REMOTE PARENTAL CONTROL FOR CHILD ACCESS TO INTERNET USAGE/ CONTENT

TAN ROCHER

UNIVERSITI TUNKU ABDUL RAHMAN

REMOTE PARENTAL CONTROL FOR CHILD ACCESS TO INTERNET USAGE/ CONTENT

TAN ROCHER

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2024

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : 

Name : Tan Rocher

ID No. : 020827-01-1460

Date : 24/9/2024

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"REMOTE PARENTAL CONTROL FOR CHILD ACCESS TO INTERNET USAGE/ CONTENT"** was prepared by **TAN ROCHER** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	
Supervisor	:	Dr. Chew Kuew Wai
Date	:	25/9/2024

Signature	:	
Co-Supervisor	:	Beh Hooi Ching
Date	:	25/9/2024

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Dr. Chew Kuew Wai and co-supervisor, Ms Beh Hooi Ching for their invaluable advice, guidance and their enormous patience throughout the development of the research on Remote Parental Control Application.

In addition, I would also like to express my gratitude to my loving parents and friends for their unwavering support and encouragement.

ABSTRACT

In today's digital age, the internet plays an important role in communication, education, and entertainment, but it also brings significant challenges. However, the increasing use of Internet, especially during the Movement Control Order (MCO) period, has increase the reliance on Internet with raising the concerns of parents about their children's internet usage. The parents are worried about their children's online safety such as exposure to inappropriate content, cyber bullying and spend too much time on screen. Therefore, this project addresses these problems by developing an Android-based mobile application with a remote parental control system, which enables parents to remotely monitor and manage their children's internet usage, creating a safe and rich online environment for their children. The application includes features like screen time management, content filtering, location tracking, an SOS function for emergency and rewards claiming. Developed using the Flutter framework with Dart programming language for the front-end to implement the user interface and Firebase for the back end, which is used for clouding, managing and storing the data. This project followed a phased Rapid Application Development (RAD) approach to ensure the efficient and user-centered development. Through a User Acceptance Test (UAT), the system was evaluated for usability and effectiveness which indicated that the application effectively meets parents' needs for monitoring and controlling their children's online activities. Most respondents reported satisfaction with the application's user-friendly interface and features, although some suggested future improvements can focus on enhancing the beauty of the interface and simplifying the functions. Overall, the project has successfully provided a reliable solution for solving parents' concerns about children's online safety and promoting children's responsible use of the Internet.

TABLE OF CONTENTS

DECLARATION **i**
APPROVAL FOR SUBMISSION **ii**
ACKNOWLEDGEMENTS **iv**
ABSTRACT **v**
TABLE OF CONTENTS **vi**
LIST OF TABLES **x**
LIST OF FIGURES **xiv**
LIST OF SYMBOLS / ABBREVIATIONS **xx**
LIST OF APPENDICES **xxi**

CHAPTER

1 INTRODUCTION **1**
1.1 General Introduction 1
1.2 Importance of the Study 2
1.3 Problem Statement 3
1.4 Aim and Objectives 5
1.5 Proposed Solution 5
1.6 Project Approach 6
1.6.1 Development Approach 6
1.7 Scope and Limitation of the Study 6
1.7.1 System Scope 6
1.7.2 Target User 7
1.7.3 Application Features 7
2 LITERATURE REVIEW **9**
2.1 Introduction 9
2.2 Review Existing Similar System 10
2.2.1 Qustodio 10
2.2.2 Mobicip 15
2.2.3 Google Family Link 20

		2.2.4 Bark	25
	2.3	Comparison	29
	2.4	Summary	31
3	**METHODOLOGY AND WORK PLAN**		**32**
	3.1	Introduction	32
	3.2	Rapid Application Development (RAD)	32
		3.2.1 Planning	33
		3.2.2 Analysis and Design	35
		3.2.3 Implementation and Testing	35
		3.2.4 Closing	37
	3.3	Software Development Tools	37
		3.3.1 Axure RP10	37
		3.3.2 Visual Studio Code	38
		3.3.3 Android Studio	38
		3.3.4 Flutter	38
		3.3.5 Firebase	38
	3.4	Work Breakdown Structure	39
	3.5	Gantt Chart	43
		3.5.1 The Project Schedule’s Overview	43
		3.5.2 Planning Phase	44
		3.5.3 Anaylsis and Design Phase	44
		3.5.4 Implementing and Testing Phase	44
		3.5.5 Closing Phase	47
	3.6	Summary	47
4	**PROJECT SPECIFICATION**		**48**
	4.1	Introduction	48
	4.2	Requirements Specification	48
		4.2.1 Functional Requirement	48
		4.2.2 Non-functional Requirement	49
	4.3	Use Case Modelling	50
		4.3.1 Use Case Diagram	50
		4.3.2 Use Case Description	50
	4.4	Prototype interface	65
	4.5	Summary	89

5 DESIGN AND IMPLEMENTATION 91

5.1 Diagrams 91

5.1.1 Entity Relationship Diagram (ERD) 91

5.1.2 Class Diagram 93

5.1.3 Interface Flow Diagram 97

5.2 Connection Setup 99

5.2.1 Software Setup 99

5.2.2 Third-party Setup 101

5.3 Screenshot and Code Snippet of Application 103

5.3.1 Parents' application 103

5.3.2 Children's Application 127

6 TESTING 137

6.1 Introduction 137

6.2 Unit Test 137

6.2.1 Test Case 001: Check the user authentication for parent login feature 137

6.2.2 Test Case 002: Check the Sign Up feature for parents' device 140

6.2.3 Test Case 003: Check the forget password feature for parents' device 145

6.2.4 Test Case 004: Check parent user profile feature 146

6.2.5 Test Case 005: Check the Dashboard for parent's device 149

6.2.6 Test Case 006: Check the Time Limit Feature for parents' device 150

6.2.7 Test Case 007: Check the Content Filtering Feature for parents' device 151

6.2.8 Test Case 008: Check the Location Tracking Feature for parents' device 153

6.2.9 Test Case 009: Check the Child Login 156

6.2.10 Test Case 0010: Check the Reward Feature for children's device 159

6.2.11 Test Case 011: Check the SOS Feature for children's device 160
6.3 User Acceptance Test (UAT) 162
6.4 Summary 171
7 CONCLUSION AND RECOMMENDATIONS 173
7.1 Conclusion 173
7.2 Project Challenges and Solution 173
7.3 Future Improvement and Recommendations 174
REFERENCES 176
APPENDICES 178

LIST OF TABLES

Table 2.1:	Feature and Interface of Qustodio	12
Table 2.2:	Feature and Interface of Mobicip	17
Table 2.3:	Feature and Interface of Google Family Link.	22
Table 2.4:	Feature and Interface of Bark	27
Table 2.5:	Comparison each system with advantages and disadvantages.	29
Table 2.6:	Comparison between existing system and proposed project.	30
Table 4.1:	User Case Description of Login Account	50
Table 4.2:	Use Case Description of Sign up Account	52
Table 4.3:	Use Case Description of Modify Profile.	54
Table 4.4:	Use Case Description of View Activity Summary Report.	55
Table 4.5:	Use Case Description of Set Limit Time.	56
Table 4.6:	Use Case Description of Block Specific Websites	57
Table 4.7:	Use Case Description of Track Location	58
Table 4.8:	Use Case Description of Save Marked Place.	60
Table 4.9:	Use Case Description of Receive Notification.	61
Table 4.10:	Use Case Description of View Time Spent.	62
Table 4.11:	Use Case Description of Send SOS.	63
Table 4.12:	Use Case Description of Claim Rewards	64
Table 5.1:	Child’s Class Diagram Description	93
Table 5.2:	Parent’s Class Diagram Description	95
Table 6.1:	Verify parent login with valid credentials	137
Table 6.2:	Verify parent login with invalid email	138

Table 6.3:	Verify parent login with incorrect password	138
Table 6.4:	Verify login with empty email field	139
Table 6.5:	Verify login with empty password field	139
Table 6.6:	Verify login with incorrect format email	139
Table 6.7:	Verify login with incorrect format password	140
Table 6.8:	Verify Sign Up with valid details	140
Table 6.9:	Verify Sign Up with invalid email format	141
Table 6.10:	Verify signup with incorrect format password	141
Table 6.11:	Verify signup with no match password	142
Table 6.12:	Verify Sign Up with empty username field	143
Table 6.13:	Verify Sign Up with empty email field	143
Table 6.14:	Verify Sign Up with empty password field	144
Table 6.15:	Verify Sign Up with already registered email	144
Table 6.16:	Verify Sign Up with valid email	145
Table 6.17:	Verify Sign Up with invalid email	146
Table 6.18:	Verify Sign Up with empty email field	146
Table 6.19:	Verify parent profile information is displayed correctly	146
Table 6.20:	Verify parent can change the password	147
Table 6.21:	Verify parent can add a child device	147
Table 6.22:	Verify parent can remove a child device	148
Table 6.23:	Verify parent can log out	148
Table 6.24:	Verify children icon is display and can be selected in dashboard	149
Table 6.25:	Verify the selected child’s time limit and app usage is display in the dashboard	149
Table 6.26:	Verify the navigator can successfully to navigate	150

