testing tool, which uses computer vision, is used in testing Flutter and Android applications. No previous research has analyzed this tool, and it is not commonly used. The study takes a close look at the advantages and disadvantages of using Repeato for Flutter application testing. As a result, developers can make an informed decision about which testing tool to use, taking into account the quality of testing provided by Repeato.

2 Literature Review

Research conducted by Amalia and Cahyono entitled "Utilization Analysis of Playwright for Web-Based

* Corresponding author: 27.ararifqi@gmail.com

Application Testing Automation (Case Study: Network Management System) ". This study discusses the use of Playwright as the latest automated tool, conducted Playwright testing experiments, and analyzed using comparative methods with Selenium. It was concluded that Playwright is more suitable to be implemented by testers who are new to automation tests because of its ease of use. Selenium is easier to implement by testers who are familiar with automated testing and have reliable programming skills. These skills are needed for making test scripts and integrating them with other tools [1].

Research conducted by Barus and Siburian entitled "Comparative study of automated testing tools for Android applications". This study discusses a comparative study of automated testing tools on Android-based mobile applications using Selendroid, Calabash, and UI Automator. Experiments were conducted to find out the advantages and disadvantages of each tool. From the results of analysis and experimentation, the author recommends UI Automator as the best tool in terms of ease of installation and running test cases in an Androidbased mobile application testing activity [3].

Research conducted by Herlinda, Katarina, and Ambarsari entitled "Automation Testing Tool in Testing Tajweed Learning Applications on the Android Platform". This study discusses several objects tested in the test including About Feature, Mahraj Feature (Oral Cavity Mahraj, Throat Mahraj, Tongue Mahraj, and Lip Mahraj), Tajweed Feature (Tanwin, Mim Dead), Mad Law, Idgham Law, Qalqalah, and Quiz

Feature. In addition, there are also sound objects that need to be tested. In testing the Tajweed application, it was found that the time needed was around 326,128 seconds or an average of 26,128 seconds to 5 minutes with 121 testing steps. Of the 121 steps, there are 3 failures, such as the screen display that cannot adjust when used in Landscape mode, causing the application display to exit the smartphone screen. In addition, Katalon cannot detect sound objects such as tajweed, tanwin, and idgham. This is because Katalon does not provide a slide feature so that the screen cannot be moved, and screenshots on Device View are static. There are also sound objects that continue to run even after moving to another screen page, and if other sound buttons are activated, the sounds will overlap. This limitation occurs because of the large memory size, so the response time becomes slower. Therefore, White Box testing is required to analyze programming instructions in applications. Based on the analysis, it can be concluded that Katalon still has a drawback, that is, the test still depends on the connection of the smartphone, and the original device must remain activated and driven [4].

Research conducted by Hasim, et.al entitled "SerenityBased Automated Testing Framework and Jenkins Automation Build". This study discusses the implementation of automated testing on the Cashier application using the Serenity and Jenkins Automated Build frameworks. The use of the Serenity framework in this study shows that testing will become more effective and efficient compared to manual testing. In a series of experiments, the implementation of automated tests using Serenity resulted in shorter test times compared to manual tests. Further research aims to develop this system to be more reliable and flexible in dealing with special cases, to avoid non-technical factors [7].

The next research is a research conducted by Laily and Triage entitled "Implementation of Quality Assurance in Android-Based Ourticle Application Development". This research conducts automated testing through quality assurance to overcome human error in manual testing. The purpose of this research is to help companies avoid human error in application testing. The results obtained from testing the quality of the application using Quality Assurance automatically based on the ISO-9126 standard are 3 (good), so the Ourticle application is feasible to use without any bug/error problems [8].

The next research is a research conducted by Rambe entitled "Automated Testing of Mobile Application Using Black-Box Technique with Appium". This research conducted automated testing using Appium tools. This research states that testing that is done repeatedly and manually will consume a lot of time and resources that the development team needs. This problem requires a solution to make it easier for the development team to test more effectively and efficiently, such as automated testing tools. Appium is a testing tool that makes it easy for developers to perform tests automatically.

This study is intended to determine the use of Appium in testing mobile applications on the Jala mobile application. The research was conducted by conducting automation testing on the Jala mobile application using black box testing techniques and applying the STLC cycle. The results concluded that automated testing conducted using Appium was successful in finding defects and errors in the Jala mobile application and could help the development team save time and resources [9].

