**UNIVERSITI TEKNOLOGI MARA**

MOBILE APPLICATION EFFORT ESTIMATION USING COSMIC FUNCTION POINT



**MUIHAMMAD ARIF FIKRI BIN MUSTAPA**



**BACHELOR OF COMPUTER SCIENCE (HONS.)**

**JANUARY 2022**

**Universiti Teknologi MARA**

**MOBILE APPLICATION EFFORT ESTIMATION USING COSMIC FUNCTION POINT**

**Muhammad Arif Fikri Bin Mustapa**

**Thesis submitted in fulfilment of the requirements for Bachelor of Computer Science (Hons.) Faculty of Computer and Mathematical Sciences**

**January 2022**

**SUPERVISOR APPROVAL**

**MOBILE APPLICATION EFFORT ESTIMATION USING COSMIC FUNCTION POINT**

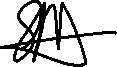
By

**MUHAMMAD ARIF FIKRI BIN MUSTAPA**

**2020461922**

The thesis was prepared under the supervision of the project supervisor, Assoc. Prof. Ts. Dr. Nur Atiqah Sia Abdullah and co-supervisor, Dr Nur Ida Aniza Rusli. It was submitted to the Faculty of Computer and Mathematical Sciences and was accepted in partial fulfilment of the requirements for the degree of Bachelor of Computer Science (Hons.).

Approved by



………………………………

Assoc. Prof. Ts. Dr. Nur Atiqah Sia Binti Abdullah

Project Supervisor

JANUARY 23, 2022

**STUDENT DECLARATION**

I certify that this thesis and the project to which it refers is the product of my own work and that any idea or quotation from the work of other people, published or otherwise are fully acknowledged in accordance with the standard referring practices of the discipline.

**Shape, arrow

Description automatically generated**

………………………………

MUHAMMAD ARIF FIKRI BIN MUSTAPA

2020461922

JANUARY 23, 2022

**ACKNOWLEDGEMENT**



Alhamdulillah, I am very grateful to Allah. All praises to Allah and His blessing for the completion of this thesis. I thank God for all the opportunities, trials and strength that have been showered on me to finish writing my Final Year Project (FYP) proposal. I experienced so much during this process, not only from the academic aspect but also from the aspect of personality. My humblest gratitude to the holy Prophet Muhammad (Peace be upon him) whose way of life has been a continuous guidance for me.

First and foremost, I would like to express my deep and sincere gratitude to my project supervisor, Assoc. Prof. Ts. Dr. Nur Atiqah Sia Binti Abdullah together with co-supervisor, Dr. Nur Ida Aniza Binti Rusli for their continuous support of my Final Year Project on Degree level. Their guidance leads me in all time when researching and writing this proposal.

Beside my supervisor and co-supervisor, I would like to thank my Final Year Project’s lecturer, Dr. Siti Khatijah Nor Abdul Rahim for her encouragement, teaching, and insightful comments. I’ve learn a lot and gain much experience from her teaching on how to successfully finish my project proposal. Without her kindness, I wouldn’t be able to understand the format of my FYP proposal.

My sincere thanks also goes to my fellow classmates especially my project partner, Azri Mustaqim that had been together with me for the stimulating discussion and sharing problems so that I can improves my skills to develop the project.

Last but not least, I would like to thank my family for supporting me without any condition throughout my life. This project couldn’t be done successfully without them.

**ABSTRACT**

In the present time, mobile applications are reigning the business world in a way that the revenue earned is expanding each passing day. The aim of this project is to develop a web-based tool to estimate the mobile application effort estimation using COSMIC Function Point. Therefore, there are some problems that developers must face when dealing with mobile application effort estimation. The major problem is wasting unnecessary resources while developing a mobile application. The waste can be significant to certain company that did not have proper plan on the development. For this project, waterfall model is used as the methodology that consist of requirement, design, implementation, and verification phase. For further understanding about this project, a review on Software Effort Estimation is conducted. Gathering information about similar system that existed and comparing the selected system to obtain the features available for the project. A review on both parametric and non-parametric estimation model is made to understand the techniques of software effort estimation. In design phase, the outcome includes system architecture, use case diagram and description, user interface design, and COSMIC Function Point algorithm design. The project continues with implementation phase where the description of hardware and software requirement is provided. The expected outcome for this project is a mobile application effort estimation tool.

**TABLE OF CONTENTS**

CONTENT PAGE

[SUPERVISOR APPROVAL ii](#_Toc93412595)

[STUDENT DECLARATION iii](#_Toc93412596)

[ACKNOWLEDGEMENT iv](#_Toc93412597)

[ABSTRACT v](#_Toc93412598)

[TABLE OF CONTENTS vi](#_Toc93412599)

[LIST OF FIGURES ix](#_Toc93412600)

[LIST OF TABLES x](#_Toc93412601)

[LIST OF ABBREVIATIONS xi](#_Toc93412602)

CHAPTER ONE: INTRODUCTION

[1.1 Project Background 1](#_Toc93412604)

[1.2 Problem Statement 2](#_Toc93412605)

[1.3 Project Objectives 3](#_Toc93412606)

[1.4 Scope of Study 3](#_Toc93412607)

[1.5 Significance of Study 4](#_Toc93412608)

1.6 Summary 4

CHAPTER TWO: LITERATURE REVIEW

[2.1 Software Effort Estimation 5](#_Toc93412611)

[2.1.1 Parametric Estimation Model 6](#_Toc93412612)

[2.1.2 Non-Parametric Estimation Model 7](#_Toc93412613)

[2.2 Function Point 8](#_Toc93412614)

[2.3 COSMIC Function Point 11](#_Toc93412619)

[2.3.1 Software Context Model 12](#_Toc93412620)

[2.3.2 Generic Software Model 13](#_Toc93412620)

[2.4 Mobile Applications 15](#_Toc93412621)

[2.4.1 Native App 15](#_Toc93412622)

[2.4.2 Web App 16](#_Toc93412623)

[2.4.3 Hybrid App 17](#_Toc93412624)

[2.4.4 Comparison of Types Of Mobile Application 18](#_Toc93412625)

[2.4.5 Mobile Application Development 18](#_Toc93412626)

2.4.5.1 Mobile Application Front-End 19

2.4.5.2 Mobile Application Back-End 19

[2.5 Similar System 19](#_Toc93412628)

[2.5.1 Estimate My App ByOOZOU 20](#_Toc93412626)

[2.5.2 App Cost Calculator By DIGITALYA 26](#_Toc93412626)

[2.5.3 How Much To Make An App By Z1 29](#_Toc93412626)

[2.5.4 Mobile App Development Cost Calculator By BUILDFIRE 32](#_Toc93412626)

[2.5.5 Comparison of Similar Systems 37](#_Toc93412626)

[2.6 Summary 42](#_Toc93412628)

CHAPTER THREE: METHODOLOGY

[3.1 Software Development Methodology 43](#_Toc93412630)

[3.2 Project Framework 44](#_Toc93412631)

[3.3 Phase 1 Requirement 45](#_Toc93412632)

[3.4 Phase 2 System Design 46](#_Toc93412635)

[3.4.1 System Architecture 46](#_Toc93412636)

[3.4.2 Use CaseDiagram 47](#_Toc93412637)

[3.4.3 COSMICFunctionPoint Algorithm 49](#_Toc93412638)

[3.4.4 User Interface Design 50](#_Toc93412639)

[3.5 Phase 3 Implementation 51](#_Toc93412640)

[3.6 Phase 4 Verification 52](#_Toc93412644)

[3.7 Summary 55](#_Toc93412646)

[REFERENCES 56](#_Toc93412647)

**LIST OF FIGURES**

**FIGURE PAGE**

[2.1 Generic Software Model 14](#_Toc93367197)

[2.2 Estimate My App UI (Step 1) 25](#_Toc93367198)

[2.3 Estimate My App UI (Final Result) 29](#_Toc93367199)

[2.4 App Cost Calculator UI (Final Result) 32](#_Toc93367200)

[2.5 How Much To Make An App UI (Final Result) 32](#_Toc93367201)

[2.6 Mobile App Development Cost Calculator UI (Final Result) 37](#_Toc93367202)

[3.1 Waterfall Model 43](#_Toc93367205)

[3.2 System Architecture 46](#_Toc93367206)

[3.3 Use Case Diagram 47](#_Toc93367207)

[3.4 Input User Interface Design 50](#_Toc93367208)

[3.5 Output User Interface Design 51](#_Toc93367209)

**LIST OF TABLES**

**TABLE PAGE**

[2.1 Weight of FP’s Component 9](#_Toc93367347)

[2.2 Description of GSCs 10](#_Toc93367348)

[2.3 Comparison of Native, Web, and Hybrid App 18](#_Toc93367349)

[2.4 Estimate My App Features and Description 22](#_Toc93367350)

[2.5 App Cost Calculator Features and Description 27](#_Toc93367351)

[2.6 How Much To Make An App Features and Description 31](#_Toc93367352)

[2.7 Mobile App Development Cost Calculator Features and Description 33](#_Toc93367353)

[2.8 Comparison of Four Similar Systems 38](#_Toc93367354)

[3.1 Project Framework 44](#_Toc93367355)

[3.2 Input Data Use Case Description 47](#_Toc93367356)

[3.3 Calculate Effort Use Case Description 48](#_Toc93367357)

[3.4 Calculate Schedule Use Case Description 48](#_Toc93367358)

[3.5 Display Output Use Case Description 48](#_Toc93367359)

[3.6 Software Requirements 52](#_Toc93367360)

[3.7 Hardware Requirements 52](#_Toc93367361)

[3.8 Functionality Test Case 53](#_Toc93367361)

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| COSMIC | Common Software Measurement International Consortium |
| FP | Function Point |
| GSC | General System Characteristics |
| CFP | COSMIC Function Points |
| API | Application Programming Interface |
| RAM | Random Access Memory |
| CPU | Central Processing Unit |
| ILF | Internal Logical Files |
| EIF | External Interface Files |
| EI | External Input |
| EO | External Output |
| EQ | External Inquiry |
| VAF | Value Adjustment Factor |
| IFPUG | International Function Point Users Group |
| FPA | Function Point Analysis |
| FUR | Functional User Requirement |
| DM | Data Movements |
| GPS | Global Positioning System |
| HTML | Hyper Text Markup Language |
| CSS | Cascading Style Sheet |
| MVP | Minimum Viable Product |
| UI | User Interface |
| OS | Operating System |

# **CHAPTER 1**

**INTRODUCTION**

This chapter provides the background and rationale for the study. It gives details of Mobile Application Effort Estimation Using COSMIC Function Point, follows by highlighting the problem statement that led to the study. Then, this chapter continues with aim and objectives, before stating the scope and significance of the study. Lastly, it ends with a summary for this chapter.