Table 6.27:	Verify that a time limit can be set for a child device	150
Table 6.28:	Verify that a time limit for today can be display in the dashboard	151
Table 6.29:	Verify that content filtering can be enabled for a child device	151
Table 6.30:	Verify that content filtering categories can be selected and applied	152
Table 6.31:	Verify that the parent can update the blocked categories	152
Table 6.32:	Verify that the parent can view the real-time location of the child device	153
Table 6.33:	Verify that parent can mark a place with a range of meter	153
Table 6.34:	Verify that parent can view the detail of marked place	154
Table 6.35:	Verify that parent can edit the marked place	154
Table 6.36:	Verify that parent can delete the marked place	155
Table 6.37:	Verify that parent can receive notification when child enter or leave the marked place	156
Table 6.38:	Verify child login with valid credentials	156
Table 6.39:	Verify child login with invalid child ID	157
Table 6.40:	Verify child login with empty name or child ID field	157
Table 6.41:	Verify child connect with parent device	158
Table 6.42:	Validate the child connect field	158
Table 6.43:	Verify that child can request the reward after done the task	159
Table 6.44:	Verify that child can claim the reward after parent approve	159
Table 6.45:	Verify that parent can receive the request from the child	160
Table 6.46:	Verify that parent can action on the request from the child	160
Table 6.47:	Verify that child can send SOS to parent	160

Table 6.48: Verify that child can stop the SOS 161

Table 6.49: Verify that parent can receive notification of SOS 161

LIST OF FIGURES

Figure 1.1:	The System Architecture of The Proposed Project.	6
Figure 2.1:	Logo of Qustodio.	10
Figure 2.2:	Feature of Filter Content and Apps for Qustodio.	12
Figure 2.3:	Feature of Set Time Limits for Qustodio.	13
Figure 2.4:	Feature of Monitor Activity for Qustodio.	13
Figure 2.5:	Feature of Track Call and SMS for Qustodio.	14
Figure 2.6:	Feature of Locate Family for Qustodio.	14
Figure 2.7:	Feature of Alert and SOS for Qustodio.	15
Figure 2.8:	Logo of Mobicip.	15
Figure 2.9:	Feature of Screen Time for Mobicip.	17
Figure 2.10:	Feature of Activity Summary for Mobicip.	18
Figure 2.11:	Feature of Social Media Monitor for Mobicip.	18
Figure 2.12:	Feature of App Blocker for Mobicip.	19
Figure 2.13:	Feature of Website Blocker for Mobicip.	19
Figure 2.14:	Feature of Family Locator for Mobicip.	20
Figure 2.15:	Logo of Google Family Link.	20
Figure 2.16:	Feature of Activity Report for Family Link.	22
Figure 2.17:	Feature of Set Screen Time Limits for Family Link.	23
Figure 2.18:	Feature of Website Filtering For Family Link.	23
Figure 2.19:	Feature of App Blocker for Family Link.	24
Figure 2.20:	Feature of Location Tracking for Family Link.	24
Figure 2.21:	Logo of Bark.	25
Figure 2.22:	Feature of Website and App Filtering for Bark.	27

Figure 2.23:	Feature of Content Monitoring for Bark.	27
Figure 2.24:	Feature of Screen Time Management for Bark.	28
Figure 2.25:	Feature of Activity Report for Bark.	28
Figure 2.26:	Feature of Location Tracking for Bark.	29
Figure 3.1:	Diagram of Phase Development Methodology	33
Figure 3.2:	The Project Schdule's Overview	43
Figure 3.3:	Schedule of Planning Phase	44
Figure 3.4:	Schedule of Planning Phase (cont.)	44
Figure 3.5:	Schedule of Analysis and Design Phase	44
Figure 3.6:	Schedule of Implementing and Testing Software Version 1 Phase	45
Figure 3.7:	Schedule of Implementing and Testing Software Version 2 Phase	45
Figure 3.8:	Schedule of Implementing and Testing Software Version 2 Phase (cont.)	45
Figure 3.9:	Schedule of Implementation and Testing Software Version 3 Phase	46
Figure 3.10:	Schedule of Implementation and Testing Software Version 3 Phase (cont.)	46
Figure 3.11:	Schedule of Implementation and Testing Software Version 4 Phase	46
Figure 3.12:	Schedule of Implementation and Testing Software Version 4 Phase (cont.)	46
Figure 3.13:	Schedule of Closing Phase	47
Figure 4.1:	Use Case Diagram of The Proposed Project.	50
Figure 4.2:	Prototype of Onboarding Screen.	66
Figure 4.3:	Prototype of Parents Login Screen.	67
Figure 4.4:	Prototype of Parent Sign up Screen.	68
Figure 4.5:	Prototype of Parents Home Screen.	69

Figure 4.6:	Prototype of Activity Summary Report Screen.	70
Figure 4.7:	Prototype of Set Time Limit Screen.	71
Figure 4.8:	Prototype of Content Filtering Screen.	72
Figure 4.9:	Prototype of Location Screen.	73
Figure 4.10:	Prototype of Location Marked Screen.	74
Figure 4.11:	Prototype of Notification Screen.	75
Figure 4.12:	Prototype of Notification Screen (cont.).	76
Figure 4.13:	Prototype of Notification of Blocked Access Screen.	77
Figure 4.14:	Prototype of Parents User Profile Screen.	78
Figure 4.15:	Prototype of Modify Parents User Profile Screen.	79
Figure 4.16:	Prototype of Children Login Screen.	80
Figure 4.17:	Prototype of Children Sign Up Screen.	81
Figure 4.18:	Prototype of Children Sign Up Screen (cont.).	82
Figure 4.19:	Prototype of Children Home Screen.	83
Figure 4.20:	Prototype of Send SOS Screen.	84
Figure 4.21:	Prototype of Stop SOS Screen.	85
Figure 4.22:	Prototype of Tasks of Rewards Screen.	86
Figure 4.23:	Prototype of Rewards of Rewards Screen.	87
Figure 4.24:	Prototype of Children Profile Screen.	88
Figure 4.25:	Prototype of Children Modify Profile Screen.	89
Figure 5.1:	Entity Relationship Diagram	91
Figure 5.2:	Children Class Diagram	93
Figure 5.3:	Parent Class Diagram	95
Figure 5.4:	Interface Flow Diagram of Access the Application.	97
Figure 5.5:	Interface Flow Diagram of Parents Application.	97

Figure 5.6:	Interface Flow Diagram of Children Application.	98
Figure 5.7:	Setup Flutter	99
Figure 5.8:	Setup Android Studio	100
Figure 5.9:	Setup Visual Studio	100
Figure 5.10:	Setup Firebase	101
Figure 5.11:	Setup Firebase in Visual Studio	101
Figure 5.12:	Setup NextDNS	102
Figure 5.13:	Setup Google Cloud	102
Figure 5.14:	Onboarding Screen	103
Figure 5.15:	Code of Onboarding Screen	104
Figure 5.16:	Parent Login Screen	105
Figure 5.17:	Code of Parent Login Screen	105
Figure 5.18:	Parent Sign Up Screen	107
Figure 5.19:	Code of Parent Sign Up Screen	107
Figure 5.20:	Forgot Password Screen	108
Figure 5.21:	Code of Forgot Password Screen	109
Figure 5.22:	Reset Password	109
Figure 5.23:	Parent User Profile Screen	110
Figure 5.24:	Parent Change Password Screen	110
Figure 5.25:	Parent Add Device Screen	111
Figure 5.26:	Parent Dashboard Screen	112
Figure 5.27:	Parent Set Time Limit Screen	113
Figure 5.28:	Parent Set Time Limit Screen	114
Figure 5.29:	Parent Content Filtering Screen	114
Figure 5.30:	Log of the Blocked Content	115

Figure 5.31: Parent Content Filtering Feature 115

Figure 5.32: Parent Location Tracking Screen 116

Figure 5.33: Code Get Child Location 117

Figure 5.34: Parent Mark a Place Feature 118

Figure 5.35: Parent Mark a Place Function 119

Figure 5.36: Parent View, Edit and Delete Marked Place Feature 120

Figure 5.37: Parent View Marked Place Function 120

Figure 5.38: Parent Delete Marked Place Function 121

Figure 5.39: Parent Notification Screen 122

Figure 5.40: Parent SOS Notification Feature 123

Figure 5.41: Parent Request Notification Feature 124

Figure 5.42: Parent Setting Screen 125

Figure 5.43: Parent About Us Feature 125

Figure 5.44: Parent Feedback Feature 126

Figure 5.45: Parent Logout Feature 126

Figure 5.46: Child Login Screen 127

Figure 5.47: Child Connect Device Screen 128

Figure 5.48: Code of Child Connect Device Screen 129

Figure 5.49: Child User Profile Screen 130

Figure 5.50: Child Dashboard Screen 131

Figure 5.51: Child SOS Screen (send) 132

Figure 5.52: Child SOS Screen (Stop) 132

Figure 5.53: Code of Child SOS Screen 133

Figure 5.54: Child Reward Screen of Tasks Tab 134

Figure 5.55: Child Reward Screen with Rewards Tab 135

Figure 5.56:	Code of Child Reward Screen	135
Figure 6.1:	Result of Usability Question 1	162
Figure 6.2:	Result of Usability Question 2	163
Figure 6.3:	Result of Usability Question 3	163
Figure 6.4:	Result of Usability Question 4	164
Figure 6.5:	Result of Usability Question 5	164
Figure 6.6:	Result of Usability Question 6	165
Figure 6.7:	Result of Functionality Question 1	165
Figure 6.8:	Result of Functionality Question 2	166
Figure 6.9:	Result of Functionality Question 3	166
Figure 6.10:	Result of Functionality Question 4	167
Figure 6.11:	Result of Functionality Question 5	168
Figure 6.12:	Result of Performance Question 1	168
Figure 6.13:	Result of Performance Question 2	169
Figure 6.14:	Result of Overall Satisfaction Question 1	169
Figure 6.15:	Result of Overall Satisfaction Question 2	170
Figure 6.16:	Result of Overall Satisfaction Question 3	170
Figure 6.17:	Result of Suggestion from Respondents	171

