Business Excellence and Management
Volume 12 Issue 1 / March 2022

FRONT-END APPLICATION FOR MULTIPLE STOREFRONTS ECOMMERCE USING CROSS-PLATFORM TECHNOLOGY

I Eng KHO
Swiss German University, Tangerang, Indonesia
le.kho@sgu.ac.id

Ahmad Windardi ALIYAZIS
Swiss German University, Tangerang, Indonesia
ahmad.aliyazis@student.sgu.ac.id

Maulahikmah GALINIUM
Swiss German University, Tangerang, Indonesia
maulahikmah.galinium@sgu.ac.id

Abstract
Cross-platform framework is becoming more and more popular. Many giant tech companies have their own offering with different programming languages. With so many choices, sometimes it is difficult for a start-up developer to choose which framework will they use for their project. Therefore, this research aims to give insight for start-up developers on which cross-platform framework is better between Flutter and React Native by comparing their performance. This research achieved its goal by creating two identical mobile applications using Flutter and React Native and then comparing their performance using a theoretical framework that measures load time, average frame rate, and memory usage. The result of this testing is that Flutter has a shorter load time at 1.69 second compared to React Native at 4.26 second and lower memory usage at 6.52MB compared to React Native at 27.6MB. Their average frame rate is very comparable hovering around 60 frames per second. With faster load time and lower memory usage Flutter has better performance wise compared to React Native.
Keywords: Cross-Platform, Mobile Application, Flutter, React Native, Performance testing
DOI: https://doi.org/10.24818/beman/2021.12.1-07

1. INTRODUCTION

From 2017 to early 2019, Tokopedia's front end application was built using React Native. They used it because of its Hot Reload features so they do not have to recompile the whole application when they are working on the front end. They are reverting back to using Flutter development framework because they received many feedbacks from users that their Homepage rendering is not smooth, and it

is terrible for User Experience (Compfest, 2019). With React Native it took 1.3 seconds for the page to load, but it only took 0.6 seconds using Flutter.

According to JetBrains Developer Ecosystem 2020 survey, most of the respondents (42%) voted for React Native as their cross-platform mobile framework of choice. Second place is Flutter at 39% and third place is Cordova with a large gap at 18% (JetBrains, 2020). Flutter's approach on user interface generation is a widget. The widgets are beautiful and highly customizable because they are based on Material Design. Flutter has a unique way of rendering its components by using its in-house lightning-fast rendering engine instead of relying on the device's OEM widgets or using web views. Facebook's own offering in the cross-platform framework market is React Native. React Native was developed from Facebook's own React framework. It brings modern web techniques to mobile despite being largely written in JavaScript and operated on JavaScript core. It uses native interface to access native hardware such as storage, speaker, and camera. This research is using Flutter and React Native because they are currently the top 2 cross-platform mobile frameworks according to JetBrains.

Mobile Application should have great performance to satisfy customers' User Experience. This statement is supported by Tokopedia's case when their customers are complaining about the decline of the application performance when they are opening the application. With many choices of cross-platform framework out there, start-up developers need insight on better framework.

The objective of this research is to find which cross-platform technology is better to develop front-end application in terms of having a shorter load time, better average frame rate, and lower memory consumption between Flutter and React Native. In this application development, there will be two identical applications using multiple storefronts approach. Multiple storefronts let admin manages many stores with one sitting. One is using Flutter and the other one is using React Native. It will follow a specific design for its UI that requires uniformity of multiple platforms. The performance, that will be measured, are load time, average frame rate, and memory usage. The scope of the research is developing an online marketplace front-end application with multiple storefronts for Android mobile application and iOS application that meet its business requirements using Flutter and React Native and then comparing their performance by measuring their load time, average frame rate, and memory usage.

2. LITERATURE REVIEW

Generally, the application performance is measured based on various performance parameters, one of them is response time (Chatterjee et.al, 2016). The indicator of response time that can be observed is the smoothness of an application, defined by how many frames per second it can sustain. The ideal frame per second is around 60fps (Jagiello, 2019). Two identical applications were

Business Excellence and Management
Volume 12 Issue 1 / March 2022

developed using React Native and Flutter. Wu (2018) aims to give insights on the positives and negative aspects of developing an application in Flutter and React Native. One of the examples is the applications performance. The performance was measured by measuring how many frames are rendered per second on a device. Frame per Second (FPS) has been accepted as a standard unit to describe the fluentness or responsiveness of an application (Wu, 2018).