The next research was conducted by Kosasih and Cahyono with the title "System Design in Testing The Point Of Sale Application (Case Study:TPOS PT. JAVASIGNA INTERMEDIA)". This research conducted automated testing through the Catalog application. The purpose of this study is to analyze the effectiveness of applications that have been built, which then whether the Katalon Automation testing tool is more effective when compared to manual testing. This study describes the mechanism of automatic waiting to play the test command. The experimental results show that wait times can be determined automatically and dynamically so that testers do not have to add wait orders manually, reducing creation time and errors [10].

The next research was conducted by Rafiq, et.al with the title "Automated VS. Manual Testing: A Scenario-Based Approach Towards Application Development". In this study, various types of application testing that can be done automatically are discussed. This research also discusses various ways to conduct automation testing using tools. This study concluded that automated testing can save a lot of time and costs, and provide more reliable test results. This study also states that the

combination of manual testing and automation testing provides more accurate results [11].

Subsequent research regarding discussions related to the automatic testing of Android applications in iterations was carried out by Zhong, et. Al with the title "Iterative Android Automatic Testing". This study conducted iterative Android automation testing experiments or IAAT testing. This study proposes testing Android applications through an iteration system to be a solution to solving complex interactive widget problems that have the potential to hinder the logic of automated testing tools. The experimental results show that the average IAAT test coverage reached 37.83%, 13.98% higher than the average test coverage under the same test conditions and scenarios [12].

Subsequent research was conducted by Kavitha, et al with the title "A Comprehensive Automated Security Testing Tool for Flutter Applications". The background of this research is the increasing popularity and development of the Flutter application which of course is directly proportional to the increased risk of security vulnerabilities, which can threaten the privacy of users' sensitive data. The researcher developed an automated security testing tool for Flutter applications. Based on this research, the Flutter application has the potential to be tested using automated testing to improve application performance and security [13].

Subsequent research was conducted by Saputra and Stefanie with the title "Automation Testing API, Android, and Website Using Serenity Bdd Pada Hospital Management System Software". The research states that utilizing automated testing will save time and costs in testing compared to testing applications manually. This study also tested the appearance of applications and websites, as well as testing the API used. This study conducted testing experiments on seven test cases.

The results of a comparison of automatic and manual testing in terms of time are automation testing faster than manual testing, one of the test results in the first test case, where automation testing produces 46.38 seconds, while manual testing is 68.94 seconds [14].

The following related research was conducted by Zulianto et al. under the title "Utilization of Katalon Studio for BlackBox Testing Automation on iPosyandu Application". This study aims to enhance the performance of the iPosyandu application through a testing process. The applied testing method is automated testing. The testing tool employed is Katalon Studio, utilized for automating the Black-Box testing process. The purpose of this testing is to minimize testing procedures that are unfeasible to conduct manually. Automated testing is also carried out to mitigate human errors. The test cases utilized in conjunction with Katalon Studio involve recording and playback. These test cases share similarities with Repeato, which also employs the record and playback approach. This automated testing is then compared with manual testing, revealing comparable outcomes, albeit with a shorter testing duration. Across the 13 test cases examined, there was a notable improvement in execution speed, reducing from 719.27 seconds in manual testing to 283.08 seconds. Consequently, this represents a speed enhancement of 2.54 times. The integration of Katalon Studio proves highly beneficial for both effectiveness and reducing human errors in the manual testing process. Access to testing outcomes is facilitated through Katalon TestOps, provided by Katalon Studio. Katalon TestOps presents testing results data in various formats, including csv, xlsx, and pdf formats. This parallels the feature in Repeato that furnishes result reports in PDF format [15].

3 Methodology

In this section, the stages carried out in automation testing analysis research are explained with the aim that this research can be completed systematically, purposefully, and clearly. The stages of research are illustrated in the flowchart in Fig.

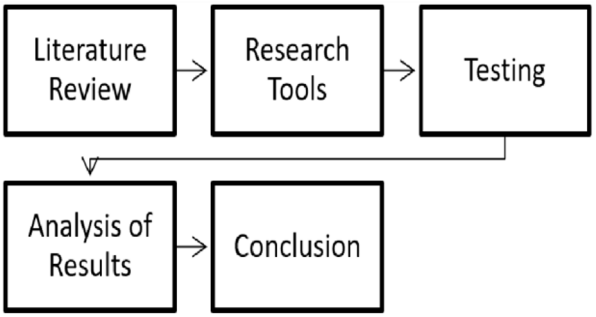

Fig. 1. Research Flowchart A.