LIST OF SYMBOLS / ABBREVIATIONS

AI	Artificial Intelligence
App	Application
ERD	Entity Relationship Diagram
CSS	Cascading Style Sheets
GPS	Global Positioning System
HTML	HyperText Markup Language
IDE	Integrated development environment
MCO	Movement Control Order
ML	Machine Learning
PC	Personal computer
RAD	Rapid Application Development
SDLC	Software Development Life Cycle
SMS	Short Message Service
UAT	User acceptance testing
UI	User interface
VPN	Virtual private network
WBS	Work breakdown structure

LIST OF APPENDICES

Appendix A: Gantt Chart 178

Appendix B: Google Survey Form 181

Appendix C: Test Results 185

CHAPTER 1

INTRODUCTION

1.1 General Introduction

In the era of digital acceleration, the Internet is vital for people to get information, communicate and entertain. With the continuous development of technology, people will rely more on digital platforms, and the Internet has become the cornerstone of various activities, including education, work and social interaction. During the lockdown of Movement Control Order (MCO), the importance of online connection became more obvious, and the traditional communication and learning mode turned to the digital field. Online courses, telecommuting and virtual social interaction have become an important part of human daily life, which highlights the important role of the Internet in maintaining social contact.

However, the increasing dependence on the Internet during MCO has also brought some challenges, especially in the use of the Internet by children. Due to the lockdown, schools are running the education systems on online platforms, and these facilities are sometimes misused or abused by the children because they don't know how to use the Internet correctly, which leads people to worry about the safety and well-being of children on the internet. Incidents such as inappropriate content exposure, cyber bullying and excessive screen time have become urgent problems, it makes people need comprehensive solutions to balance the benefits and potential risks related to online activities.

A recent study by Pew Research Center in 2018 shows that 65% of the 1,000 parents surveyed are worried that their children spend too much time online. Some surveyed parents (58%) supervise their children’s use of science and technology, others use parental controls to restrict website access (52%), some parents limit the time and duration of children surfing the Internet or using smart phones (58%) (Monica, 2018).

The remote parental control project for children's access to Internet usage or content is a timely response to these challenges. The project aims to provide parents with the necessary tools to ensure that their children have a safe and reliable online experience. By providing remote control function, parents can actively monitor and manage their children's Internet activities, protect them from potentially harmful content, and reduce the risks associated with unlimited online access.

In conclusion, the Internet can also be called a double-edged sword, which brings convenience as well as challenges. A little carelessness will lead to the risk of leakage. Therefore, the transition to digital platforms during MCO not only emphasizes the importance of the Internet, but also emphasizes the responsibility of parents and the necessity of guiding children to use the Internet correctly. This project then provides parents with a comprehensive solution to adapt to the developing trends in the digital field. Through customizable control, real-time monitoring and content filtering, the remote parental control system has become an overall method to create a safe and rich online environment for children, ensuring the benefits of using digital connection responsibly and safely.

1.2 Importance of the Study

The importance of this project is its ability to tackle urgent societal problems and benefit society in total. The project goal is to address important social issues related to online safety and digital literacy by using remote parental control systems to manage children's access to the internet and content. In the modern era of technology, children having limitless internet access can make them vulnerable to risks like seeing inappropriate content, cyberbullying, and spending too much time staring at screens. These dangers not only endanger children's health, but also add to larger societal issues.

An important focus of this research is its ability to steer children away from harmful online paths and towards safer, more productive digital engagements. By enabling parents to remotely monitor and control their children's online behaviour, it aims to create a protective cover from harmful influences and guide children to get positive online content and experience. By

doing this, this project helps support the overall goal of encouraging responsible digital citizenship and fostering positive online habits in children.

Furthermore, this system could help reduce family disputes and conflicts regarding screen time and online activities by empowering parents to better control their children's internet usage and content access. By implementing remote parental control systems, parents can set definite boundaries and time restrictions for their children's internet activities, leading to fewer arguments and miscommunications within the family unit. This does not just help cultivate better parent-child bonds but also encourages a more peaceful family setting that supports overall health.

In conclusion, the importance of this research goes beyond just personal households to include wider social impacts. Remote parental control systems have the potential to have a significant impact on both families and society as a whole by tackling social issues related to online safety, steering children towards positive digital experiences, and minimizing family conflicts surrounding internet usage.

1.3 Problem Statement

In today's rapidly developing technology environment, parents are increasingly worried about their children online safety as children engage in various activities on the Internet due to children (Up, 2023). Although Internet access is mainly introduced to meet school-related needs, such as homework (43%), children often watch entertainment videos (40%) and play games (29%) more frequently as children are more prefer entertainment videos and games over school-related activities. It must be pointed out that the age of children's contact with the Internet is generally between 5 and 8 years old. The gap between what parents expect and how their children actually use devices highlights the need for strong content filtering in remote parental control apps. Parents are very worried that the Internet is not a safe space, and most of them are worried about their children expose to inappropriate content such as gambling, drug and alcohol, and pornography. Parents give priority to protecting their children from such content, when children access to internet. Therefore, the content filtering

function in the remote parental control application becomes very important to ensure that children's online experience meets parents' expectations and security concerns, reassure parents, and allow children to explore the digital world safety.

According to Inc (2023), mobile devices are becoming more and more common among children, which causes parents to worry about their safety and health, especially in emergencies. Many families are become more concerned about their children's safety when they are not under parents' direct supervision. Parents are concerned about various possible threats, including accidents, losing their way, encountering unfamiliar individuals, or dealing with emergencies like injuries or sudden sickness. With the enhancement of independence, for example, when children go to tuition on foot or by bus, visit friends or participate in extracurricular activities independently, the need for reliable communication and help becomes very important. In this case, having a parental control application with SOS feature is important for children can quickly seek help from their parents when needed, thus greatly enhancing security measures.

In addition, Emerson et al. (2019) had mentioned that parental monitoring consistently predicts children's anxiety, which will also lead to communication problems in the family. Parents who are overly controlling their children may cause their children to feel like lack in autonomy and freedom. This feeling of limited freedom will worsen children's anxiety, which may make it difficult for them to express themselves and communicate openly in the family unit. This challenge will hinder the healthy parent-child relationship and the development of trust and mutual understanding. Therefore, the feature for children to claim the rewards of extra screen time by doing some tasks is necessary to develop in the application in order to allow children to exchange extra time for themselves.

In conclusion, the problems found in existing remote parental control systems underscore some features in need of enhancement. Integrating an effective SOS feature into the system essential to guarantee the safety of children when outdoors. Improving website filtering mechanisms is crucial for ensuring strong protection against inappropriate online content. It is important

to also focus on parental control problems that add to children's anxiety and difficulties in communicating within families to promote positive parent-child connections. These aspects will receive special attention in the upcoming project in order to improve the overall safety, security, and well-being of children in the current digital era.

1.4 Aim and Objectives

1) To review similar parental control applications and define the requirements specification.
2) To develop a mobile application with tracking and content filtering features for parents to monitor their children's internet activities.
3) To evaluate the developed mobile application using User Acceptance Test (UAT).

1.5 Proposed Solution

The proposed solution for this project is to develop a mobile application for Android devices. This mobile application is for the parents to oversee and monitor their children's activities, while the children can use it to make requests to their parents. The mobile application will be created using the Flutter framework with Dart programming language for the front-end and Firebase for the back-end. This extensive parental control application seeks to provide efficient ways for families to protect their children's safety and health in the digital age. Parents can easily monitor and manage their children's online activities, such as screen time, application usage, and website access, location with a user-friendly interface. In order to ensure the data integrity and reliability, the application makes use of Firebase's powerful capabilities such as secure authentication, real-time database processing, and cloud storage. Using Flutter's flexibility and efficacy, the application provides a seamless experience across several platforms, allowing parents to control their children's devices from anywhere. This proposed approach allows families to successfully manage the complex digital world, develop healthy digital behaviours, and provide a safer online space for children. Figure 1.1 shown the system architecture of the proposed solution.