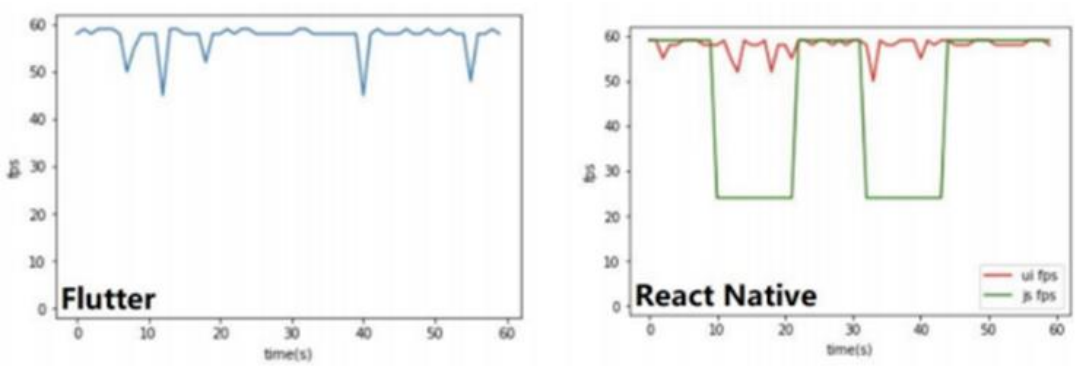


FIGURE 1. FPS COMPARISON BETWEEN FLUTTER AND REACT NATIVE

Source: Author

Performance of Flutter and React Native based on list scrolling is similar on the surface. The average fps while scrolling has been over 60 for both of them. However, React Native FPS in the JavaScript thread has many significant drops as shown in Figure 1 (Wu, 2018). React Native only renders a few items at once to reduce memory usage. If a user is scrolling swiftly, React Native will display blank squares as a placeholder while the fps in JavaScript thread is tanking. Meanwhile, Flutter does not have this problem and its fps is very stable. Table 1 shows the comparison between Wu's work and this research. Real devices and real case study are used in this research.

TABLE 1. COMPARISON BETWEEN WU AND THIS RESEARCH

	Wu's Work	**This Research**
Testing Platform	Emulator	Real devices
Testing Procedure	Scrolling aimlessly	Performing a real use case

Source: Author

Akamai Technologies (2015) discusses consumer expectation on an e-commerce performance wise. For example, Akamai Technologies – 2014 Consumer Web Performance Expectation Survey showed that 49% of respondents expect time to load a page in under 2 seconds regardless of device. This result differs greatly from five years ago which 63% of consumers would choose to wait patiently

for the page to load instead of leaving. From the same survey, 50% of consumers will abandon the website and leave for another website to get what they want.

Dalmasso (2013), Latif (2016) and Rahul Raj & Tolety (2012) conducted a testing with multiple mobile cross-platform framework. PhoneGap, Sencha Touch 2.0, and a bit old Application Craft. Albeit, the performance metrics that they used is quite good. They tested memory usage, CPU usage, and battery consumption. Memory usage is important to test because lower memory usage will allow other applications to be ran simultaneously.

Based on several related works, Figure 2 is created as a framework to measure the performance of a cross-platform application by combining several performance metrics from different related works. Load Time should be swift because according to Akamai Technologies (2015) today's consumers need low load time, preferably under 2 seconds. And it has been proven by Tokopedia's case in 2019. According to Jagiello (2019) and Wu (2018), Frame Rate is an important part of mobile application performance metrics because it represents the input delay that the consumer feels. The higher the number, the lower the input delay which means consumer will have a smooth experience while using the application. Memory Usage is one of the performance metrics that Dalmasso (2013) used in his paper. It represents the flexibility that consumers expect while using an application. Meaning that consumer won't need to close other applications to use our application. By combining Load Time, Frame Rate, and Memory Usage, a performance metrics is created to decide whether a cross-platform framework is good enough performance wise.