**1.1 Project Background**

In the present time, mobile applications are reigning the business world in a way that the revenue earned is skyrocketing with each passing day. From the studies it has been found that the total number of downloads is expected to go up to 352.9 billion by 2021. The present era of mobile applications is enjoying the digitization phase due to the communication has become fast-paced and simpler. Regarding to this situation, a demand in mobile apps development increased drastically day by day. However, due to the increasing in demand, the cost of the development also should be put into consideration to avoid any uncertainty.



The development cost includes manpower, resources and schedule for developing the mobile application. Estimating effort has always been considered an important element at the start of each software development project (Farih, Nafil & Messousi, 2021).

Effort estimation is a key factor for software project success, defined as delivering software of agreed quality and functionality within schedule and budget. Traditionally, effort estimation has been used for planning and tracking project resources. Effort estimation methods founded on those goals typically focus on providing exact estimates and usually do not support objectives that have recently become important within the software industry, such as systematic and reliable analysis of causal effort dependencies (Trendowicz et al., 2021). As for function point measurement, there are several ISO standard that recognized and one of them are Common Software Measurement International Consortium (COSMIC). COSMIC Function Point is a method to measure a standard functional size of a piece of software. It will be focusing on four data movement, entry, read, write, and exit data movement.

Parametric estimating is a statistics-based technique for calculating the estimated amount of financial resources or time needed to complete and finish a project, an activity, or a piece of a project (Project-management.info, 2018). A statistical (or supposed) correlation between a parameter and a cost or time value is used to calculate an estimate. The size of the present project is then scaled based on the observed correlation (PMI Practice Standard for Project Estimating, 2nd edition, 2019).

Non-parametric estimation is a statistical method that allows the functional form of a fit to data to be obtained in the absence of any guidance or constraints from theory. As a result, the procedures of nonparametric estimation have no meaningful associated parameters.

**1.2 Problem Statement**

Though the advantage and the needs of effort estimation, there are some problems that developers must face when dealing with cost estimation. According to a study, about 66% of the total software projects investigated are characterized either by a delay in the schedule or have passed the foreseen budget, causing the loss of huge resources for the involved organizations (Charette, 2021).

The cost of developing software can be over-estimated by the company and developers. The US economy, from 2005-2010, faced a loss of about 25 to 75 billion dollars due to the impact of the software project’s failure (as cited by Emam, 2021). Both overestimations and underestimations of the software project efforts are a disaster for the organizations.

**1.3 Project Objectives**

The aim of this project is to develop a web-based tool to estimate the mobile application effort estimation using COSMIC Function Point. The objectives of this project are as follow:

1. To identify COSMIC function point for mobile application effort estimation.
2. To construct a web-based tool to estimate the mobile application effort.
3. To test the functionality of the web-based tool.

**1.4 Scope of Study**

This project involves developing a web-based tool to estimate the overall cost of a mobile application using COSMIC Function Point. The targeted user of this project includes project manager and mobile application developers. The estimation is calculated based on the limited parameters of COSMIC function point and certain types of mobile application. This web-based tool is used to calculate the effort and cost needed for a small-scale mobile application. The estimated output of this project is manpower (effort), schedule and cost for the development of a mobile application.

This project will be focusing on the movement of data from every feature that is provided as it is the requirement for COSMIC Function Point calculation. The data movements are entry, exit, read, and write data. Thus, the user interface will be consisting of eight type of feature which are platform, size, user and account, generated content, dates and location, generated revenue, and social and engagement.

**1.5 Significance of Study**

This project will provide new insight for a company on the development of its own software. Through this project, the top management that involved is able to handle the possible budget for the mobile application development. This will bring benefits to the company as there should be no waste of available resources. Successful strategies for a more predictive measure of a software development may provide management with increased accuracy in delivery expectations, thus providing greater alignment with organizational budgets and customer expectations.

This project will help to manage efficiently the scheduling and manpower to the mobile application development. It will provide development managers and teams with practical and effective strategies for accurate effort estimations. Therefore, this project able to engender positive social change for a variety of stakeholders.

**1.6 Summary**

This chapter describes the importance of mobile application effort estimation. Mobile application can be considered as the fastest and effective ways of communication throughout the globe. Nonetheless, mobile application is widely used for e-commerce business as it has grown bigger every year. The major problem is cost of developing a mobile application can be underestimated or overestimated due to improper estimation.

Therefore, the purpose of this project is to develop a web-based tool to estimate the mobile application effort using COSMIC Function Point. This project will focus on the parameters of COSMIC function point and certain types of mobile application. The expected output will be manpower (effort), schedule and cost. This project is beneficial to project manager and mobile application developers.

**CHAPTER 2**

**LITERATURE REVIEW**

This chapter describes the overview of related research for this project. It started with section 2.1 that will give an overview of Software Effort Estimation. Next, in section 2.2 will review about the Function Point method. Section 2.3 will review about COSMIC Function Point method that is used in this project. In section 2.4, an overview of mobile application is given including the type of mobile application, comparison of mobile application and mobile application development. Section 2.5 will be a review about similar system that has the same concept with the project together with the comparison. Finally, in section 2.6 the summary of this chapter is written.

* 1. **Software Effort Estimation**

The practice of determining the most realistic amount of effort (expressed in terms of person-hours or money) required to develop or maintain software based on incomplete, and ambiguous data is known as effort estimation in software development (Kumar, Behera, Kumari K, Nayak, Naik, 2020). Project plans, iteration plans, budgets, investment evaluations, pricing processes, and bidding rounds can all benefit from effort estimation. Predicting the cost and effort required in the early stages of planning is a difficult assignment for project managers. Estimating software before it is developed helps reduce risk and improve the project's success rate (Kumar et al., 2020).

Software effort estimation is one of the most important activities in software project development management. It is crucial to optimal planning and is important for controlling the software development process (Ezghari & Zari, 2018). Software estimation can find software size estimates, software development effort, software development cost and software development schedule for the software project in an environment by using defined methods, tools, and techniques (as cited in Chemuturi, 2017). A good software estimation method will produce optimal resource by utilization and improve the quality of the final products to reach higher level of customer satisfaction (as cited in Kirmani and Wahid, 2017). Software Effort Estimation can be divided into two approaches, parametric estimation model and non-parametric estimation model.



* + 1. **Parametric Estimation Model**

Parametric estimating model is a statistical-based technique for calculating the estimated number of financial resources or time needed to complete and finish a project, an activity, or a piece of a project (Project Management Institution, 2021). A statistical (or supposed) correlation between a parameter and a cost or time value is used to calculate an estimate. The size of the present project is then scaled based on the observed correlation (Project Management Institution, 2021).

The statistical examination of the project's historical data is the centre of the algorithmic model. It requires precise input of particular parameters such as line of codes (LOC), function point (FP), the number of user displays, and the complexity of the graphical user interface (GUI). Many algorithmic models exist, including the Constructive Cost Model (COCOMO), FP model, Use Case Point (UCP), and COSMIC Function Point as well as the Software Life Cycle Management (SLIM) model.

To produce an estimation using parametric technique, the project manager will split the project down into sub-components (typically a deliverable) and match them with the relevant equation. While the equations can be derived from previous projects, the details of these projects will be erased when the equations are developed (Planisware, 2019). Parametric technique uses the relationship between variables to calculate the cost or duration. Essentially, a parametric estimation is determined by identifying the unit cost or duration and the number of units required for the project or activity. The measurement must be scalable to be accurate (Goodrich, 2017). For this project, parametric estimation model will be used as for the parameters that are considered in measuring the size of a software.

* + 1. **Non-Parametric Estimation Model**

Non-parametric modelling typically does not impose restrictions on the functional form that describes the relationship between the dependent and independent variables and for this reason it can be considered as a robust estimation procedure (as cited in Hardle, 2017). Moreover, non-parametric modelling has the ability to detect non-linear dependencies among the response variable and the independent predictors. Expert judgement, thumb rule decision, Delphi technique, Software Productivity Research SPR, broad band Delphi technique, probe software process PSP, and team software process technique (TSP) are all examples of non-parametric models (as cited in Hardle, 2017).

The non-parametric method is a branch of statistics in which the data are not assumed to come from prescribed models that are determined by a small number of parameters (Investopedia, 2021). The non-parametric analysis is often best suited when considering the order of something, where even if the numerical data changes, the results will likely stay the same (Investopedia, 2021). Parametric and non-parametric methods are often used on different types of data. Parametric statistics generally require interval or ratio data.

Although non-parametric statistics have the advantages of having to meet few assumptions, they are less powerful than parametric statistics. Since non-parametric statistics makes fewer assumptions about the sample data, its application is wider in scope than parametric statistics (Westfall, 2021). In cases where parametric testing is more appropriate, non-parametric methods will be less efficient. This is because nonparametric statistics discard some information that is available in the data, unlike parametric statistics (Westfall, 2021).