3.1 Literature Review

The literature review stage aims to collect and analyze various relevant sources from various scientific journals. The literature study conducted is to analyze research related to automation testing and how the process is carried out in automation testing.

3.2 Tools Research

The Tools Research Stage aims to conduct research and find out automation testing tools for Flutter and mobile applications. Through this stage, we can find tools that can be used to automate the testing of Flutter-based and mobile applications. One of the research tools is Repeato which has not been widely used and has not been widely analyzed. With this research, it is expected that readers and application developers who will test mobile applications can see the advantages and disadvantages of Repeato in conducting automation testing. In this study, Repeato version 1.3.7 was used. It should be noted, currently, Repeato is quite frequent to update and always fix bugs found or reported by users.

3.3 Testing

At this stage, researchers will carry out the process of testing the Yess Nutrition application using the Repeato tool. The way Repeato works in doing automation testing is by connecting a real device (Smartphone with Android Operating System) and a computer that has Repeato installed via a USB cable. After that,

researchers open and activate Repeato and make sure the real device is connected.

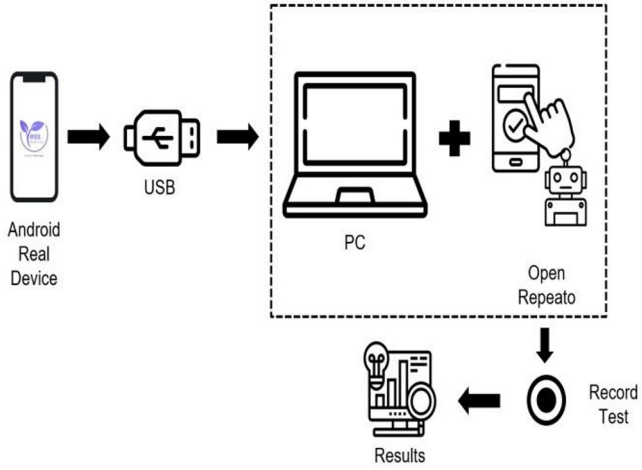

Fig. 2. How to Test with Repeato

To use Repeato, simply follow the steps shown in Fig. 2. Connect your smartphone to your laptop using a USB cable, and open the application you want to test on your phone. In our research, we used an app called Yess Nutrition, which was developed by the researchers using the Flutter framework. On your laptop, log in to Repeato. It's important to know that Repeato uses Computer Vision technology to evaluate both the functionality and visual interface of the application, including examining elements within the app to ensure they're working correctly. Once you're ready, start recording your testing session. Explore the app's various features and functions, then replay the recorded session within Repeato. This tool will automatically evaluate the app's visual aspects and functionality.

The application testing system in this study uses iterationbased testing which is divided based on the features in the application.

Here's Fig. 3 the initial view of Repeato when the real device has been successfully connected.

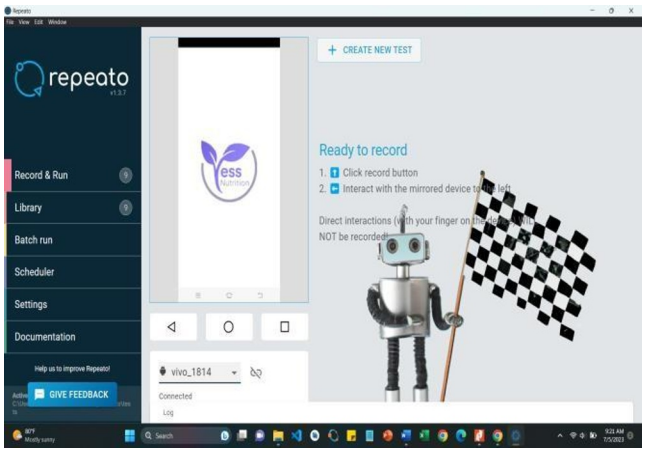

Fig. 3. Interface of Repeato

After successfully connecting a real device with Repeato, researchers conduct testing by recording the test activity to be carried out. After that, researchers try and test one by one the features of the application to see more detailed results for each test unit. When you're done trying out the app's features. Researchers press the save button and run automation testing on Repeato. Repeato works with computer vision-based, so it can test the appearance of the application intelligently and can record test time, and display test errors in the event of a test failure.