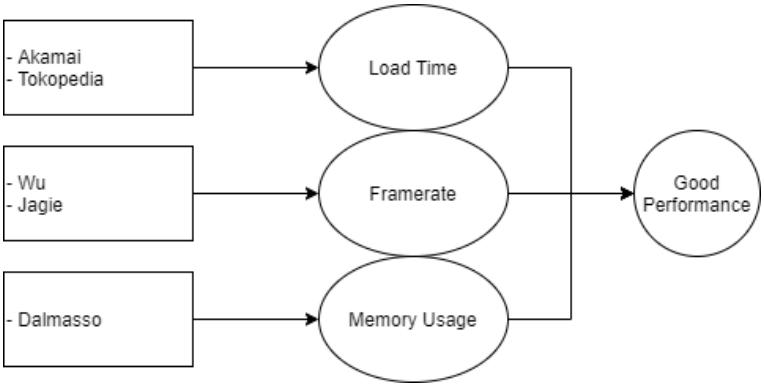


FIGURE 2. THEORETICAL FRAMEWORK FOR MEASURING PERFORMANCE

Source: Author

3. RESEARCH METHODS

Figure 3 is a research overview as a guideline, which comprises requirements gathering, designing the system, developing the product, testing the product, and finally deploying the product.

Business Excellence and Management
Volume 12 Issue 1 / March 2022

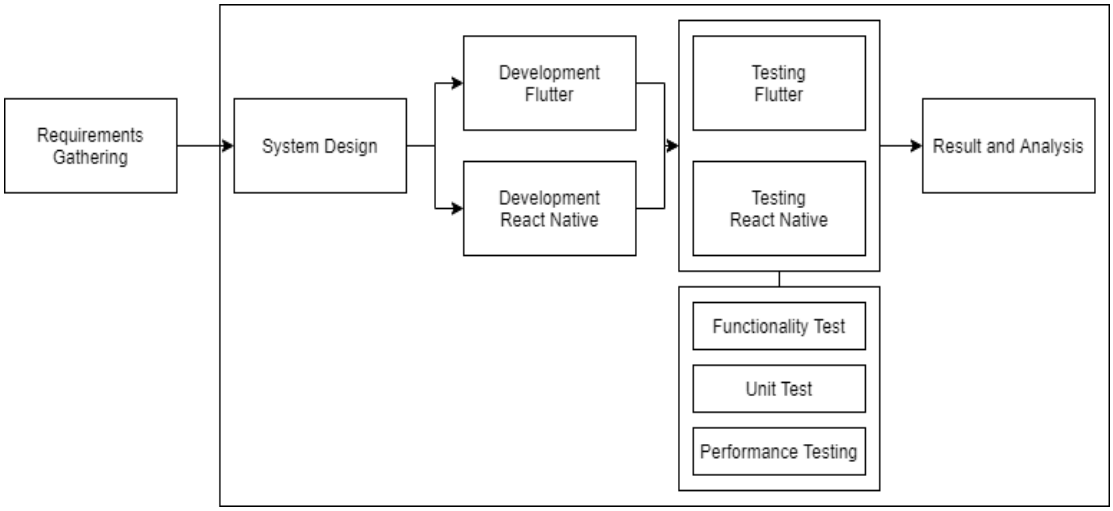
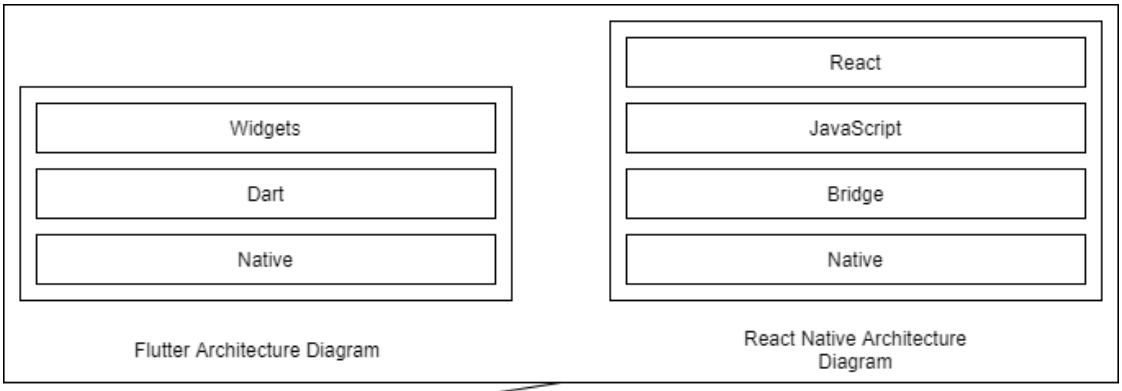