* 1. **Function Point**

The function point is a "unit of measurement" to express the amount of functionality in a software provided to a user. Function points are used to compute a functional size measurement (FSM) of software (Tanebata, Hazeyama, Yamada, Furukawa, 2021). They are widely accepted as an industry standard for functional sizing. It focuses on functions that are visible to users such as data input or output and quantifies size of software by the number of functions and their weights (Tanabata et al., 2021). Functions of a software system consist of the following components according to IFPUG’s Counting Practice Manual :

* Internal Logical File (ILF): data or control information maintained through one or more elementary process of the target application.
* External Interface File (EIF): data or control information referenced through one or more elementary processes.
* External Input (EI): an elementary process to maintain an ILF or alter the behavior of the application.
* External Output (EO): a process that presents information to user through processing logic.
* External Inquiry (EQ): an elementary process for presenting information to the user through retrieval of data or control information from an ILF or EIF.

The formula of calculating software function point (FP) are as follows :

Unadjusted Function Point (UFP) is the main step of this process where all the components mentioned above are added together and labeled as unadjusted function point count. The formula of UFP calculation is as follow:

Table 2.1 shows the function types and the weighting factors for the varying complexities. For each function identified above the function is further classified as simple, average or complex and a weight is given to each. The sum of the weights quantifies the size of information processing and is referred to as the Unadjusted Function points.

**Table 2.1** Weight of FP’s component.

|  |  |  |  |
| --- | --- | --- | --- |
| **Function type** | **Simple** | **Average** | **Complex** |
| Internal Logical Files (ILF) | 7 | 10 | 15 |
| External Interface Files (EIF) | 5 | 7 | 10 |
| External Input (EI) | 3 | 4 | 6 |
| External Output (EO) | 4 | 5 | 7 |
| External Inquiry (EQ) | 3 | 4 | 6 |

(*Source*: Tanebata et al., 2021)

Value Adjustment Factor (VAF). In this step the value adjustment factor is determined. VAF contains 14 General System Characteristics (GSC) of the system or application that defines the types of application characteristics and is rated on a scale of 0 to 5. The sum of all the 14 GSC rates are calculated to give out a mathematical value and is labeled as Total Degree Influence (TDI). TDI is used in the calculation of VAF and its value may vary from 0 to 35 (Fingent, 2020).

The degrees of influence range on a scale of zero to five, from no influence to strong influence. Each characteristic is assigned the rating based upon detail descriptions provided by the IFPUG 4.1 Manual (Fingent, 2020). The rating are as follows:

|  |
| --- |
| 0 - Not present, or no influence  1 - Incidental influence  2 - Moderate influence  3 - Average influence  4 - Significant influence  5 - Strong influence throughout |

Table 2.2 shows the 14 General System Characteristics (GSC) and its description:

**Table 2.2** Description of GSCs

|  |  |  |
| --- | --- | --- |
| **General System Characteristic** | | **Brief Description** |
| 1 | Data communications | How many communication facilities are there to aid in the transfer or exchange of information with the application or system? |
| 2 | Distributed data processing | How are distributed data and processing functions handled? |
| 3 | Performance | Did the user require response time or throughput? |
| 4 | Heavily used configuration | How heavily used is the current hardware platform where the application will be executed? |
| 5 | Transaction rate | How frequently are transactions executed daily, weekly, monthly, etc.? |
| 6 | On-Line data entry | What percentage of the information is entered On-Line? |
| 7 | End-user efficiency | Was the application designed for end-user efficiency? |
| 8 | On-Line update | How many ILF’s are updated by On-Line transaction? |
| 9 | Complex processing | Does the application have extensive logical or mathematical processing? |
| 10 | Reusability | Was the application developed to meet one or many user’s needs? |
| 11 | Installation ease | How difficult is conversion and installation? |
| 12 | Operational ease | How effective and/or automated are start-up, back up, and recovery procedures? |
| 13 | Multiple sites | Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations? |
| 14 | Facilitate change | Was the application specifically designed, developed, and supported to facilitate change? |

(*Source*: Fingent, 2020).

**2.3 COSMIC Function Point**

Common Software Measurement International Consortium (COSMIC) Function Point (COSMIC FP) is a unit of measure of software functional size. The size is a consistent measurement (or estimate) which is very useful for planning and managing software and related activities (Scopemaster, 2021). The process of measuring software size is called functional size measurement (FSM). COSMIC FP is applicable to business, real-time and infrastructure software at any level of decomposition (from a whole software system down to a single re-usable component or a user story). It is independent of the technology or processes used to develop the system and also an ISO standard (Scopemaster, 2021).

COSMIC FP was designed based on the FPA but directed to applications in real time and multi-platform software. Its characteristics include estimates the development effort, evolves software quality, compares specified systems in different languages, in terms of productivity and cost maintenance, considering concepts such as functional requirements of users, software users, layers and boundaries (Kaur, 2018).

The COSMIC Function Point method is the latest in the family of function point methods, and it is expected that other methods will change in the future to better adapt to the COSMIC FP method. It was created to overcome the problems that affected its predecessors from the mentioned family (Rankovic, Ivanoviv and Lavic, 2021). The COSMIC FP method observes the elementary communication and data movement that is exchanged in the system. Four basic types of data movement can be exchanged between users and systems, between system components, between systems and data warehouses (as cited in Angara, 2018). The COSMIC Function Point sizing method of measuring software requirements is based on two main principles, software context model and generic software model (Scopemaster, 2021).

**2.3.1 Software Context Model**

The Software Context Model defines the software to be measured (Scopemaster, 2021). The definitions are as follows:

* Software is bounded by hardware and typically structured into layers.
* The scope of any piece of software to be measured shall depend on the purpose of the measurement and shall be confined wholly within a single layer.
* The functional users of a piece of software to be measured shall be identified from its Functional User Requirements (FUR) as the senders and/or intended recipients of data to/from the software respectively.
* A precise COSMIC size measurement of a piece of software requires that its FUR are known at a level of granularity at which its functional processes and sub-processes may be identified.
* An approximate COSMIC size measurement is possible if its FUR are measured at a high level of granularity by an approximation approach and scaled to the level of granularity of the functional processes.

**2.3.2 Generic Software Model**

The Generic Software Model defines the generic concepts that is applicable to all software (Scopemaster, 2021). The definitions are as follows:

* A piece of software interacts with its functional users across a boundary, and with persistent storage within the boundary.
* The FUR of a piece of software can be mapped into unique functional processes.
* Each functional process is started by its triggering Entry data movement. The data group moved by the triggering Entry is generated by a functional user in response to a triggering event.
* A functional process shall include at least one Entry data movement and either a Write or an Exit data movement. There is no upper limit to the number of data movements in a functional process
* Each functional process consists of sub-processes, data movements (DMs) and data manipulations.
* As an approximation for measurement purposes, the COSMIC method assumes that the functionality of any data manipulation is accounted for by the data movement with which it is associated.
* There are four data movement types, Entry, Exit, Write and Read.
* A data movement moves a single data group, which consists of a unique set of data attributes that describe a single object of interest.

The definition of four data movement are as follows:

1. Entry : A data movement that moves a data group from a functional user across the boundary into the functional process where it is required.
2. Exit : A data movement that moves a data group from a functional process across the boundary to the functional user that requires it.
3. Read : A data movement that moves a data group from persistent storage into the functional process that requires it.
4. Write : A data movement that moves a data group from inside a functional process to persistent storage.

Diagram

Description automatically generatedFigure 2.1 illustrates the four major components in COSMIC FP. The four components (Entries, Exits, Reads, and Writes) is included in Figure 2.1.

**Figure 2.1** Generic Software Model

(*Source*: ScopeMaster 2021)

From Figure 2.1, functional users (hardware devices, other software, or humans) is responsible for the data entries. The total number of entry data movement (Entries) across the boundary moved into the functional process. Then the total number of writes data movement (Writes) moves a data group from the functional process to the persistent storage. The total number of reads data movement (Reads) moves the data group from persistent storage to the functional process. Finally, the total number of exits data movement (Exits) moves the data group from the functional process to the functional users that requires it. A movement is considered as one COSMIC Function Point (CFP). The formula of calculation the size of software using COSMIC FP is shown below:

**2.4 Mobile Application**

A mobile application, also referred to as a mobile app or simply an app, is a computer program or software application designed to run on a mobile device such as a phone, tablet, or watch. Mobile applications were designed to aid productivity in areas such as email, calendars, and contact databases, but public demand prompted rapid expansion into other areas such as mobile games, factory automation, GPS and location-based services, order tracking, and ticket purchases, resulting in millions of apps now available (Techopedia, 2020). Mobile applications are divided into three broad categories, which are native apps, web apps, and hybrid apps.

**2.4.1 Native App**

Native apps are built for a specific mobile operating system, usually iOS or Android. Native apps enjoy better performance and a more finely tuned user interface (UI), and usually need to pass a much stricter development and quality assurance process before they are released (Techopedia, 2020). Native apps are coded using a variety of programming languages. Some examples include Java, Kotlin, Python, Swift, Objective-C, C++, and React (CleverTap, 2020).

Native apps have the advantage of being faster and more dependable in terms of performance due to their exclusive focus. They use the device's resources more efficiently than other sorts of mobile apps. Native apps make use of the device's native user interface, providing a better customer experience (CleverTap, 2020). Because native apps interact directly with the device's hardware, they have access to a wide range of functions like as Bluetooth, phonebook contacts, camera roll, and Near Field Communication (NFC) (CleverTap, 2020).