3.4 Analysis of Results

At this stage, analysis of application test results using the Repeato tool is carried out. At this stage, the advantages and disadvantages of the Repeato tool in conducting automatic testing of Flutter-based applications will also be discussed. This research will also show the duration used when performing automatic testing of flutter-based applications using Repeato.

3.5 Conclusion

The conclusion stage of automated testing is the final part of the testing process that involves analyzing the results of automated tests that have been carried out. At this stage, the collected test data and metrics are thoroughly evaluated to evaluate the quality and feasibility of Repeato as an automation testing tool. The results of this analysis are used to formulate conclusions about the success, failure, and performance of Repeato, as well as to compile reports containing recommendations for improvement or necessary actions based on the results of automated testing.

4 Result and Discussion

In this test, automation testing was carried out using computer vision-based tools called Repeato. The use of automation testing aims to avoid subjectivity in manual testing. Automation testing is also able to avoid large costs and time on manual testing, Automation testing shifts existing manual or conventional testing. Automation testing no longer involves human intervention [3]. The way this tool works is to connect a real device to a PC using USB and open the application, then the tool will be intelligently able to run unit testing of the application. Here are some results of automated testing using Repeato:

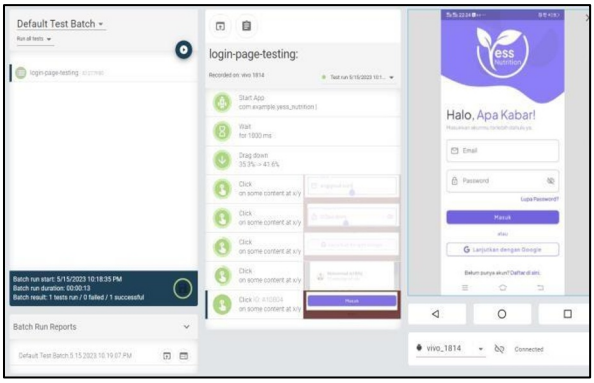

Fig. 4. Automated Testing Using Repeato

In Fig. 4 it can be seen that Repeato can run existing test scenarios in detail on each view in the application. Repeato also displays the duration of the test so that

application developers can see if the developed application has a fast or slow display rendering time. Repeato provides features to add test scenarios or delete test scenarios if needed. On the test results page, Repeato will display detailed images of the part of the application being tested.

Repeato also provides a feature to generate test document reports, but application developers must first save the test scenarios performed. In the event of a test failure, Repeato will provide information stating that the function of the tested application failed and will be shown the part of the application that has failed, especially in terms of rendering the appearance of the application.

Fig. 5. Summary of Feature Test Results Using Repeato

In Fig. 5, the time when the test was created will also be displayed. This is useful for observing the testing process, allowing developers to document each test sequentially and in a structured manner. Furthermore, the time when the testing is executed can also be seen, enabling developers to create test scenarios only once and run them multiple times. Subsequently, developers can review the testing history for steps of improvement and performance enhancement in the application. The package name of the application's source code file will also be shown, serving the purpose of identifying each tested application, thus preventing developers from confusion when documenting test results.

Repeato will also display the version of the application package, showcasing the version of the ongoing development of the application. The duration of the testing process will also be shown, allowing developers to observe the time required to run the testing using Repeato. Installation and application update times will also be displayed, along with the physical device on which the application is executed, as informed by Repeato. If all test scenarios are successfully executed, at the bottom, Repeato will indicate the number of test steps and the obtained results.

It can be seen in the summary of application test results using Repeato that the duration of testing for each feature is found. The following is the time or duration of application testing which consists of two stages of application testing iterations:

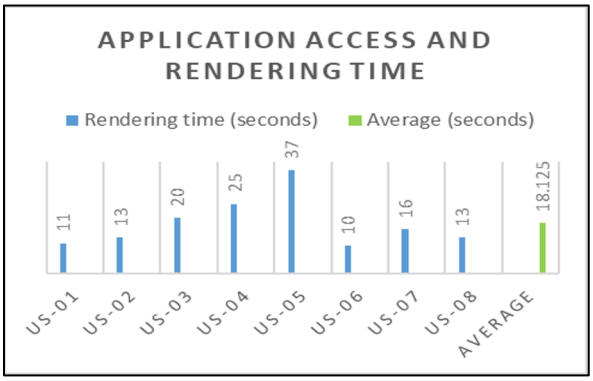

Fig. 6. Automation Unit Testing Duration Iteration 1

Fig. 6 shows the duration of the first automated testing iteration on the Yess Nutrition application. In the first iteration's test case, testing was conducted on 8 user stories: sign up feature took only 11 seconds, log-in took 13 seconds, profile setup took 20 seconds, home page loading took 25 seconds, prevention of stunting education took 37 seconds, monitoring daily macro-nutrient intake took 10 seconds, meal scheduling took 16 seconds, and prenatal health education took 13 seconds. The timings were obtained by examining the test run duration as shown in Fig. 5. The total time taken to execute the application's testing playback was 145 secon18ds, with an average time of 18.125 seconds.