FIGURE 3. RESEARCH OVERVIEW

Source: Author

From the Architecture Diagram shown on Figure 4, the mobile application is running on their respective API platform on the Flutter framework and React Native framework. The HTTP request is handled on the backend by Request Handler and then Request Handler is running services depending on the request. Services are communicating with the Database Handler to get data from MySQL Database. Figure 5 also shows the use case of the developed application, which includes all the features done by admin and/or customers. The application case study is based on the reference by Livani et.al. (2020).

There are 2 application developments used in this research which are Flutter application and React Native application. Development environment for Flutter is using Android Studio while for React Native is using Visual Studio Code. The reference framework used for target development is Android Developers Guides on Device compatibility. Based on the distribution dashboard, this application is targeting xhdpi normal screen size or 1080p because that is the most common Android screen size and resolution based on data collected on 1-8th of January 2021. This application is also targeting minimum of Android 4.4 Kitkat or API 19 based on Android API Distribution chart because it has 98.1% distribution and still has features needed for the application.



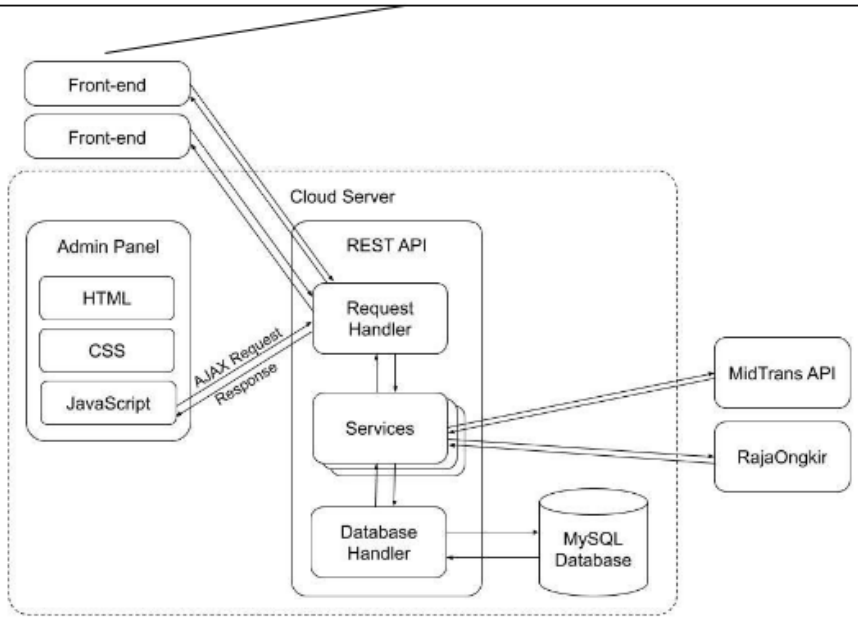


FIGURE 4. ARCHITECTURE DIAGRAM

Source: Author

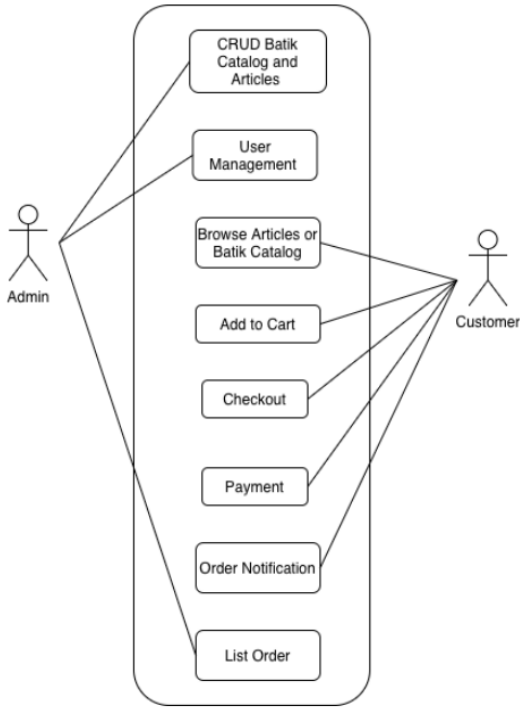


FIGURE 5. USE CASE DIAGRAM

Source: Author

Table 2 shows the unit testing where parts of functions are individually tested. In this research, unit testing is conducted manually to ensure everything runs well. While Table 3 describes the functionality testing where the functionality requirements of system is tested. The test is also conducted manually.

TABLE 2. UNIT TESTING SCENARIO

No.	Use Case Name	Details