The issue with native apps, on the other hand, is that once you start developing them, you'll have to duplicate your efforts for each platform. You can't reuse the code you write for one platform on another. Costs rise significantly. Not to mention the time and effort required to maintain and upgrade each version's codebase (CleverTap, 2020). And then, every time there’s an update to the app, the user has to download the new file and reinstall it. This also means that native apps do take up precious space in the device’s storage (CleverTap, 2020).

**2.4.2 Web App**

Web apps behave similarly to native apps but are accessed via a web browser on your mobile device. Web apps are used in HTML5 or CSS and require minimum device memory since its run through a browser. The user is redirected on a specific web page, and all information is saved on a server-based database. Web apps require a stable connection to be used (Techopedia, 2020). Because it is web-based, there is no need to adapt it to a certain platform or operating system. This reduces the development costs.

Web apps do not take up as much memory on your device as native apps do, making maintenance easier and faster. The update does not need to be downloaded from the app store. However, it's important to note that web apps are completely reliant on the device's browser. There will be features that are available in one browser but not in another, potentially offering consumers different experiences (Valdellon, 2020). Web apps also would not work totally offline because they are just shells for websites. Even if it has offline mode feature, it will still require an internet connection to back up your data, provide new data, or refresh what's on screen (Valdellon, 2020).

**2.4.3 Hybrid App**

A hybrid application (hybrid apps) is a software application that combines both native apps and web apps. Web apps that have been wrapped in a native software shell are known as hybrid apps. The shell can connect to whatever features the mobile platform provided through a browser embedded in the app once it's downloaded and installed locally from an app store. The browser and its plug-ins are invisible to the end user since they run on the back end (Tech Target, 2019).

Hybrid apps are popular because they allow developers to write code for a mobile app once and still accommodate multiple platforms. Because hybrid apps add an extra layer between the source code and the target platform, they may perform slightly slower than native or web versions of the same app (Tech Target, 2019). The features of hybrid apps includes the ability to function whether or not the device is connected, integration with the mobile device’s file system, integration with web-based services, and provides an embedded browser to improve access to dynamic online content (Tech Target, 2019).

Hybrid applications are similar to Web apps, but they are downloaded to the device like native apps. Hybrid apps, like Web apps, are usually written in HTML5, CSS, and JavaScript. Code is run inside a container in hybrid apps. HTML and JavaScript are rendered using the device's browser engine, while native APIs are used to access device-specific hardware (Tech Target, 2019).

The advantages of hybrid apps are that it is able to operates on different platform, faster build times compared to native apps, cheaper to develop in terms of cost, easier to launch patches and updates, and has the ability to work online and offline (Tech Target, 2019).

However, there are some drawbacks which the appearance may be vary from platform to platform. This problem is due to leaning development on one platform. Supposed a development team leans their work on one platform, another supported platform may lack in quality, or suffer from bugs (Tech Target, 2019). Other than that, User experience may fall if the user interface is not similar to and well enough designed to what browsers the user is used to (Tech Target, 2019).

**2.4.4 Comparison of Types Of Mobile App**

The comparison of hybrid, native, and web apps in terms of development language, code portability, access devices-specific features, leverage existing knowledge, advanced graphic, upgrade flexibility, and installation experience are shown in Table 2.3:

**Table 2.3** Comparison of hybrid, native and web apps.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Native apps | Web apps | Hybrid apps |
| Development languages | Native only | Web only | Native and Web or Web only |
| Code portability | None | High | High |
| Access device-specific feature | High | Low | Medium |
| Leverage existing knowledge | Low | High | High |
| Advanced graphics | High | Moderate | Moderate |
| Upgrade flexibility | Low | High | Moderate |
| Installation experience | High | Medium | High |
| Development Cost | High | Low | Moderate |
| Maintenance Cost | High | Low | Moderate |
| Development Time | Slow | Fast | Moderate |

(*Adapted:*  ResearchGate, 2021)

**2.4.5 Mobile Application Development**

Mobile application development is the process of creating software applications that run on a mobile device, and a typical mobile application utilizes a network connection to work with remote computing resources. Hence, the mobile development process involves creating installable software bundles (code, binaries, assets, etc.) , implementing backend services such as data access with an API, and testing the application on target devices (AWS, 2021). In today’s modern smartphone market, there are two main platforms. Apple Inc.’s iOS platform is one of them. The iOS platform is the operating system that runs on Apple’s popular iPhone smartphones. Google’s Android is the second. Many other OEMs, in addition to Google, use the Android operating system to build their own smartphones and other smart devices (AWS, 2021).

Although there are some similarities in terms of app development between the two platforms, creating for iOS and Android requires the use of distinct software development kits (SDKs) and a different development toolchain (AWS, 2021). While Apple utilizes iOS exclusively for its own devices, Google makes Android available to other firms if they meet certain criteria, such as delivering devices with Google applications. Targeting both platforms allows developers to create apps for hundreds of millions of devices (AWS, 2021). The mobile application “Front-End,” which resides on the mobile device, and the services “Back-End,” which supports the mobile front-end, are two interconnected key components of a mobile application.

**2.4.5.1 Mobile Application Front-End**

The visual and interactive element of the application that the user interacts with is known as the mobile front-end. It usually resides on the device, or at the very least, an icon for the app is displayed on the home screen or pinned in the device’s application catalogue. The app can be downloaded from the platform app store, side-loaded directly onto the device, or accessed via the device’s browser, like with Progressive Web Application (PWA) (AWS, 2021).

Front end development technologies include components that are made of code. Typically, this code is HTML, CSS, or even JavaScript (Good Firms, 2020). These are known as front end developer languages. These files will determine the colour and interactive features of the front end and this is what the user sees and experiences when they open the app (Good Firms, 2020).

**2.4.5.2 Mobile Application Back-End**

The server-side of a website or application is also known as the backend. It organizes and stores data, as well as ensuring that everything on the website’s client side performs well (Lizard Global, 2020). While the backend does not interact directly with users, it plays an important role behind the scenes by providing essential functionality to a website (Lizard Global, 2020). The front-end will not function effectively without a clean and proper backend. So, even though it does not interact with the backend directly with the frontend, it are indirectly involved in all of the backend activities (Lizard Global, 2020).



The backend includes activities such as writing APIs, creating libraries, and working with system components without user interfaces. The core functionality of web apps is usually managed by the backend (Lizard Global, 2020). For example, when purchasing something from an online store, the backend manages the actual money transactions when going through the checkout process. While the front-end makes sure the “checkout” button is positioned properly on the page and sends you to the next page, the backend interacts with services outside of your own app or website, such as bank app or PayPal (Lizard Global, 2020).

The server-side of an application or website has its own programming languages. These languages are made in such a way that they can interact easily with the front-end, creating an optimally functioning piece of software. Back-end language include NodeJS, JavaScript, C++, Swift, and Java (Lizard Global, 2020)



**2.5 Similar Systems**

**2.5.1 Estimate My App By OOZOU**

Estimate My App is a web-based system developed by OOZOU. The functionality of this web-based system is to calculate the effort estimation required to build a web or mobile application (OOZOU, 2020). The system provides three types of software platform which is desktop, android and iOS in order to make calculation.

The system provides functionality such as User Interface (UI) levels, users and account that allows user to sign up or sign in using variation of social media. For the generated content, the system provides option such as dashboard, activity feeds, user profile, transaction emails etc. This function gives insight to the developers before developing an application. There are also features like app icon design, cloud syncing, barcode scanner etc. The overall functionality and features of the system including the estimation are based on oozou rates.

All cost estimates are intended to be indicative of development costs and timescales only and are exclusive of all hosting costs, paid services or purchased assets of any kind. Table 2.4 shows all the functions and features with description in Estimate My App. The features consist of size of the application, level of UI, user and account, user generated content, mobile specific feature, dates and location, social and engagement, billing and ecommerce, admin, feedback and analytics, external APIs and integration, and security

**Table 2.4** Estimate My App features and description

|  |  |  |
| --- | --- | --- |
| Features | | Description |
| Size of application | Small | Application contains 2-3 key features screen (excluding any static contents, sign in, sign up etc.) |
| Medium | Application contains 4-7 key features screen (excluding any static contents, sign in, sign up etc.) |
| Large | Application contains 8-12 key features screen (excluding any static contents, sign in, sign up etc.) |
| Level of UI | MVP | Minimum viable product. Contains only small scale of core features. |
| Basic | Contains only basic features of UI. |
| Polished | Professionalized UI design. Contains some animation and transition. |
| User & Accounts | Email/Password sign up | Registration via email and password. |
| Social Media sign up | Registration via Social Media (Facebook, Twitter, Google+, Github, and LinkedIn account). |
| User generated contents | Activity feed | Provides user’s recent activity. |
| Media uploading | User is able to upload audio, video, and photo content. |
| User profile | User is able to manage and publish their profiles to other users within application. |
| Tags | Allow user to categorize information so that others can find relevant information. |
| Ratings or review | Provides use case of ratings and review such as satisfaction ratings. |
| Media manipulation | User is able to modify audio, video, and photo contents on their devices. |
| Searching | User is able to search for contents. |
| Mobile specific features | App icon design | Professionally designed app icon for multiple device resolution. |