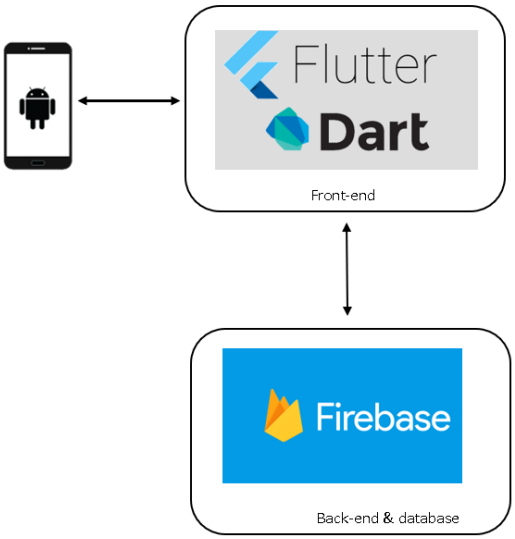


Figure 1.1: The System Architecture of The Proposed Project.

1.6 Project Approach

1.6.1 Development Approach

The project used the phased development methodology as part of the Rapid Application Development (RAD) methodologies. During phased development, the project was separated into different phases, with each phase concentrating on particular deliverables or functionalities. This method allows for step-by-step development by adding features gradually instead of all at once. By dividing the development process into several stages, it is easier to plan, execute and monitor progress effectively. In conclusion, it ensures that the final product is satisfying for the users which meets the needs and expectations of users. For more information including in Chapter 3.2 Rapid Application Development.

1.7 Scope and Limitation of the Study

1.7.1 System Scope

The scope of the project includes the development of a remote parental control system, which enables parents to remotely monitor, manage and control their children's Internet use and content access. This includes developing a user-friendly interface for seamless navigation to ensure that parents can easily access and track their children's online activities from any location. The main functions include content filtering to detect and inappropriate content, real-time notification of suspicious activities, screen time management, activities

reporting, SOS feature and rewards feature. Priority will be given to security measures to prevent unauthorized access, and parents will be provided with effective and safe tools to actively monitor their children's digital interaction.

1.7.2 Target User

The project's target users are parents and children who under the age of 13, with a focus on the parental group. While both parents and children will use the application, parents are able to use the application to remote monitor and manage their children's online activities effectively. Parents can set up the parental controls through the application, allowing them to monitor screen time, and block access to inappropriate content of the websites and track the real-time location of their children, thereby ensuring a safe and secure digital environment for their children. However, children will also interact with the application while exploring the online world under parental guidance. Children can do some tasks for exchanging the extra screen time. This project aims to encourage healthy digital habits and responsible internet usage among families by meeting the needs of both parents and children.

1.7.3 Application Features

1.7.3.1 Login and Sign up for users

This feature allows the parents and children to sign up or login to the application. It helps users verify their identity and set up an account within the application.

1.7.3.2 User Profile

This feature allows parents and children to view and manage their personal information in the application.

1.7.3.3 Screen time management

This feature allows parents to control and monitor the time children spend on devices. Parents able to set daily or weekly screen time limits for their children.

1.7.3.4 Activity Summary

This feature provides parents with a dashboard that summarizes their children's online activities.

1.7.3.5 Content filtering

This feature will filter the content websites by some categories like pornography, gambling and other. It will also block the specific websites.

1.7.3.6 Location Tracking

This feature allows parents to monitor the child's real-time whereabouts, to ensure the safety of children. It also supports geofencing that allow parents to get real-time alert when children arrive or leave some place in a meter of range.

1.7.3.7 SOS feature

This feature allows children to send SOS to parents when they are in trouble. Parents will receive a real-time alert with the location of children.

1.7.3.8 Rewards

This feature allows children to claim and request extra screen time by completing tasks such as housework, exercise or reading. Parents can approve or reject based on the children’s behaviours.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In the today’s digital era, the Internet has a wide influence on all aspects of people’s lives, especially on children's online behaviour. Therefore, it is important to understand the implications of children's internet usage. The literature surrounding this topic delves into the multifaceted nature of children's online experiences, exploring both the opportunities and challenges presented by their interactions with digital platforms.

First and foremost, the research is taken from the history of the remote parental control application. The remote parental control applications such as Cyber Patrol and Net Nanny were the initial version which developed in the late 1990s. These applications mainly focus on the content filtering on the Internet and block access to some websites that are not suitable for children such as content with gambling, hatred or violence and pornography (Ivor.Pro, 2023). They will check the content of every website the children visit and block it. These applications set the stage for the future development of remote parental control technology.

As time goes on, the remote parental control applications have made some enhancements in features and functions to adapt to the changing digital environment. After that, the time management features such as set limit screen time, and the time use on the Internet or application are developed to allow the parents to monitor their children on Internet. These time management functions are constantly developing and give the parents more flexibility and choices to customize screen time limits and internet or applications usage schedules to fit their family's unique needs and daily routines.

Grossinger (2013) had mentioned that the parental control systems were extremely popular in the mid-2000s. This is because of the appearance of smartphones and tablets in the late 2000s and early 2010s, parental control

systems available on cross-platform devices, and allowed parents to manage their children remotely. The Applications such as Qustodio and Norton Family give parents the option to monitor and manage their children's internet activities using any device connected to the Internet. As technology keeps improving, the new features like geofencing and social media tracking are helping parents to ensure their children's online safety (Global Newswire, 2024). Geofencing allows users to mark a location with its latitude and longitude.

Nowadays, there are many different remote parental control applications available in market, and each offering unique features and competing with one another. Based on the market research, the global Parental Control market is projected to grow by approximately 10% annually from 2021 to 2026 (metafisica, 2024). This factor is due to the fact that children spend more time using the internet, which will have some negative effects on their health. However, the remote parental control applications nowadays have solved certain problem, but there are still exist some challenges in terms of usability, efficacy, and privacy. Therefore, in this chapter, a literature review of the existing similar system was conducted.

2.2 Review Existing Similar System

2.2.1 Qustodio

Figure 2.1: Logo of Qustodio.

Qustodio is one of the popular remote parental systems in market which designs for ensuring every child has a secure digital experience. It aims on protecting the children form inappropriate content and cyberbullying and enable the parents to foster a healthy environment. It has used by more than 6 million users

in 8 languages and 180 countries. Qustodio was founded in 2010 by a team of cybersecurity experts who are Eduardo Cruz, Josep Gaspar, and Josh Gabel. Its logo is designed as shown in the Figure 2.1. Their vision is to give the parents power to protect their children online (Qustodio, n.d.). It continues to develop and improve in adding new features and expanded to cover various devices and operating systems.

Qustodio is a paid system, but it provides parents with a 30-day free trial and can be used on all platforms, including Windows, Android, iOS, MacOS, Chromebook and Kindle. It provides some features that help the parents to monitor and manage their children. These features including websites and application filtering feature that only works on supported browser such as Google Chrome, Microsoft Edge, and Safari. This feature lets the parents to select the websites that children are allowed to visit. Parents can set filtering rules for specific websites from 25 categories such as games, entertainment, education, gambling, pornography and more. Also, parents can block these websites, and the unknown websites. Therefore, when the children access the blocked websites or accidentally access unknown websites, parents will receive a real-time alert on the device. The interface of filter content and application shown in the Figure 2.2.

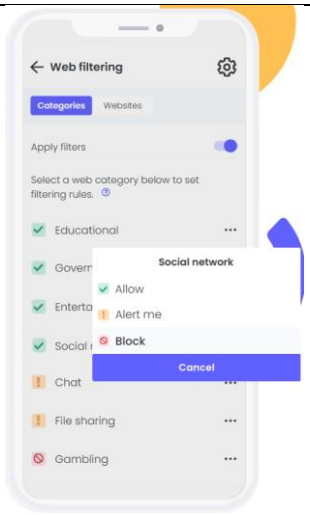
Besides, Qustodio also has a feature that allow parents to set a daily screen time for their children to use the device as show in Figure 2.3. The time set from parents will display in the children's application which will remind them how much the time left. Parents can choose to lock children's device or lock their navigation which means they can only access the basic application in the device such as calculator and calendar. Children's device will be locked automatically when the times up or manually from parents.

Moreover, Qustodio provide a dashboard for parents to view the daily, weekly, bi-weekly and monthly activity report. In the dashboard, parents can get more information on children's online activities. For example, each application that accessed or websites that visited will display on the dashboard. The interface of the feature can view in the Figure 2.4. Furthermore, Qustodio

able to track the real-time location of children and let parents know where their children are. It shows the timeline of the locations where children have been before to ensure the safety of their children as the interface show in the Figure 2.6. Parents are also encouraged to add some places as “favour” like home, school and tuition centre so that they will receive the notification when their children arrive at and leave the saved place. This is the so-called geofencing.

In addition, the interface of call and SMS tracking feature shown in Figure 2.5, which helped parents to monitor children’s call and messages. Parents can check the records of their children’s call and text messages as well as the call time. This feature able to detect the suspicious contacts and set allow and block call for specific contact. For instance, parents can set themselves as allow calls and unknown contacts as block calls. Plus, a panic button feature designed from Qustodio is an important feature for the children to send SOS to their parents when they are in trouble as shown in the Figure 2.7. When the children press the button for SOS, an alert will send to parents’ device immediately with the real-time location of their children. All the interface of the features are shown in the Table 2.1.