HTTP: GET Listing all use cases that fetch data from the backend		
1	Browse Article or Batik Catalog	GET product list, article list, product detail, and article detail from backend
2	Checkout	GET payment method list, address list, and courier list from backend
3	Order Notification	GET order list and order details from backend
HTTP: POST Listing all use cases that send data to the backend		
1	Checkout	POST order to the backend

Source: Author

TABLE 3. FUNCTIONALITY TESTING SCENARIO

No	Use Cases	Features
1	Browse Article or Batik Catalog	Scroll through batik or article list
		Select individual batik or article
2	Add to Cart	Pressing add to cart button
3	Checkout	Selecting Address
		Selecting payment method
		Selecting courier
		Pressing checkout button
4	Order Notification	Scroll through order list
		Select individual order

Source: Author

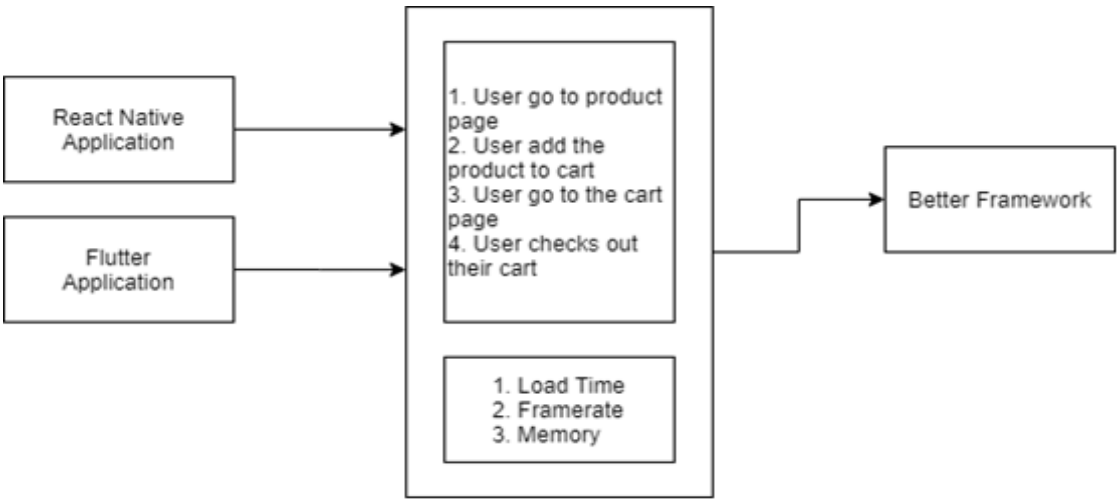


FIGURE 6. PERFORMANCE TESTING FRAMEWORK

Source: Author

Figure 6 shows the performance testing framework that is used in this research. First, both applications are given a set of task or scenario to complete. It resembles how a customer would behave when they want to buy a product. During the scenario, application performance is measured including load time, frame rate, and memory usage. After the results are get, a comparison will be conducted to see which cross-platform framework has better performance. Load time is measured by using a stopwatch to count how long it took for the app to start from a cold start. Frame rate is measured by using React Native and Flutter debugger during scenario. Memory usage is measured by looking at the phones task manager. Testing scenario is done on 5 different devices and repeated 5 times for each device per application. The devices that are used for performance testing according to Table 4, are Google Pixel 3a, Xiaomi Redmi Note 5 pro, Asus Zenfone Zoom S, Asus Zenfone 5 ZE620KL, and Asus