**Table 2.4** Continued

|  |  |  |
| --- | --- | --- |
|  | Cloud synching | Allow user data to be shared between mobile and/or desktop devices. |
| Device sensor data | User is able to use and/or record data from device sensor (Accelerometer, Gyroscope or compass). |
| Barcode or QR code | User is able to scan for barcodes or QR codes. |
| Health data | Collection and/or use user health and activity related data. |
| Apple watch | Develop an Apple Watch counterpart (for iOS application). |
| Dates & location | Calendaring | Display and capture of data in calendar format. |
|  | Display of map data/ geolocation | Shows user the map with data points (venue location, driver location etc.). |
| Display of custom map marker/region | Allowing the user to visually select a map area or customs icon for different location. |
| Bookings | Provides booking feature where user can select start and end dates, managing capacity etc. |
| Social & engagement | Messaging | Allow user within the app to send messages to other account users or a group of users. |
|  | Forums or commenting | Classic forum functionality for account users or simple commenting on information. |
| Social sharing | User is able to share a piece of information in a controlled way on social media account to drive engagement. |
| Push to Facebook open graph | Push content from app directly to facebook open graph. |
| Push notification | Real-time notification between users (unread message counts, notification of editing etc.). |
| Billing & ecommerce | Shopping cart | User is able to browse for product and add them to cart. |
|  | In-app purchase | User is able to purchase additional contents or functionality within the app. |
| Payment information collection | Ability to collect credit card or other payment method from users for use with a 3rd party payment provider. |
| Payment processing | System will process adhoc or regular from users and manage refunds. |

**Table 2.4** Continued

|  |  |  |
| --- | --- | --- |
| Admin, feedback & analytics | Intercom | A leading 3rd party platform for managing user engagement, drip emails, feature announcement etc. |
|  | Usage analytic | Finds out where the users come from and how they use the app (Flurry, mixpanel etc.). |
| Crash reporting | User is able to acknowledge the misfunction in the app. |
| Multilingual support | Provides support for multiple language for application. |
| External APIs and integration | Connect to one or more third party services | Provide a data feeds that needed to integrate with or a partner app. |
|  | SMS messaging | Allow app to send SMS messages. |
| Phone number masking | Calls made through application with masked phone numbers. |
| Security | Two-factors authentication | Requires user to provide additional information other than password. |

Figure 2.2 shows Estimate My App first page where user need to determine which platform of application that need to be develop. There are three option available, which are web, iOS, and Android.

Graphical user interface, website, timeline

Description automatically generated

**Figure 2.2** Estimate My App UI (Step 1)

(*Source:* EstimateMyApp.com)

Figure 2.3 shows the estimation cost from Estimate My App. The calculated cost are manpower, scheduling and overall budget cost for an iOS application.

Treemap chart

Description automatically generated with medium confidence

**Figure 2.3** Estimate My App UI (Final result)

(*Source:* EstimateMyApp.com)

The advantage of Estimate My App is developers can choose from variety of feature that is needed for the application development process. Thus, they also can change the rate of every feature. The calculated cost includes manpower, schedule and overall cost for the development.

**2.5.2 App Cost Calculator By DIGITALYA**

App Cost Calculator is a web-based system that is developed by DIGITALYA. This system is used to calculate the overall cost of mobile application development by using questionnaire to retrieve the information of functionality needed (DIGITALYA, 2021).

App Cost Calculator helps developers to choose the necessary function that is needed to develop an application. The features and functionality include types of users, in-app features, generated revenue, and competitive advantage. The calculation of estimation consists of duration and overall cost of development. The worst- and best-case scenario are also included for both criteria.

Table 2.5 shows all the function and features description in App Cost Calculator. The features consist of user of product, features include in MVP, generated revenue, and technical and competitive advantages:

**Table 2.5** App Cost Calculator features and description

|  |  |  |
| --- | --- | --- |
| Features | | Description |
| User of product | Visitor | An anonymous user/guest will have limited permission in accessing the application. |
| Registered user | User will have to log in and manage the profile. He/she will have access to all content based on subscription. |
| Moderators | A moderator can perform administrative tasks like approve, create or publish content, ad or delete sections. |
| Administrators | The administrator can assign application-specific roles, gives different permission to users, and make change to the app. |
| Features include in MVP | Dashboard | A dashboard provide full overview of the metrics tracked by application. |
| Activity feed | An activity feed provide users a list of the most interesting and recent activities performed by other users and newly published content. |
| Advanced analytics | Discovering deeper insight about users, make predictions, or generate recommendation. |
| Calendar | Enable users to plan activities/events and have a complete overview of their schedule. |
| E-learning | Enable users to view their curriculum, get in contact with trainer, and access other online materials like e-books, videos, and audio files. |
| Media | Enable users to interact with images, audio, and video contents. |
| Maps/geolocation | Identification of user geographic location. |
| Booking | Provide a complete analysis of resources and help users prevent overbookings. |
| Events | Enable user to plan and design different events, invite other users to participate or interact with contents. |
| Video streaming | Enable users to record live videos or to save and download video content to watch/rewatch it later. |
| Messaging | Enable user to chat with other users within the application. |
| Rating system | Users can give grades or leaves review about their experiences. |
| Invitation system | Users can share developer website or application by inviting friends or colleagues. |

**Table 2.5** Continued

|  |  |  |
| --- | --- | --- |
|  | CMS | Content Management System enable content modification without developer’s help. |
| Generated revenue | Recurring payments | Users authorize the merchant to pull funds from their accounts automatically at regular intervals for the goods and services provided. |
| Shopping cart system | Enable users to browse and add product to shopping cart. |
| Upfront or custom fees | Initial sum of money owed in a purchase or business venture. |
| Technical competitive advantages | Artificial intelligent (AI) | Integrate AI algorithm in order to understand and interpret user’s needs by using technologies like predictive analytics, speech recognition, computer vision, language understanding etc. |
| Virtual/Augmented reality (VR/AR) | Enhance natural environment and offer enriched experience to users by integrating VR/AR technologies. |
| Internet of things (IoT) | Gives users opportunity to have a constant overview of their devices. |
| Process automation (RPA) | Empower users to automate repetitive tasks. |

Graphical user interface, application, Word

Description automatically generatedFigure 2.4 shows the overall estimation cost from App Cost Calculator. The calculated cost include scheduling and overall budget cost of development.

**Figure 2.4** App Cost Calculator UI (Final Result)

(*Source:* https://digitalya.co/app-cost-calculator/)

The advantage of using App Cost Calculator is the scheduling and overall cost calculation includes the best- and worst-case scenario. Developers can manage the resources accordingly by knowing how much time taken for their application development process. However, there are important feature that doesn’t provided in the system that considered to be an important criterion such as size of the application and security details.

**2.5.3 How Much To Make An App By Z1**

How Much To Make An App is a web-based system that is developed by Z1. This system is built to assist mobile application developers to produce an cost estimation of mobile application. These estimates are based on experience of years designing and building amazing digital products such as mobile apps, web platforms, and marketing websites. The budgets are more meant to be ballparks based on averages for hitting a high-quality standard for the final product (Z1, 2021).

The functionality and features that are available include user authentication, user profile feature, revenue, feedback, APIs, app design, and icon design. However, estimation can vary greatly depending on how you decide to execute on a feature set and what your goal with the end product is (i.e., first version as a test, to be a best-in-class product right now, etc.). But these are the ballpark average budgets we've seen necessary to achieve top quality results (Z1, 2021).

Table 2.6 shows all the function and features in How Much To Make An App. The feature consists of user authorization, personal profile, generated revenue, rating system, APIs, UI, and app icon design:

**Table 2.6** How Much To Make An App available features.

|  |  |  |
| --- | --- | --- |
| Features | | Description |
| User authorization | Email | User login via email and password. |
| Social | User login via social media. |
| Personal profile | Yes | User can create personal profile to view their activities. |
| No | User cannot create personal profile. |
| Generated revenue | Upfront cost | Initial sum of money owed in a purchase or business venture. |
| In-app purchase | User is able to purchase additional contents or functionality within the app. |
| Free | Users doesn’t need to make any kind of payment. |
| Rating system | Yes | Provide rating system for users to share their experiences. |
| No | Users are unable to make rating and review of the overall services. |
| Application Programming Interfaces (APIs) | Yes | Enable users to connect the app with developer’s website. |
| No | User is unable to connect the app with developer’s website. |
| User interface (UI) | Bare-bones | Prioritize the task at hand and all variables that it has to deal with. |
| Stock | Simple graphical user interface (GUI) |
| Beautiful | Professionalized UI design. Contains some animation and transition. |
| App icon design | Yes | Design of app icon is necessary. |
| No | Design of app icon is unnecessary. |

Graphical user interface, application

Description automatically generatedFigure 2.5 shows the estimated cost from How Much To Make An App. The calculated costs is overall cost of development.

**Figure 2.5** How Much To Make An App (Final Result)

(*Source:* <http://howmuchtomakeanapp.com/>)

How Much To Make An App is the recommended system for developers to acquire their rough budget. Therefore, to get the overall insight of developing a mobile application and get the minimum requirement the system provide an easy step by step way by listing the basic feature for development process. However, the scheduling and manpower cost is not calculated by the system.

**2.5.4 Mobile App Development Cost Calculator By BUILDFIRE**

Mobile App Development Cost Calculator is a web-based system developed by BUILDFIRE.