Table 2.1: Feature and Interface of Qustodio

Feature	Interface
Filter Content and Apps • Filter and block inappropriate applications, games, and websites such gambling content, violence and other potentially harmful threats. • Parents will receive alerts when their children access the blocked content.	 Figure 2.2: Feature of Filter Content and Apps for Qustodio.

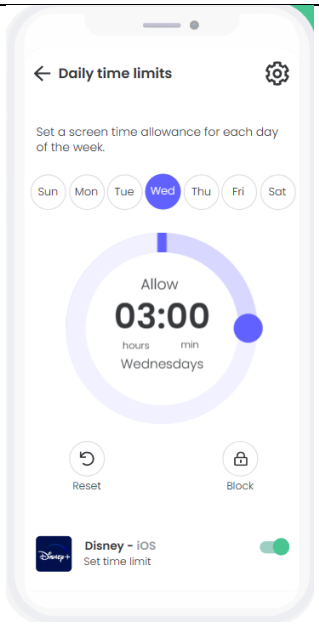
<table>
<tr>
<td>Set Time Limits
• Parents able set the daily screen time limit.
• Children’s device will be locked when times.</td>
<td> 

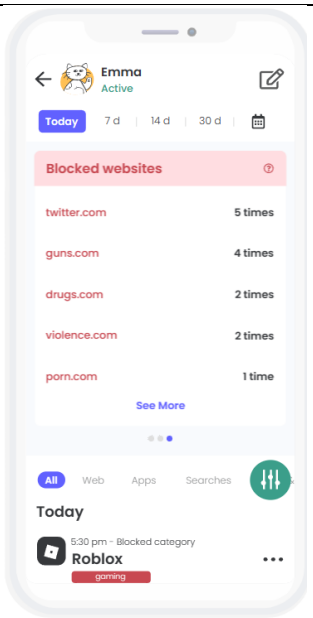
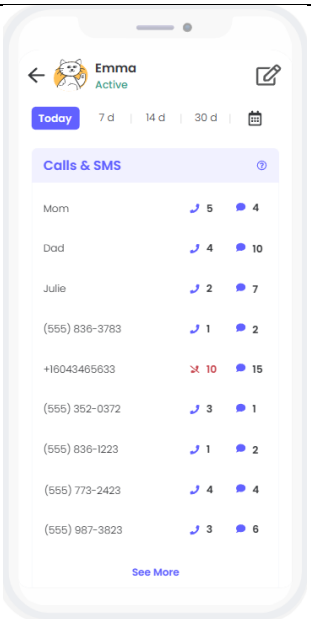
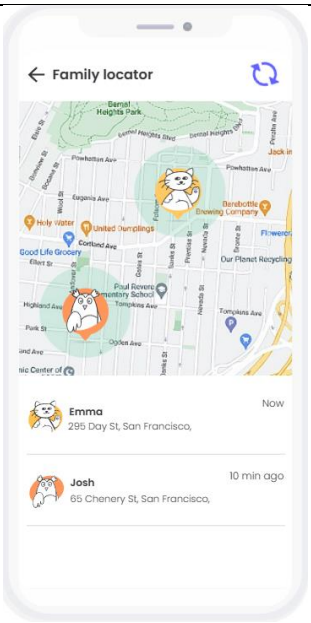
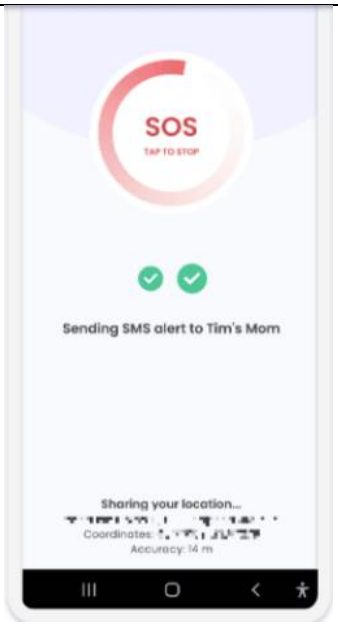
Figure 2.3: Feature of Set Time Limits for Qustodio.</td>
</tr>
<tr>
<td>Monitor Activity
• Provide the real-time dashboard.
• Parents can view their children’s browsing history, social media used time, YouTube views, screen time and the activity timeline.</td>
<td> 

Figure 2.4: Feature of Monitor Activity for Qustodio.</td>
</tr>
</table>

Track Call and SMS for Android and iOS • Detect suspicious contacts. • Parents able to see who is exchanging calls and messages with children and can view their chat history. • Block call to and from specific contact.	 Figure 2.5: Feature of Track Call and SMS for Qustodio.
Locate Family • Check real-time locations of children on the map. • Parents will receive alerts when children arrive and leave the locations that parents saved.	 Figure 2.6: Feature of Locate Family for Qustodio.

Alert and SOS • Press the button for sending SOS. • Parents will receive alert of SOS with the location.	 Figure 2.7: Feature of Alert and SOS for Qustodio.

2.2.2 Mobicip

Figure 2.8: Logo of Mobicip.

Figure 2.8 shows the logo of Mobicip. Mobicip provide a user-friendly interface and powerful functions to enable parents to monitor and control their children by filter the content accessed, set screen time and other functions on various devices and platforms. It is also a paid system which is available on iOS, Android, macOS, Windows, Chromebook and Kindle. Mobicip's mission is to enable parents to protect their children on the Internet. It offers a wide range of tools designed to minimize the risk associated with the Internet use and encourage children to develop positive digital habit. It is trusted by millions of

parents around the world and promise to offer modern user-friendly solutions that are efficient and easy to use.

Mobicip was founded in 2018 by Suren Ramasubbu with a primary goal of solving the increasing worries among the parents regarding children's Internet safety. By realize the importance of being proactive in safeguarding children in the digital era, Ramasubbu set out to develop a solution for parents to efficiently monitor and manage their children's online activities. The central focus of Mobicip's approach is to continuously enhance and adjust to the constantly evolving digital trends and challenges. The company stays alert in addressing new threats and issues to make sure their solutions stay efficient and appropriate in an evolving environment.

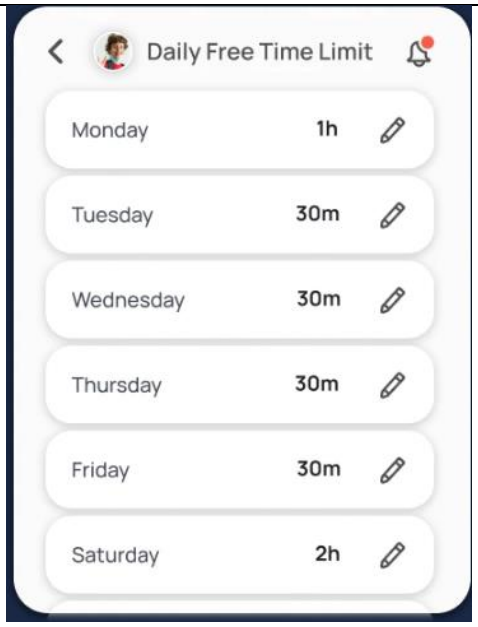
Table 2.2 shown all the interface of Mobicip, which gives parents with many features for monitoring their children, one of which is time management. Parents are ability to limit their children's screen time daily on the interface on Figure 2.9. Once the time set is used up, the device of children will be blocked. The scheduling feature allows parents to create an unlimited number of custom applications and web filters for specific days and times in increments of 5 minutes. For parents have a summary of children's online activities, a dashboard is designed as Figure 2.10 for them to view the usage and time spend and online browsing history from their children.

Besides, Mobicip's application blocking feature enables parent to limit your children's access to certain category of the applications, such as social media, entertainment and games. Parents can also search for and block it from the app store which is the unique feature among the remote parental control applications. Similarly, Mobicip uses advanced AI and ML technologies to filter the websites in real-time in order to guarantee that only suitable content is accessible to children (Mobicip, n.d.). Parents can also block more than 20 categories of specific websites which including inappropriate words, phrases and metadata to determine whether are suitable for children. Mobicip is unique in that it can filter YouTube videos separately on iOS and Android devices, and

also manage YouTube access on PCs, Macs and Chromebooks. Figure 2.12 and Figure 2.13 shown the interface of the app blocker and website blocker feature.

Furthermore, Mobicip offers features including social media monitoring of Instagram, Facebook, and Snapchat as well as marking bullying or violence, which show the interface in Figure 2.11. It checks the content of the social media for related to bullying, pornography, drugs, violence and self-harm and Parents will get a reminder with content snippets to review immediately. Moreover, Mobicip able to track the children's real-time location at the interface on Figure 2.14 using GPS technology. Google Maps accurately tracks their location, with a 30-day history, so parents can share their location with children (WU, 2024). The geofence alarm will alert parent when children enter or leave the defined area, covering a distance of around 1km, offering thorough monitoring and safety control over their digital and physical activities. The table below (Table 2.2) shows the features and interfaces of Mobicip.