Business Excellence and Management
Volume 12 Issue 1 / March 2022

Zenfone 3 ZE552KL. Some of them might have different specs but all of them are on different conditions (e.g., number of applications running).

TABLE 4. LIST OF DEVICES FOR TESTING

Device	Chipset	GPU	RAM	Android Version
Google Pixel 3a	Snapdragon 670	Adreno 615	4GB	Android 11
Xiaomi Redmi Note 5 pro	Snapdragon 636	Adreno 509	4GB	Android 10
Asus Zenfone Zoom S	Snapdragon 625	Adreno 506	3GB	Android 8
Asus Zenfone 5 ZE620KL	Snapdragon 636	Adreno 509	4GB	Android 9
Asus Zenfone 3 ZE552KL	Snapdragon 625	Adreno 506	3GB	Android 7

Source: Author

4. FINDING AND RESULT

User Requirement Results

R.C. is a Chief Technology Officer at Xintesa. F.K. is a Software Engineer at Shopee and A.W. is an iOS Developer at Apple. Based on the Table 5, mobile developers do care about their applications performance. The most common answers on their top priority for application performance measures are memory usage, frame rate, and load time. They also put application performance as a factor on choosing which cross-platform framework they want to use.

TABLE 5. USER REQUIREMENTS INTERVIEW RESULT

Questions	R.C. from Xintesa	F.K. from Shopee	A.W. from Apple
Do you think that application performance is important?	I do think it is important as it affects user experience.	Application performance is very important, the benchmark of each language and framework would be a big decider on what to choose.	Yes.
What should be counted as application performance parameters?	Something that can be used to measure application performance objectively.	Memory usage, business logic, rendering speed.	Cust. Satisfactory, error rate, application response, throughput, security, mem usage, time delay.
What are the 3 most important application performance parameters?	Load time, memory usage, frame rate.	CPU and memory intensive benchmark (Gauss-Legendre or Borwein algorithm), frame rate, how fast it interacts with phone API.	Mem usage, time delay, throughput.
Why do you choose	Load time because	Because it's scientific,	Because I already

those 3 as the most important application performance parameters?	customers hate waiting, memory usage so that customers can multitask and frame rate because it affects response time and the overall smoothness of the application.	and also, I would like to have a smooth experience while using the app.	research about the similar area as an iOS developer in order to build mobile app based on Questionnaire.
Do you think application performance can be used as a factor to know cross-platform framework is good or not?	I do think application performance can be used as a factor to help me choose which cross-platform framework I want to use.	Yes, it should be a factor.	Yes.

Source: Author

5. DISCUSSION

Performance testing result of React Native mobile application has average load time of 4.26 seconds, average frame rate of 59.8 frames per second and average memory usage of 27.6MB, as shown in Table 6 based on this paper's performance testing scenario.

TABLE 6. REACT NATIVE PERFORMANCE TEST RESULT

Device	Load Time	Frame Rate	Memory Usage
Google Pixel 3a	4.21s	60.0fps	29MB
Xiaomi Redmi Note 5 pro	3.98s	60.0fps	34MB
Asus Zenfone Zoom S	4.39s	59.2fps	16MB
Asus Zenfone 5 ZE620KL	4.41s	59.9fps	36MB
Asus Zenfone 3 ZE552KL	4.31s	59.9fps	23MB

Source: Author

Performance testing result of Flutter mobile application has average load time of 1.69 seconds, average frame rate of 60.0 frames per second, and average memory usage of 27.6MB as shown in Table 7 based on this paper's performance testing scenario.