Table 2.7 shows the functions and features in Mobile App Development Cost Calculator. The features consist of content, social, fitness, media, forms, ecommerce, loyalty, customs, users, user contents, user engagement, integration, external APIs, generated revenue, maintenance and hosting, basis maintenance, setup and design packages:

**Table 2.7** Mobile App Development Cost Calculator features and description

|  |  |  |
| --- | --- | --- |
| Features | | Description |
| Content | Events | Include a list of events with detailed information about the time, place and event. |
| Map & location | Simple maps that display a place, allows for pins and short description and basic calculation of distance. |
| Weather | Get real-time weather forecasts for the users current location. |
| Host documents | Manage, organizes, and share files, image and video with users. |
| Manage task and checklist | Track and manage tasks and to-do lists for the individual or for the team. |
| Scheduled content | Release new content by scheduling the post by time and day. |
| Flash cards | Release new content by scheduling the post by time and day. |
| Social | Chat and message board | Create message boards and chatrooms where user can like and comment. |
| Social media | Link Facebook and twitter to the application. |
| Fitness | Weight tracker | Users can set fitness goal, track their progress and even monitor their hydration, exercise and diet, as well as send customized encouragement messages. Screen reader support enabled. |
| Media | Stream video | Video streaming with simple controls, pause, play, volume controls (Youtube, vimeo etc.). |
| Stream audio | Audio streaming with simple controls, pause, play, volume controls (Souncloud etc.). |
| Image gallery | Organize images in gallery for users to browse and view. |
| Forms | Forms | Capture information by allowing users to submit forms. |
| Quizzes and surveys | Increase users engagement by creating quizzes, QnA system, and customer support surveys. |
| Book appointment and reservation | Scheduling meeting and appointment made easy. |
| Ecommerce | E-commerce | Ability to list items (image, description, price etc.), shopping cart feature, checkout, email confirmation, purchase history etc. with payment included. |

**Table 2.7** Continued

|  |  |  |
| --- | --- | --- |
|  | QR code scanning and online purchase | Scan and save QR code to mobile devices. |
| Create coupons | Provide coupons for users that are filterable, searchable and mappable. |
| Loyalty | Loyalty points | Reward users with points that can be used access deals or benefits inside the app. |
| Customer feedback | Users can provide feedback through a star rating system and custom commenting. |
| Custom | Small feature | Feature in the app that is straightforward and around 1-2 screens deep. |
| Bigger feature | A feature that is straightforward but is larger in size and may have some business logic that accompanies it. |
| Feature with a backend | Develop a whole section of the app that will leverage a custom backend or possibly integrate with our own existing backend. |
| Building a system | A team of consultant and not just robotic developers to help achieve goals. |
| Users | Email | Login through email and passwords combination. Reset password option is available. |
| Social media | Login through social media (Facebook, twitter, etc.). |
| Single Sign On (SSO) | Login through set of login credentials that are used across multiple application. |
| User contents | User profile info | Details about the users (image, bio, email, etc.). |
| Segment users | Segment users into categories in order to have a better picture of user base (Admin, VIPs, Employees, etc.). |
| User activity | Track of users activities (download, viewed contents, etc.). |
| User engagement | Send push notification | Alert users with push notification to let them know about new updates. |
| Subscription to newsletter | Allow users to subscribe to newsletters that will keep them up to date with everything new that’s going on the app (deals, articles, news, updates, etc.) |
| Allow the users to chatting | Allow users to communicate with other user in real-time environment. |
| Integration | Camera | Take photo or upload image from camera rolls. |

**Table 2.7** Continued

|  |  |  |
| --- | --- | --- |
|  | Geo-fences | Create a virtual fence around an area to trigger specifics action (notify users) in the app when users enters or exits the geo-fences. |
| Bluetooth | To communicate and transfer data between devices. |
| Gyroscope | Help determine orientation |
| Accelerometer | Measure non-gravitational acceleration. |
| External APIs integration | Public APIs | Integrate with third party APIs to send or retrieve data. |
| Backend system | Application is able to communicate with backend system (servers, database). |
| Generated revenue | Upfront cost | Users will be charged once or pay per monthly subscription fee. |
| In-app purchase | Users will be able to purchase premium feature in the app that will improves experience. |
| Free | Free for everyone to enjoy. |
| Maintenance and hosting | Enterprise | Premium support and functionality to maximize the business value of your app ($499/month). |
| Business | Additional capabilities to get more out of your app ($349/month). |
| Growth | The basics needed to get you started with a business app ($189/month). |
| Basis maintenance | Android maintenance | Keep your android app up to date with the latest updates and make enhancements to the application. |
| iOS maintenance | Keep iOS application up to date with the latest updates and make enhancements to the application. |
| Server maintenance | Monitor uptime and maintain redundancy. |
| Setup and design packages | Premium setup | App Design Mock-ups   * 2 app design options * 2 revisions   App Size / Setup   * Small app * 10-15 screens * 3 hours of content entry |

**Table 2.7** Continue

|  |  |  |
| --- | --- | --- |
|  | Signature setup | App Design Mock-ups   * 2 app design options * 2 revisions   App Size / Setup   * Small app * 10-15 screens * 3 hours of content entry   Expert Assistance   * 2 – 60 minute Expert Assistance Sessions   Marketing Materials   * App Phone Mock-ups * Social media ads +copy   Integrations  Zapier |
| Advance signature setup | App Design Mock-ups   * 2 app design options * 2 revisions   App Size / Setup   * Small app * 10-15 screens * 3 hours of content entry   Expert Assistance   * 2 - 60 minute Expert Assistance Sessions   Marketing Materials   * App Phone Mock-ups * Social media ads +copy * Promo Video * Premium App Store Screenshots   Integrations   * Zapier * Server to server API * Segment Analytics   Advanced Functionality   * Customizations to existing features * Access to premium plugins worth $5,000 |

Graphical user interface, application, Teams

Description automatically generatedFigure 2.6 shows the estimation cost for Mobile App Development Cost Calculator. The calculated cost includes overall cost, scheduling and manpower.

**Figure 2.6** Mobile App Development Cost Calculator UI (Final Result)

(*Source:* https://buildfire.com/how-much-to-make-a-mobile-app-calculator/#)

Mobile App Development Cost Calculator provides very details needs for developing application. The requirement satisfies all the necessary needs for developers to get the overview of their project. The calculation for manpower, scheduling and overall cost is provided in the end of the result.

**2.5.5 Comparison of Similar Systems**

Table 2.8 shows the comparison of four similar system that are mentioned in previous section. The feature includes size of application, levels of UI, user & accounts, user generated contents, mobile specific features, dates & location, generated revenue, admin, feedback & analytic, external APIs and integration, security, media, technical and competitive advantage and basis maintenance.

**Table 2.8** Comparison of four similar system

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Features** | **Estimate My App** | **App Cost Calculator** | **How Much To Make An App** | **Mobile App Development Cost Calculator** | **Score ( /4)** |
| **Size of application** | | | | | |
| Small | / | - | - | / | 2 |
| Medium | / | - | - | / | 2 |
| Large | / | - | - | / | 2 |
| **Levels of UI** | | | | | |
| MVP | / | - | - | - | 1 |
| Basic | / | - | - | - | 1 |
| Polished | / | - | - | - | 1 |
| Bare-bones | - | - | / | - | 1 |
| Stock | - | - | / | - | 1 |
| Beautiful | - | - | / | - | 1 |
| **User & Accounts** | | | | | |
| Email | / | - | / | / | 3 |
| Social media | / | - | / | / | 3 |
| Single Sign On (SSO) | - | - | - | / | 1 |
| Visitor | - | / | - | - | 1 |
| Registered user | - | / | - | - | 1 |
| Moderators | - | / | - | - | 1 |
| Administrators | - | / | - | - | 1 |
| **User Generated Contents** | | | | | |
| Activity feed | / | / | - | / | 3 |
| Media uploading | / | / | - | / | 3 |
| User profile | / | - | / | / | 2 |
| Tags | / | - | - | - | 1 |
| Media manipulation | / | / | - | / | 3 |
| Searching | / | - | - | - | 1 |
| Events | / | / | - | / | 3 |

**Table 2.8** Continued

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Map & location | / | / | - | / | 3 |
| Weather | / | - | - | - | 1 |
| Host documents | - | - | - | / | 1 |
| Manage task and checklist | - | - | - | / | 1 |
| Scheduled content | - | - | - | / | 1 |
| Flash cards | - | - | - | / | 1 |
| Segment Users | - | - | - | / | 1 |
| Dashboard | - | / | - | / | 2 |
| **Mobile Specific Features** | | | | | |
| App icon design | / | - | / | - | 2 |
| Cloud synching | / | - | - | - | 1 |
| Device sensor data | / | - | - | / | 2 |
| Barcode or QR code | / | - | - | / | 2 |
| Health data | / | - | - | / | 2 |
| Apple watch | / | - | - | - | 1 |
| **Dates & Location** | | | | | |
| Calendaring | / | / | - | / | 3 |
| Display of map data/ geolocation | / | / | - | / | 3 |
| Display of custom map marker/region | / | / | - | / | 3 |
| Bookings | / | / | - | / | 3 |
| **Social & Engagement** | | | | | |
| Messaging and Chatting | / | / | - | / | 3 |
| Forums or commenting | / | - | - | / | 2 |
| Social sharing | / | - | - | / | 2 |
| Push to Facebook open graph | / | - | - | - | 1 |
| Push notification | / | - | - | / | 2 |
| Subscription to newsletter | - | - | - | / | 3 |

**Table 2.8** Continued

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Generated Revenue** | | | | | |
| Shopping cart | / | / | - | - | 2 |
| In-app purchase | / | - | / | / | 3 |
| Payment information collection | / | - | - | - | 1 |
| Payment processing | / | - | - | - | 1 |
| QR code scanning and online purchase | - | - | - | / | 1 |
| Upfront cost | - | / | / | / | 3 |
| Create coupons | - | - | - | / | 1 |
| Recurring payments | - | / | - | - | 1 |
| Subscriptions | - | / | - | / | 2 |
| Free | - | / | / | / | 3 |
| **Admin, Feedback & Analytics** | | | | | |
| Intercom | / | - | - | - | 1 |
| Usage analytic | / | - | - | / | 2 |
| Crash reporting | / | - | - | - | 1 |
| Multilingual support | / | - | - | - | 1 |
| Loyalty points | - | - | - | / | 1 |
| Customer feedback | / | / | / | / | 4 |
| Forms | - | - | - | / | 1 |
| Quizzes and surveys | - | - | - | / | 1 |
| Rating and review | / | / | / | / | 4 |
| **External APIs And Integration** | | | | | |
| Connect to one or more third party services | / | - | - | / | 2 |
| SMS messaging | / | - | - | / | 2 |
| Phone number masking | / | - | - | - | 1 |
| Public APIs | - | - | - | / | 1 |