Table 2.2: Feature and Interface of Mobicip

Feature	Interface
Screen Time • Allow to set schedules for activities. • Set daily screen time.	 Figure 2.9: Feature of Screen Time for Mobicip.

Activity Summary

- Track the usage and time spend.
- View online browsing history.

Figure 2.10: Feature of Activity Summary for Mobicip.

Social media monitor

- Scan the social media for harmful content.
- Receive real-time notifications for any dangerous or improper actions detected.

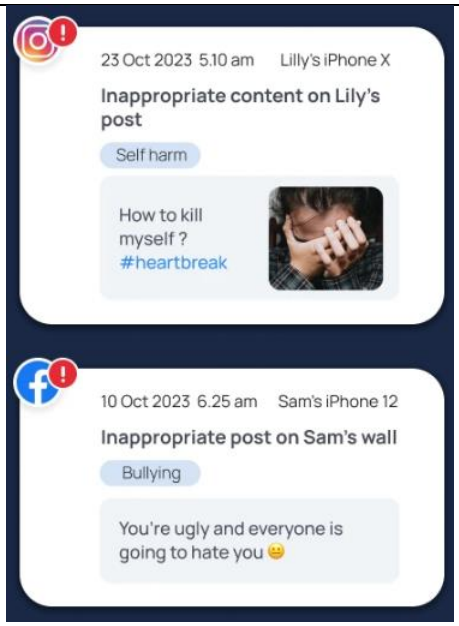


Figure 2.11: Feature of Social Media Monitor for Mobicip.

App Blocker

- Limit the time spent on application.
- Block certain application.

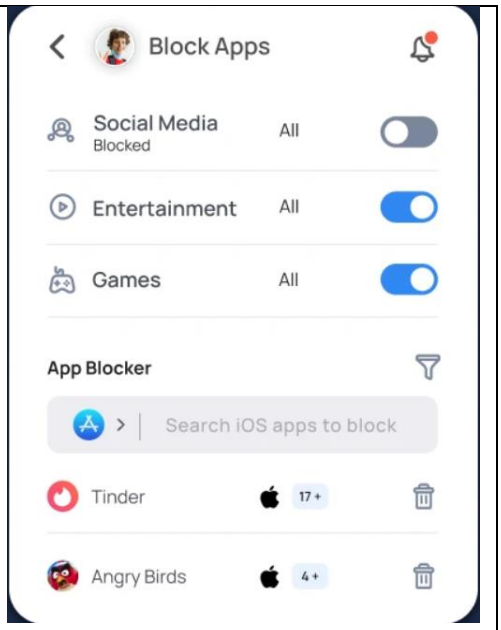


Figure 2.12: Feature of App Blocker for Mobicip.

Website Blocker

- Filter the content of the websites that visited.
- Block the inappropriate websites.
- Set some websites to allow access.

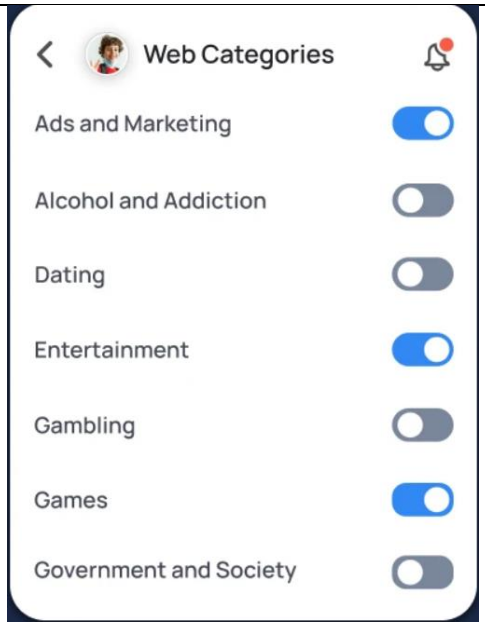
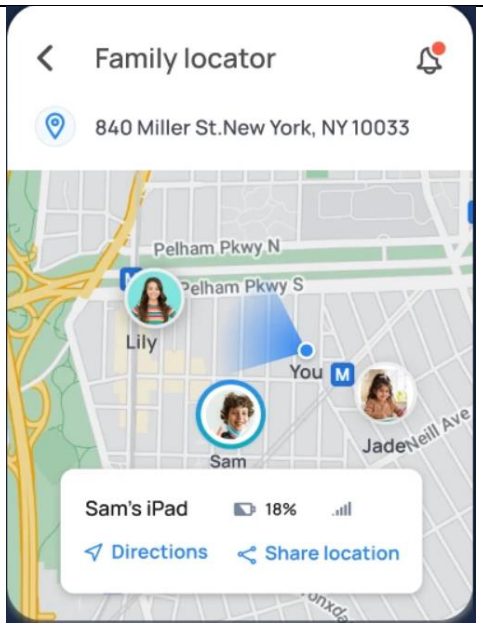


Figure 2.13: Feature of Website Blocker for Mobicip.

Family Locator • Track and share the real-time location. • Receive alert when children reached or leave a saved place.	 Figure 2.14: Feature of Family Locator for Mobicip.

2.2.3 Google Family Link

Figure 2.15: Logo of Google Family Link.

Google Family Link is a convenient service offered by Google which allows parents to easily monitor their children's devices through parental controls. The Family Link's logo is designed as shown in Figure 2.15. This service is launched in March 2017 and now available in 38 countries including Malaysia, Japan, and Australia. It provides the parental control applications for many platforms such as iOS, Android, and web browsers as well as applications for children on Android and Chromebook. It is a free application for parents, and they need to login using their own Google Account.

Google Family Link is popular because of its user-friendly interface and comprehensive functions. The ongoing enhancement and expansions show their effort on solving the current issues of parents. The primary mission is to provide parents with a tool to help them to create a healthy and positive digital habit for their children (families.google.com, n.d.). It provides the feature including screen time management, content filtering, and location tracking for parents concerned about their children's online activities and seeking to track application usage.

Google Family Link designed in a user-friendly interface with only three tabs which are Highlights, Controls, and Location. In Highlights, parents can have an overview on the time spent by their children on the device and the applications they use. The interface can view in the Figure 2.16. Google has partnered with Common Sense Media, ConnectSafely, and the Family Online Safety Institute to offer support to parents in monitoring and managing their children’s internet usage (Shakir, 2022).

Moreover, the most features are in the Control tab, including screen time management, activity reports, app blocker and website filtering. Parents are allowed to set daily use limits and bedtime limits shown in Figure 2.17. These limits can be customized for each day of the week and downtime settings can prevent children from using device for a specified time. Parents are also allowed to view their children's daily screen time and the usage time of specific applications. Besides, Google Family Link enables parents to manage applications access and block specific applications. It manages children's access to browsing, buying, or downloading on Google Play which other parental control does not have it. It has prevented children from installing or purchasing applications form Google Play without permission from parents. The interface of application blocker is shown in Figure 2.19.

Furthermore, website filtering feature is also included in Control tab as shown in Figure 2.18. Google Family Link contains the function such as SafeSearch which is used to block explicit content on Chrome search results, customizable filters for explicit content on websites, and options for allowing

access to approved websites. As the children cannot delete the search history on Chrome, it prevents the use of Incognito Mode and notifies parents of any changes of setting made by their children.

In Location tab which shown in Figure 2.20, Google Family Link offers simple location supervision feature, enabling parents to view their children's real-time location on Google Maps. Parents will receive an alert when children arrive or leave a specific place such as home or school, but it cannot set up a zone range that called geofencing function (WU, 2024).

Table 2.3: Feature and Interface of Google Family Link.

Feature	Interface
Activity Report • Activity report that shows the daily screen time and time usage of applications.	 Figure 2.16: Feature of Activity Report for Family Link.

Set Screen Time Limits

- Set time limit for the use of the device and bedtime.
- Device will lock when times up.

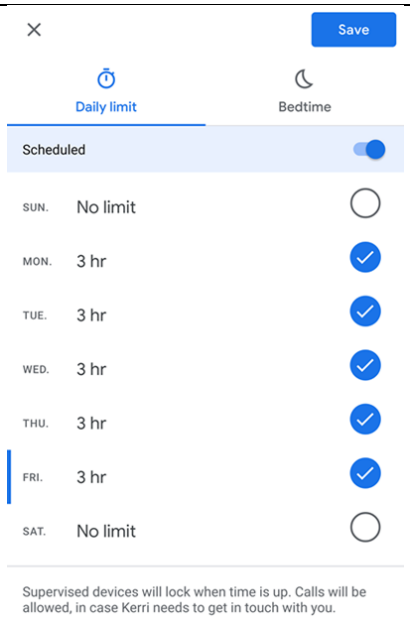


Figure 2.17: Feature of Set Screen Time Limits for Family Link.

Website Filtering

- Filter and block the explicit content on websites.
- Filter and block the explicit content on search engine.