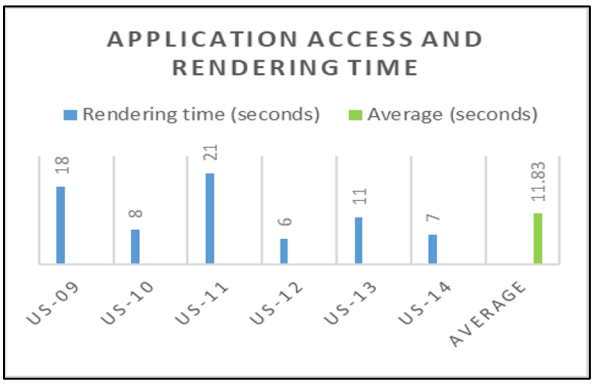

Fig. 7. Automation Unit Testing Duration Iteration 2

Furthermore, Fig. 7 shows the duration of the second automated testing iteration on the Yess Nutrition application. In the first iteration's test case, testing was conducted on 6 user stories: the nutrition information feature took 18 seconds, Body Mass Index (BMI) Check took only 8 seconds, Calorie Needs Check took 21 seconds, the village health worker/community health center page took 6 seconds, food recipes and complementary feeding (MPASI) took 11 seconds, and early stunting detection took 7 seconds. The total time taken to execute the application's testing playback was 71 seconds, with an average time of 11.83 seconds.

Based on the two test case iterations conducted, it is observed that the testing time is short and does not require a significant time investment, thus not incurring substantial costs. The total testing time obtained was 216 seconds, equivalent to just 3.6 minutes. With automated testing, the testing time and costs can be significantly reduced compared to manual testing. Additionally, automated testing provides

comprehensive and structured documentation of the test results.

Repeato is software used for automated testing in the context of mobile applications. This software is specifically designed to perform Computer Vision-based testing, which means Repeato can recognize visual elements in applications automatically.

With Repeato, users can record test scenarios by capturing screenshots of applications and identifying important elements such as buttons, text input, or images. Then, users can specify the testing steps that Repeato must perform, such as clicking buttons, filling out forms, or verifying display results [6][16].

Repeato uses Computer Vision capabilities to compare the actual view of an application with an expected or pre-recorded view. Thus, Repeato can detect discrepancies between actual and expected conditions, as well as identify failures in applications [6]. The advantage of Repeato lies in its ability to automate visual-based testing, which can save the time and effort required in manual testing. In addition, Repeato can also be used for regression testing, where changes or updates in the application can be tested automatically to ensure that no new bugs appear.

However, as is the case with other testing tools, Repeato also has its limitations and drawbacks. Repeatos may not be able to recognize very complex visual elements or have rapidly changing patterns. In addition, the use of Repeato also requires knowledge of the right settings and configurations to optimize test performance.

Overall, Repeato is an automated testing software that leverages Computer Vision to automate mobile application testing. With the ability to recognize visual elements and compare actual views to expected views, Repeato can help improve the efficiency and effectiveness of mobile application testing.

Conclusion and Recommendation

Repeato is a software that automates mobile application testing using Computer Vision technology. Repeato can conduct two rounds of tests on the Yess Nutrition application within 216 seconds, which is equivalent to 3.6 minutes. Repeato also has limitations and disadvantages. Repeatos may not be able to recognize visual elements that are very complex or have arbitrary patterns. In addition, the use of Repeato also requires knowledge of the right settings and configurations to optimize test performance. With the ability to recognize visual elements and compare actual views to expected views, Repeato can help save application testing costs and time.

To conduct future research, different types of testing can be used such as functional, performance, usability, security, integration, and regression testing. There are various testing tools that can be utilized for this purpose. Moreover, future studies can compare testing experiments using various automated testing tools.