TABLE 7. FLUTTER PERFORMANCE TEST RESULT

Device	Load Time	Frame Rate	Memory Usage
Google Pixel 3a	1.96s	60.0fps	5.2MB
Xiaomi Redmi Note 5 pro	1.91s	60.0fps	2.9MB
Asus Zenfone Zoom S	1.62s	60.0fps	1.9MB
Asus Zenfone 5 ZE620KL	1.45s	60.0fps	8.62MB
Asus Zenfone 3 ZE552KL	1.51	60.0fps	14MB

Source: Author

From comparison of Table 6 and Table 7, both Flutter and React Native does not have big difference in regards of average frame rate. Both of them reached around 60 frames per second, which is great. The difference lies in their load time and memory usage. The result from this project's performance testing shows that Flutter application has approximately half load time compared to React Native. Flutter has around 2 seconds of load time which is deemed acceptable according to Akamai Technologies (2015). Meanwhile, React Native's 4 seconds load time is very bad. This number is very high because React Native must load all the JavaScript modules. Flutter application also has lower Memory Usage compared to React Native because Flutter does not have to load the JavaScript modules meanwhile React Native does. Both of those can be remedied by importing less react modules.

6. CONCLUSIONS

This research is aimed to design and implement a software using Flutter and React Native and then compare their performance to see which one has better performance. From the result, Flutter is better than React Native due to having shorter load time and lower memory usage. This aligns with two hypotheses that stated Flutter has a shorter load time than React Native and Flutter has a lower memory usage than React Native. But, contrary to one of the hypotheses, React Native did in fact reach around 60 frames per second during usage. For further studies related to this research, there are some performance metrics that can be tested which are battery usage and CPU Usage. In addition to that, the application can also be tested in some other platforms, such as iOS.

REFERENCES

Akamai Technologies (2015), *Performance Matters Key Consumer Insights*. Retrieved 13 February 2021, from https://www.akamai.com/us/en/multimedia/documents/content/akamai-performance-matters-key-consumer-insights-ebook.pdf

Chatterjee, S., Chowdhury, K. S., & Sengupta, S. (2016), *Application performance measurement and reporting,* U.S. Patent No. 9,311,211. Washington, DC: U.S. Patent and Trademark Office.

Compfest (2019), *How Tokopedia Improve Its Most Important Page on iOS App*. Retrieved 12 February 2021, from https://compfest.wordpress.com/2019/08/14/how-do-we-improve-our-most-important-page-on-ios-app/

Dalmasso, I. *et al.* (2013), 'Survey, comparison and evaluation of cross platform mobile application development tools', the Proceedings of *9th International Wireless Communications and Mobile Computing Conference, IWCMC 2013*, pp. 323–328. doi: 10.1109/IWCMC.2013.6583580.

Jagiello, J. (2019), *Performance comparison between react native and flutter*. Bachelor Thesis, Umea University (accessed 2022) https://www.diva-portal.org/smash/get/diva2:1349917/FULLTEXT01.pdf

JetBrains (2020), *The State of Developer Ecosystem in 2020 Infographic*. Retrieved 12 February 2021, from https://www.jetbrains.com/lp/devecosystem-2020/

Latif, M. *et al.,* (2016), 'Cross platform approach for mobile application development: A survey', *the Proceedings of International Conference on Information Technology for Organizations Development, IT4OD 2016*. doi: 10.1109/IT4OD.2016.7479278.

Livani, Kho, I.E, Purnama, J. (2020), 'Business and System Analysis in Batik Online Platform for Plus Size', Journal of Applied Information, Communication and Technology, vol. 7(2).

Rahul Raj, C. P. and Tolety, S. B. (2012), 'A study on approaches to build cross-platform mobile applications and criteria to select appropriate approach', *2012 Annual IEEE India Conference, INDICON 2012*, pp. 625–629. doi: 10.1109/INDCON.2012.6420693.

Wu, W. (2018), *React Native vs Flutter, cross-platform mobile application frameworks*, Bachelor Thesis, Metropolia University of Applied Science (accessed 2022) https://www.theseus.fi/bitstream/handle/10024/146232/thesis.pdf?sequence=1