**Table 2.8** Continued

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Backend system | - | - | - | / | 1 |
| Camera | / | - | - | / | 2 |
| Geo-fences | - | - | - | / | 1 |
| Bluetooth | - | - | - | / | 1 |
| **Security** | | | | | |
| Two-Factor Authentication | / | - | - | - | 1 |
| DOS Protection | / | - | - | - | 1 |
| SSL Certificate Based Security | / | - | - | - | 1 |
| **Media** | | | | | |
| Video Streaming | - | / | - | / | 2 |
| Audio Streaming | - | - | - | / | 1 |
| Image gallery | / | / | - | / | 3 |
| **Technical And Competitive Advantage** | | | | | |
| Artificial Intelligent (AI) | - | / | - | - | 1 |
| Internet Of Thing (IoT) | - | / | - | - | 1 |
| Augmented/Virtual Reality (AR/VR) | - | / | - | - | 1 |
| Process automation (RPA) | - | / | - | - | 1 |
| **Basis Maintenance** | | | | | |
| Android maintenance | - | - | - | / | 1 |
| iOS maintenance | - | - | - | / | 1 |
| Server maintenance | - | - | - | / | 1 |

From the comparison of all four similar systems, Mobile App Development Cost Calculator is the most comprehensive web-based system to calculate the estimation for mobile application development since the system score 56 out of 84 points from overall features that is listed in the table above.

Therefore, features like user and accounts (email, social media), user generated contents (activity feeds, media uploading, media manipulation, events, maps and location), dates and location (display of map data, calendaring, display of custom map region, bookings), generated revenue (in-app purchase, upfront cost, free), admin, feedback and analytics (customer feedback, rating and review), and media (image gallery) is considered as the basic feature that a mobile application should have. This is because from the comparison above, those mentioned features are at least contained in every system with score of minimum 3 out of 4.

In COSMIC FP, the criteria that is mandatory to calculated is the size of application. Regarding the similar system mentioned, there are three type of size measurement, small, medium and large. The four component that is described in section 2.3 which is entry, exit, read, and write is calculated based on the feature provided.

**2.6 Summary**

This chapter describes in detail about software effort estimation. A deeper explanation about two approach of software effort estimation, parametric and non-parametric estimation models. A brief description about Function Point (FP) is explained before moving on to the method. A calculation example is provided together with the explanation. Next section describes more details about COSMIC FP, the method that is used in this project. Then, an explanation about the mobile application is provided. A review about type of mobile application alongside the comparison is described. In the last section of this chapter, a list of similar systems is described together with the comparison and review of each system also provided.

**CHAPTER 3**

**METHODOLOGY**

This chapter conveys the flow of the project method from the beginning phase until the complete project report. The entire flow of this study such as method used and how it is implemented is explained here. Research methodology is a systematic way for solving the research issue by legitimately embracing different advances. Methodology comprehends not just the final system but also the process itself.

**3.1 Software Development Methodology**

Figure 3.1 describes the waterfall model as the software development methodology that briefly shows each phase inclusive of the activities carried out and the developed outcome from the activities.



**Figure 3.1** Waterfall Model

(*Source*: Hughey, 2009)

From Figure 3.1, there are five main phases in waterfall model which are requirements, design, implementation, verification, and maintenance. However, this project will exclude maintenance phase due to the project needs to carry out only the functionality test in the verification phase. All the other phases are included in the project framework.

* 1. **Project Framework**

This section describes the project framework for this project (see Table 3.1). The phases in the project framework will follow the phases in the software development methodology. The objectives are mapped to the phases in the project framework.

|  |  |  |  |
| --- | --- | --- | --- |
| Objective | Phase | Activity | Outcome |
| Objective 1: To identify COSMIC Function Point for mobile application effort estimation. | Requirements | 1. Review on Software Effort Estimation. 2. Review on parametric and non-parametric estimation models. 3. Review on COSMIC Function Point. 4. Review on similar systems. | 1. Problem Statements. 2. Objectives. 3. Scope. 4. Significance. 5. Comparison table for similar systems. |
| Objective 2: To construct a web-based tool to estimate the mobile application effort. | Design | Design on the following diagram:   1. System Architecture 2. Process Flow 3. Use Case Diagram 4. Use Case Description 5. Algorithm Design 6. User Interface Design | 1. System Architecture for web-based calculator. 2. Use Case Diagram for the web-based calculator 3. Four tables of Use Case Description 4. Two User Interface Design |
| Implementation | 1. Hardware requirements (processor, memory, display etc.) 2. Software requirements (frontend, backend). | Mobile Application Effort Estimation |
| Objective 3: To evaluate the functionality of the web-based tool. | Verification | 1. Test Cases Generation | 1. Functionality Test Results |

**Table 3.1** Project Framework

From Table 3.1, objective 1 can be achieved by implementing requirement phase. The outcome for objective 1 is the definition of problem statements, objectives, scope, significance, and the comparison of similar systems. Objective 2 can be achieved by implementing design and implementation phase. The outcome for objective 2 is mobile application effort estimation tool. For objective 3, verification phase is implemented. The outcome for objective 3 is the functionality test result.

**3.3 Phase 1 Requirements**

This phase is to gather all the information regarding software effort estimation for mobile application development. To further understand this, research on software effort estimation has been done and further focused into effort estimation for mobile application. Then, a review on techniques that is used for effort estimation is made to get a clarification about the project. The techniques are from parametric and non-parametric estimation models. Method of gathering this information is through online publications e.g., journal, articles, proceedings, etc.

To get a detailed information on the method used for calculation, a review on COSMIC Function Point (COSMIC FP) is made. The information gathered are calculation examples, formula and steps to calculate the effort estimation. Then, an analysis on mobile application is made to obtain information about types of mobile app and the development of mobile application. A review on four similar systems is made and a comparison table is constructed to identify the common features of a mobile application. The common features include user and account, user generated content, dates and location, generated revenue, feedback and analytic, and media.

After gathering all the necessary information, further reading was done with the objective of achieving a better understanding regarding the study. The problem statement can be identified as well as the objectives of the project. The outcomes of this phase have produced the problem statements, objectives, scope and the significance of the project.

**3.4 Phase 2 Design**

This phase aims to design the proposed system which includes the system architecture, use case diagram, use case description, and user interface design.

**3.4.1 System Architecture**

Figure 3.2 illustrates the system architecture for the proposed system. It shows the interaction layout between application components, user interface, and database.

Graphical user interface, diagram

Description automatically generated

**Figure 3.2** System Architecture

(*Adapted:* LITSLINK, 2021)

From Figure 3.2, the user interface is displayed when user enters the system, it allows the user to input data into the system. The data include entry, exit, write, and read as the size of the software. At the backend, the database acts as a storage for the system. The calculation occurs in the backend of the system for the effort and duration of the development. The result of the estimation is sent to the frontend for displaying purposes.

**3.4.2 Use Case Diagram**

Figure 3.3 shows the use case diagram of proposed web-based calculator. It has a single actor, the User, as well as eight use cases. The four use cases, include input data, calculate effort, calculate schedule, and display output are used to represent the complete functionality of the proposed tool.

Diagram

Description automatically generated

**Figure 3.3** Use Case Diagram

The detailed processes from the use case diagram in Figure 3.3 are described further in the following tables. Table 3.2, Table 3.3, Table 3.4, and Table 3.5 show the use case descriptions for entering input data, calculating effort, calculating schedule, and displaying output respectively.

**Table 3.2** Input Data Use Case Description

|  |  |
| --- | --- |
| Use Case ID | UC 001 |
| Use Case | Input data |
| Purpose | To allow user to select mobile application features |
| Actor | User |
| Trigger | The user clicks on multiple features that are provided |
| Precondition | The user enters the system |
| Main Flow | 1. The user opens the system 2. The user selects the available feature to be included |
| Exception Flow | None |

**Table 3.3** Calculate Effort Use Case Description

|  |  |
| --- | --- |
| Use Case ID | UC 002 |
| Use Case | Calculate effort |
| Purpose | To calculate the estimation of manpower needed |
| Actor | User |
| Trigger | The user selects on each feature |
| Precondition | The user enters the system |
| Main Flow | 1. The user selects on available features 2. System will calculate the cost of manpower using the designed algorithm |
| Exception Flow | The user doesn’t select any features. Calculation is null. Use case ends. |

**Table 3.4** Calculate Schedule Use Case Description

|  |  |
| --- | --- |
| Use Case ID | UC 003 |
| Use Case | Calculate schedule |
| Purpose | To calculate the estimation of development schedule |
| Actor | User |
| Trigger | The user selects on each feature |
| Precondition | The user enters the system |
| Main Flow | 1. The user selects on available features 2. System will calculate the development schedule using the designed algorithm |
| Exception Flow | The user doesn’t select any features. Calculation is null. Use case ends. |

**Table 3.5** Display Output Use Case Description

|  |  |
| --- | --- |
| Use Case ID | UC 004 |
| Use Case | Display output |
| Purpose | To display the overall estimation cost for mobile application development |
| Actor | User |
| Trigger | The user clicks on submit button |
| Precondition | The user completes the selection of available features |
| Main Flow | 1. System check for any unselected feature. 2. COSMIC Function Point algorithm will calculate the estimation cost for selected features.   System will display the estimation output (effort, schedule, and overall cost) |
| Exception Flow | 1. The user doesn’t select any features. Calculation is null. No output is displayed. Use case ends. 2. Unselected feature will not calculated by the system. |

**3.4.3 COSMIC Function Point Algorithm**

The method for this project is COSMIC Function Point (COSMIC FP). The calculation is based on four data movements, entry, exit, write, and read. The algorithm and steps to obtain the COSMIC FP are as follows:

Step 1: Input data from user

Step 2: Identification of data movement

Step 2.1: Identification of entry data

Step 2.2: Identification of exit data

Step 2.3: Identification of write data

Step 2.4: Identification of read data

Step 3: Calculate size of project using the following COSMIC FPP formula:

Step 4: Calculate the effort (per month) estimation using the formula as follow:

Effort (per month) = Size / 2

Step 5: Set the average duration to complete the project

Step 6: Calculate overall effort estimation using the formula as follow:

Total Effort = Effort (per month) / duration

Step 7: Set the average cost for one software developer

Step 6: Calculate the total cost using the following formula

Total Cost = Total Effort \* Average Cost

**3.4.4 User Interface Design**

Figure 3.4 shows the main interface of the proposed system. It consists of the common features obtained from the comparison of similar systems in the previous chapter. The web-tool contains features such as user and account, feedback and analytics, generated content, generated revenue, social and engagement, and date and location. The user can select multiple features available for the application's development.