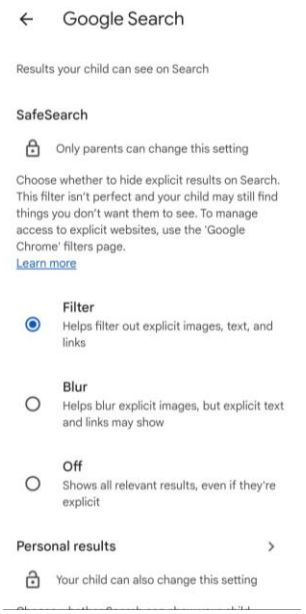


Figure 2.18: Feature of Website Filtering For Family Link.

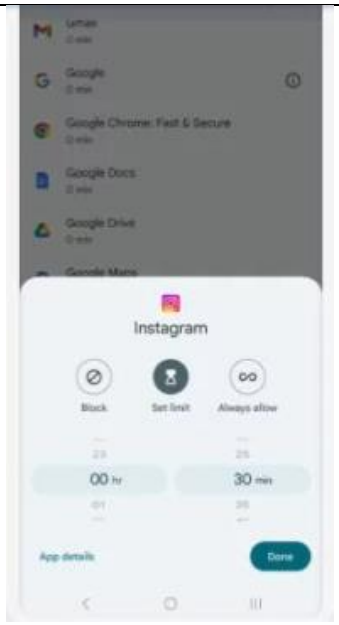
<table>
<tr>
<td>App Blocker
Block a specific application.
Set time limit for a specific application.</td>
<td> 

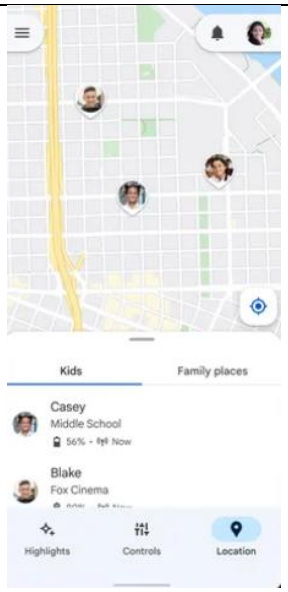
Figure 2.19: Feature of App Blocker for Family Link.</td>
</tr>
<tr>
<td>Location Tracking
Track the real-time location.
Receive alert when children reached or leave a specific location.</td>
<td> 

Figure 2.20: Feature of Location Tracking for Family Link.</td>
</tr>
</table>

2.2.4 Bark

Figure 2.21: Logo of Bark.

Bark, a tech company founded in 2015 by Andrew Michael and Brian Bason, located in Atlanta, Georgia, specializes in software that monitors children's internet and social media activities. Bark Parental Control System is one of the services provided by the Bark company that need to pay for use on device like smartphones, tablets, computer which is available on iOS, Android, macOS, Windows, Chromebook and Kindle. It focuses on helping parents to track their children's digital communications on different platforms, including email, SMS, and social media. Figure 2.21 shown the logo of Bark.

The primary purpose is to alert parents about issues in time. For instance, when children suffer from cyberbullying or signs of suicidal thoughts, parents can step in and provide help in time. One of the key advantages of Bark is its Artificial Intelligence (AI) technology used for detecting the harmful content and potential threats (Bark, n.d.). It not only reminds parents about the potential problems, but also gives them advice on how to handle these situations of children.

Figure 2.22 shown the interface of website and application blocker feature. This feature helps parents manage and control their children's internet usage effectively. By using a virtual private network (VPN), it has the ability to prevent children from accessing certain websites and mobile apps such as social media, streaming services, and games while also enforce safe search for search engine and YouTube. It will filter the websites based on the categories such as games, pornography, drug and other inappropriate content and parents can select

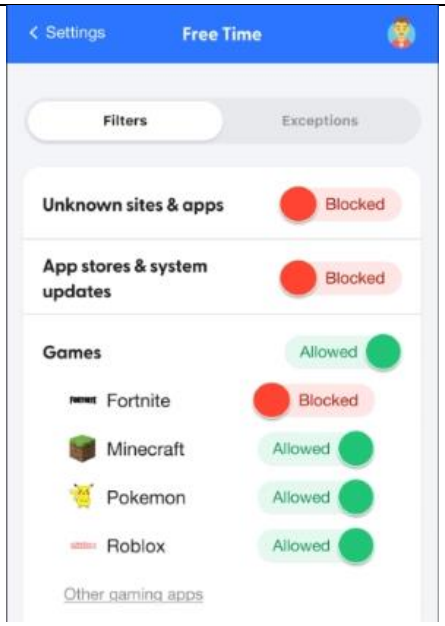
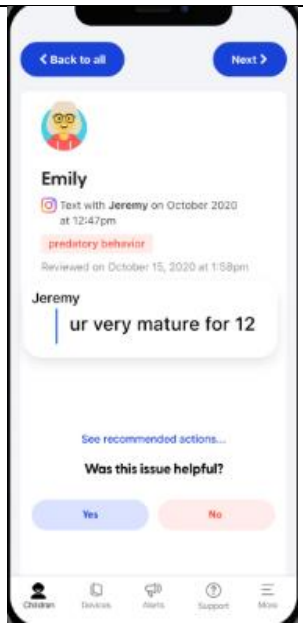
to allow access or block the specific websites. When children try to access the inappropriate content, blocked websites or blocked applications, an alert will be sent to parents.

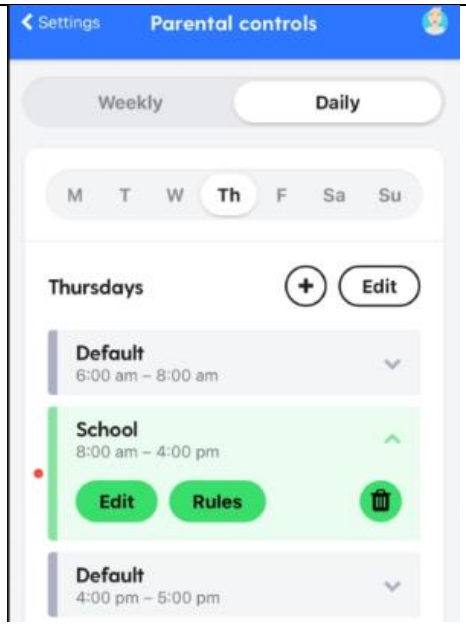
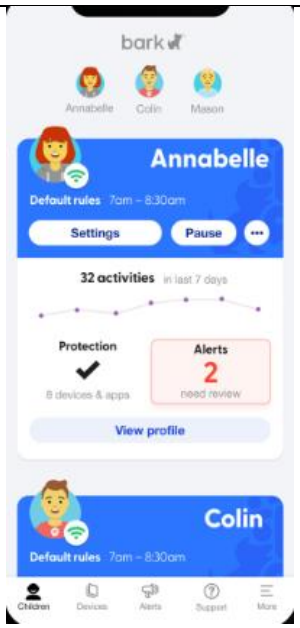
Besides, Bark provides content monitoring for text messages, images, audios, videos and over 30 social media platforms and applications such as WhatsApp, WeChat, TikTok and Instagram. It uses advanced algorithms to identify possible problems such as bullying, depression, drugs, and violence in different categories as shown in Figure 2.23. When detected the potential problem, parents will receive a timely alert as well as display the text messages or content.

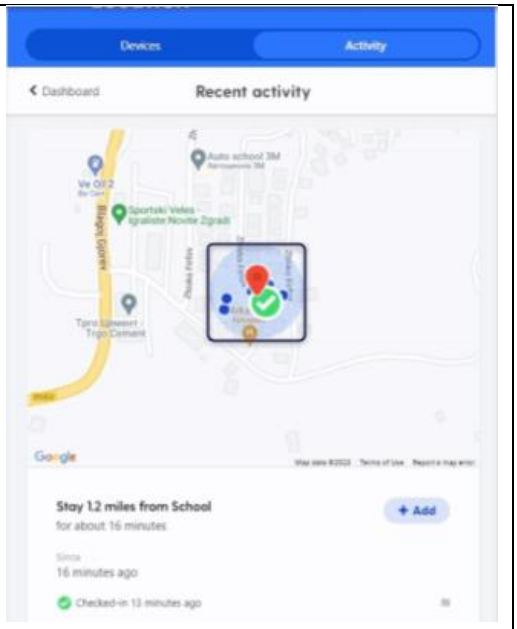
In addition, parents can create a schedule for each day and easily switch between applications and website access in the interface shown in Figure 2.24. Bark's screen time management feature allows parents to apply different website and application filtering rules according to the time of day such as free time, school time and bedtime. Moreover, parents can have an overview of children's online activities through the activity report feature as the interface shown in the Figure 2.25. It shows which apps your child uses most and what activities were blocked or allowed.

Furthermore, Figure 2.26 shown the interface of the location tracking feature, which enables parents to keep track of their children's current whereabouts with the live Global Positioning System (GPS). Parents can set up geofencing to get alerts when children reach or leave a specific place with the radius of up to 200 meters such as home or school. The Family Map feature allows parents to view all monitored devices in a single location and gives parents direction to children's location. Bark also stores a record of where children have been within a week, displaying patterns and duration in certain places (HRISTOVSKA, 2024).

