1. INTRODUCTION

The development of mobile applications in recent years has grown rapidly [1]. According to statistics provided by Statista, the number of downloads of mobile applications worldwide from 2016 to 2020 increased by more than 50%. Therefore, developers must develop applications quickly in order to keep up with rapid technological developments [2]. However, poor design can have a negative impact on application quality [2]. One of the quality attributes of software is performance [3]. The quality of software can be measured based on performance estimates such as application speed, latency, network error, and so on [4]. The effectiveness and efficiency of an application can determine its performance and increase user satisfaction [3]. This makes performance issues very important.

There are two phases in software design, namely architectural design and detailed design [5]. Both phases are very important to do to evaluate the quality of the application. The detailed design phase will define the components and design of the system in detail. Application developers are trying to determine the best design approach to build applications that are effective and efficient in terms of performance in the midst of rapid technological developments [3]. One of the design approaches that can be used by developers to overcome performance problems is the design patterns. Design patterns can solve problems in software development using object-oriented design principles [5]. however, as this research progressed, another approach was found that can be used to solve application quality problems based on the design problems found, namely refactoring the code smells and implementing design principles. Code smells and violations of design principles can be overcome by refactoring the source code [6],[7].

Many studies have been conducted related to this research problem. Aggarwal et al [4] suggests that the quality of an application can be controlled by evaluating performance estimates based on different performance criteria. This study also adds that application performance can be estimated by utilizing different criteria such as application speed, application latency, and network error. Mahendra et al [8] have conducted research on the performance of mobile applications by comparing several metrics such as Central Processing Unit (CPU) and memory usage, response time, frame rate, and application size between Flutter, React Native, and Native Android applications. In another study, Qasim et al [3] further investigated performance in terms of energy use by applying several design patterns such as Façade, Observer, and Abstract Factory patterns. The study found that the performance of mobile applications in terms of power usage can be optimized by applying design patterns. Meanwhile, Hect, G. et al [9] identified performance metrics in their research on the effect of code smell on performance. The study describes the metrics used to measure performance, namely Frame Time, Number of Delayed times, Memory Usage, and Number of Garbage Collection Calls. Willocx, M. et al [10] also describes the metrics used to measure performance in mobile applications, namely Response Time, CPU Usage, Memory Usage, Disk Space, and Battery Usage.

Gamma et al [11] explained that the design pattern is communication between classes and objects that are adapted to solve general design problems in a particular context. There are 23 design patterns categorized based on their intent, namely Creational Patterns, Structural Patterns, and Behavioral Patterns [12]. Design problems arise when the selection of software design has a bad influence on quality attributes [13]. Some examples of design problems are Cyclic Dependency, Scattered Concern, Ambiguous Interface, and Fat Interface. Several strategies can be used by developers in identifying design problems, namely smell-based, problem-based, principle-based, element-based, quality attribute-based, and pattern-based [13]. Code smells are a symptom that is an indication of the emergence of code or design problems in the source code [6],[14]. Code smells can infect a method, a class, or the entire code and can cause bugs and also increase the risk of code maintenance which results in high maintenance costs [6]. Code smells can be divided into several types such as Bloaters, Object-Oriented (OO) Abusers, Change Preventers, Dispensables, Couplers, and Incomplete Library Classes [15]. The software design principle also is the basis for creating quality attributes. The design principle will maximize profit and reduce development costs [16]. There are several Object Oriented design principles

[17],[7] , namely Single Responsibility, Open-Closed, Liskov Substitution, Dependency Inversion, Interface Segregation, Separation of Concerns, Law of Demeter, DRY (Don't Repeat Yourself), and YAGNI (You Ain't Gonna Need It).

Research [5],[6],[7],[18] discusses software architecture and software design such as design patterns, design principles, and code smells that are interconnected with each other in determining the quality of a software in terms of development, performance, and maintainability. Therefore, further research is carried out to determine the relationship between the design pattern approach, code smells, and design principles in influencing application quality, especially on mobile application performance. This study aims to analyze the impact of the application of design patterns, design principles, and refactoring of the code smells on the change of the value of mobile application performance metrics measured using the average value of Central Processing Unit (CPU) usage, memory usage, and frame rate. This study was also conducted by comparing the results of each approach to changes in the value of mobile application performance metrics measured using several metrics such as CPU usage, memory usage, and frame rate before and after implementation.