Graphical user interface, application, Teams

Description automatically generated

**Figure 3.4** Input User Interface Design

From Figure 3.4, each feature selection from the user, the system will auto calculate the size based on four data movements (entry, exit, write, and read). For example, the customer feedback feature involves three data movements which are entry, exit, and write movement. Thus, the feature will be counted as 3 COSMIC Function Points (CFP).

Figure 3.5 shows the output user interface of the proposed system. The proposed system will display the estimated cost of mobile application development, including the schedule, effort, and overall cost.

Graphical user interface, text, application, chat or text message

Description automatically generated

**Figure 3.5** Output User Interface Design

From Figure 3.5, the schedule indicates the estimated duration for a mobile application development. Then, the effort shows the manpower (programmers) involved in a month for the project. Lastly, overall cost indicates the development cost which derived from Total Effort \* Average Cost.

**3.5 Phase 3 Implementation**

This section describes the implementation of the project design. It includes software and hardware requirements that is used to develop this project.

Table 3.6 shows the software requirement for this project. The requirement includes application software for frontend and backend. The software that is used for frontend is HTML, CSS, and JavaScript while for the backend are PHP and MySQL.

**Table 3.6** Software requirement

|  |  |  |
| --- | --- | --- |
| **Software** | | **Description** |
| Backend | PHP | PHP is a backend development language only. PHP is the programming language used for creating dynamic web pages and is mostly used in web applications. |
| MySQL | MySQL is the database used for storing information and data and is linked with PHP application. |
| Frontend | HTML | HTML is a markup language used for the design of the user interface. |
| CSS | CSS is a designed language used to decorate the user interface by applying the code with HTML. |
| JavaScript | JavaScript is a programming language used for validation of the user interface. |

Table 3.7 shows the hardware requirement for this project. The requirement includes device specification that is required to develop the project. The specification comprises of processor, RAM, operating system, screen display and resolution and system type.

**Table 3.7** Hardware requirement

|  |  |
| --- | --- |
| **Hardware Requirement** | |
| Processor | Intel(R) Core (TM) i5-7200U CPU @ 2.50GHz 2.70 GHz |
| RAM | 8 GB |
| OS | Windows 10 Home Single Language |
| Display | 15.60 inches |
| 1920x1080 pixels |
| System Type | 64-bit operating system, x64-based processor |

**3.6 Phase 4 Verification**

This phase is to verify the functionality of the web-based tool. Accuracy test is evaluated by comparison of counting the estimation cost manually and counting using the web-based calculator. Table 3.8 shows the functionality test case for the web-based calculator.

**Table 3.8** Functionality test case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Scenario** | **Test Case** | **Pre-Condition** | **Test Steps** | **Test Data** | **Expected Result** |
| TC\_Platform\_01 | Verify the platform feature for clickable button. | Select type of platform feature. | 1. Enter the system. | 1. Navigate to the platform section. 2. Click all the button provided in the section. | No Data | All the buttons should be clickable. |
| TC\_Size\_02 | Verify the size feature for clickable button. | Select type of size feature. | 1. Enter the system | 1. Navigate to the size section. 2. Click all the button provided in the section. | No data | All the buttons should be clickable. |
| TC\_UnA\_03 | Verify the user and account feature for clickable button. | Select type of user and account feature. | 1. Enter the system | 1. Navigate to user and account section. 2. Click all the button provided in the section. | No data | All the buttons should be clickable. |
| TC\_FnA\_04 | Verify the feedback and analytic feature for clickable button. | Select type of feedback and analytic feature. | 1. Enter the system | 1. Navigate to feedback and analytic section. 2. Click all the button provided in the section | No data | All the buttons should be clickable. |
| TC\_GC\_05 | Verify the generated content feature for clickable button. | Select type of generated content feature. | 1. Enter the system. | 1. Navigate to generated content section. 2. Click all the button provided in the section | No data | All the buttons should be clickable. |
| TC\_GR\_06 | Verify the generated revenue feature for clickable button. | Select type of generated revenue feature. | 1. Enter the system. | 1. Navigate to generated revenue section. 2. Click all the button provided in the section. | No data | All the buttons should be clickable. |
| TC\_SnE\_07 | Verify the social and engagement feature for clickable button. | Select type of social and engagement feature. | 1. Enter the system. | 1. Navigate to social and engagement section. 2. Click all the button provided in the section. | No data | All the buttons should be clickable. |

**Table 3.8** Continued

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC\_DnL\_08 | Verify the dates and location feature for clickable button. | Select type of date and location feature. | 1. Enter the system. | 1. Navigate to dates and location section. 2. Click all button provided in the section. | No data | All the buttons should be clickable. |
| TC\_Submit\_09 | Verify the submit button functionality. | Submit the form for calculation process. | 1. Enter the system 2. Select the features that are provided in the user interface. | 1. Click the submit button. | Selected features. | The system should navigate to the next page for displaying estimation cost. |
| TC\_Reset\_10 | Verify the reset button functionality. | Reset the form. | 1. Enter the system 2. Select the features that are provided in the user interface. | 1. Click the reset button. | No data | All the selection made should be reset to null. |

**3.7 Summary**

This chapter fully covers the proposed system in terms of the research framework, which outlines the phases of requirement, design, implementation, and verification. For requirement phase, identification of problem statement, objectives, scope, significance, and the comparison of similar system are defined to get an overview of the proposed system.

Then, in design of system architecture, use case diagram with the description, and the interface are made for the next phase which is design phase. For implementation phase, the requirement of software and hardware needed to build the system is defined. A functionality test case is made for the verification phase. The test case shows the functionality of every button and navigation of the system.

**REFERENCES**

Ali, A., & Gravino, C. (2021). Improving software effort estimation using bio-inspired algorithms to select relevant features: An empirical study. Science of Computer Programming, 205. <https://doi.org/10.1016/j.scico.2021.102621>.

Institute of Electrical and Electronics Engineers. (n.d.). CEIT 2018 : 2018 6th International Conference on Control, Engineering & Information Technology (CEIT) : 25-27 October 2018, Istanbul, Turkey.

Bluemke, I., & Malanowska, A. (2021). Software Testing Effort Estimation and Related Problems. In ACM Computing Surveys (Vol. 54, Issue 3). Association for Computing Machinery. <https://doi.org/10.1145/3442694>.

Scholarworks, S., & Roark, K. R. (n.d.). Strategies in Software Development Effort Estimation. <https://scholarworks.waldenu.edu/dissertations>.

Kumar, P. S., Behera, H. S., Anisha Kumari, K., Nayak, J., & Naik, B. (2020). Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades. In Computer Science Review (Vol. 38). Elsevier Ireland Ltd. <https://doi.org/10.1016/j.cosrev.2020.100288>.

Kaur, A., & Kaur, K. (2019). A COSMIC function points based test effort estimation model for mobile applications. Journal of King Saud University - Computer and Information Sciences. <https://doi.org/10.1016/j.jksuci.2019.03.001>.

Rankovic, N., Rankovic, D., Ivanovic, M., & Lazic, L. (2021). COSMIC FP method in software development estimation using artificial neural networks based on orthogonal arrays. Connection Science. <https://doi.org/10.1080/09540091.2021.1981824>.

Tanabata, K., Hazeyama, A., Yamada, Y., & Furukawa, K. (2021). Proposal of an evaluation method of individual contributions using the function point in the implementation phase in project-based learning of software development. Procedia Computer Science, 192, 1524–1531. <https://doi.org/10.1016/j.procs.2021.08.156>.

Ezghari, S., & Zahi, A. (2018). Uncertainty management in software effort estimation using a consistent fuzzy analogy-based method. Applied Soft Computing Journal, 67, 540–557. <https://doi.org/10.1016/j.asoc.2018.03.022>.

<https://project-management.info/parametric-estimating/#0-what-is-parametric-estimating>

<https://www.sourcesoftsolutions.com/reasons-that-explain-the-increasing-demand-of-mobile-app-development/>

<https://www.scopemaster.com/introduction-to-cosmic-function-points/>

<https://www.tutorialspoint.com/estimation_techniques/estimation_techniques_function_points.htm>

<https://www.techopedia.com/definition/30911/function-point-fp>

<https://www.techtarget.com/>

<https://estimatemyapp.com/>

<https://digitalya.co/app-cost-calculator/>

<http://howmuchtomakeanapp.com/>

<https://buildfire.com/how-much-to-make-a-mobile-app-calculator/>

<https://www.pmi.org/>

<https://www.investopedia.com/>

<https://clevertap.com/blog/types-of-mobile-apps/>

<https://www.researchgate.net/>