Thanks to Stephan Petzl as CEO of Repeato (https://www.repeato.app/about/) who has developed Repeato so that it has many benefits for mobile application testing.

References

1. A. Amalia and A. B. Cahyono, “Utilization Analysis of Playwright for Web-Based Application Testing Automation (Case Study: Network Management System),” *Automata*, 2022, [Online]. Available: https://journal.uii.ac.id/AUTOMATA/article/view/21896%0Ahttps://journal.uii.ac.id/AUTOMATA/article/download/21896/12036
2. Ministry of Environment and Forestry, “Uji Emisi Kendaraan Sebagai Bentuk Kontribusi Masyarakat Terhadap Pengendalian Pencemaran Udara -,” 2021. https://www.menlhk.go.id/site/single_post/4078 (accessed Apr. 12, 2022).
3. A. C. Barus and S. Leo, “Android Comparative Study of Automated Testing Tools for Android,” *JTIIK J. Teknol. Inf. dan Ilmu Komput.*, vol. 6, no. 6, pp. 645–654, 2019, doi: 10.25126/jtiik.20196953.
4. H. Herlinda, D. Katarina, and E. W. Ambarsari, M.Kom., “Automation Testing Tool in Testing Tajweed Learning Application on the Android Platform,” *STRING (Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 4, no. 2, p. 205, 2019, doi: 10.30998/string.v4i2.5285.
5. R. Purbaningtyas, “Implementation of Functional Testing in Feasibility Testing of the Smart Malnutrition Detection Mobile Application,” Techno.Com, vol. 18, no. 3, pp. 251–263, 2019, doi: 10.33633/tc.v18i3.2504.
6. S. Petzl, “Manual Testing vs Automation Testing: what is better and when to use it?,” *Repeato: Computer Vision Mobile Testing*, 2022. https://www.repeato.app/manual-vs-automated-testing/ (accessed Jun. 06, 2023).
7. J. Akhmad *et al.*, “Serenity-based Automated Testing Framework with Jenkins Automated Build,” *JUTI*, vol. 19, no. JUTI: Jurnal Ilmiah Teknologi Informasi, pp. 102–110, 2021.
8. D. Y. Laily, U. Islam, and N. Sumatera, “Implementation of Quality Assurance in Android-Based Ourticle Application Development,” *Sibatik J.*, vol. 2, no. 3, pp. 793–804, 2023.
9. A. R. Rambe, “Automated Testing of Mobile Application Using BlackBox Technique with Appium,” Universitas Islam Indonesia, 2022.
10. Y. Kosasih and A. Budi Cahyono, “System Design in Testing The Point Of Sale Application (Case Study: TPOS PT. JAVASIGNA INTERMEDIA),” *Automata*, vol. 2, no. 1, pp. 24–30, 2020.
11. M. Rafiq, R. Ashraf, and H. Abid, “Automated VS . Manual Testing : A Scenario Based Approach Towards Application Development,” *Gyancity J.*

Electron. Comput. Sci., vol. 5, no. 1, pp. 47–55, 2020, doi: 10.21058/gjecs.2020.51006.

12. Y. Zhong, M. Shi, Y. Xu, C. Fang, and Z. Chen, "Iterative Android automated testing," *Front. Comput. Sci.*, vol. 17, no. 5, pp. 1–12, 2023, doi: 10.1007/s11704-022-1658-8.
13. V. Kavitha, S. Kishore, S. K, and G. G, "A Comprehensive Automated Security Testing Tool for Flutter Applications," *Int. J. New Innov. Eng. Technol.*, vol. 22, no. 1, pp. 167–172, 2023.
14. B. D. Saputra and A. Stefanie, "Automation Testing of API, Android, and Website Using Serenity BDD in Hospital Management System Software," *J. Ilm. Wahana Pendidik.*, vol. 2023, no. 10, pp. 114–126, 2023, [Online]. Available: https://doi.org/10.5281/zenodo.7983405.
15. A. Zulianto, A. Purbasari, N. Suryani, A. I. Susanti, F. R. Rinawan, and W. G. Purnama, "Utilization of Katalon Studio for Black-Box Testing Automation on iPosyandu Application," *J. Edukasi dan Penelit. Inform.*, vol. 7, no. 3, p. 370, 2021, doi: 10.26418/jp.v7i3.46954.
16. S. Petzl, "The Basics of Flutter Testing," *Repeato: Computer Vision Mobile Testing*, 2023. https://www.repeato.app/flutter-testing-basics/