Table 2.4: Feature and Interface of Bark

Feature	Interface
Website and App Filtering • Block inappropriate apps, games, and websites. • Allow children to visit children-friendly websites and automatically prevent them from viewing potentially harmful ones	 Figure 2.22: Feature of Website and App Filtering for Bark.
Content Monitoring • Detect the text or content exist any potential issue from social media.	 Figure 2.23: Feature of Content Monitoring for Bark.

Screen Time Management • Set time to block or allow access to application and websites. • Create a schedule for each day.	 Figure 2.24: Feature of Screen Time Management for Bark.
Activity Report • Provide dashboard to have an overview of children's online activities.	 Figure 2.25: Feature of Activity Report for Bark.

Locate Tracking • Check children's locations. • Receive alerts when they arrive and leave specific locations	 Figure 2.26: Feature of Location Tracking for Bark.

2.3 Comparison

Table 2.5: Comparison each system with advantages and disadvantages.

Existing system	Advantages	Disadvantages
Qustodio	- Support geofencing and real-time location tracking. - Provide SOS feature. - Useful web filtering.	- No social media monitoring. - Children cannot request for extra time
Mobicip	- Useful web filtering. - Support geofencing and real-time location tracking. - Filter and block inappropriate application on any app store	- Cannot monitor calls and SMS. - No provide SOS feature.

Google Family Link	- Free version. - User-friendly interface. - Filter on Google Play for needed approval before installation.	- No support geofencing and location history. - Lack of filtering options with content categories. - No provide SOS feature - Cannot monitor calls and SMS.
Bark	- Useful web and content filtering. - Excellent social media monitoring - Support geofencing - Protect children's privacy that will not show the chat history.	- Not allow to view the browser and call history. - No real-time location tracking. - No provide SOS feature

Table 2.6: Comparison between existing system and proposed project.

	Qustodio	Mobicip	Family Link	Boomerang	Proposed Project
Websites filtering	Yes	Yes	Yes	Yes	Yes
Tracking of calls and SMS	Yes	No	No	Yes	No
Social media Monitoring	No	Yes	No	Yes	Yes
Screen Time Management	Yes	Yes	Yes	Yes	Yes
Activity Summary	Yes	Yes	Yes	Yes	Yes
Real-time Location Tracking	Yes	Yes	Yes	No	Yes
Geofencing	Yes	Yes	No	Yes	Yes

SOS feature	Yes	No	No	No	Yes
Rewards	No	No	No	No	Yes

2.4 Summary

There are a lot of remote parental control system available in market, each offering unique features and functionalities with their own strength and weaknesses. After thorough research and analysis, it is obvious that the large number of remote parental control systems reflects the growing demand for tools to manage children's online activities. Each system has unique characteristics and functions to meet the needs and preferences of different parents. Through in-depth study of these systems, researchers aim to deeply understand their effectiveness, usability and influence on children's digital behaviour, and finally guide parents to make wise decisions to ensure children's safety and well-being in the digital age.

The project proposed introduces a unique feature that is not available in other system which is rewards feature. This feature allows children to gain extra screen time by completing tasks such as housework, exercise or reading. This has encouraged children to participate in beneficial tasks that enhance their overall well-being, while also promoting a healthier balance between screen time and other activities through the use of rewards feature. This new strategy seeks to not just regulate screen time, but also promote good habits and responsible online activities in children.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

In this chapter, the selected software development methodology will be explored, providing a detailed analysis of its structure and discussing each stage of the process. Furthermore, tools used during the project's development will be examine, with offering a deeper understanding of their roles and functions. Moreover, this chapter will cover the Gantt Chart and a work breakdown structure, which will present a visual depiction of project timelines and task allocation.

3.2 Rapid Application Development (RAD)

This project used a phased development methodology which breaks the overall system into a sequential version during development. Each stage, including planning, analysis and design, implementation and testing, and closing, starts only after the previous stage is completed. Then categorize the requirements into a series of versions, such as system version 1, system version 2 and so on. This method prioritizes features based on their importance, allowing iterative development and testing until the system is fully developed and ready for implementation. Figure 3.1 displays the phased development methodology’s overview in this project.

Phased software development methodology is chosen for developing remote parental control mobile application is because it allows in focusing on basic features first before moving to more complex features. For example, in system version 1 the simple feature such as time management features, activities summary and rewards features, then proceed to system version 2 which can implement and test for websites filtering, and last version for location tracking and SOS feature. By breaking down the development into few system version, it helps to keep a clear focus, iterate based on feedback, and adapt to evolving user needs and technological advancements effectively. Figure 3.1 shown the diagram of the Phase Development Methodology.

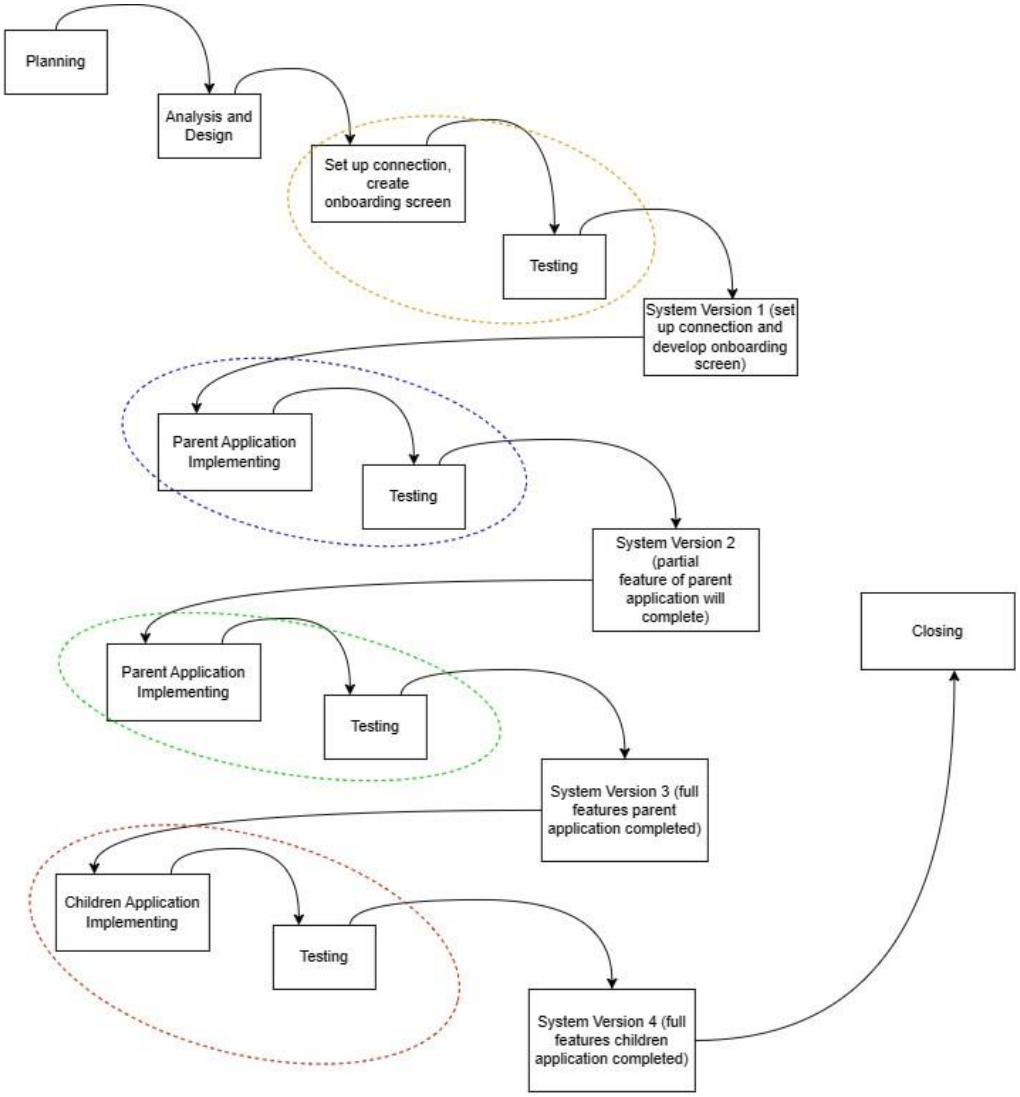


Figure 3.1: Diagram of Phase Development Methodology

3.2.1 Planning

Every SDLC development process starts from the planning stage, and phase development methodology is no exception. During planning phase, the project begins with a draft of project proposal that outlines the goals, scope, objective, and problem statement. Following this, thorough requirement gathering and elicitation activities are conducted to collect detailed information about the project's functional and non-functional requirements from research on existing system. Once these requirements are understood, the project scheduling phase involves creating timelines, allocating resources, and setting milestones to guide the development process effectively.

3.2.1.1 Project Proposal

The project proposal is the first task taken in during the development of the project. The project proposal for this initiative begins with a thorough analysis of the problems faced by the target users, parents and their children under 13 years of age, in Internet, and find the goal to solve these problems. The importance of studying this project is also included to gain a deeper understanding of the specific needs and problems faced by parents in effectively monitoring and managing their children's online activities. Clear project objective will help to define what the project aims to achieve upon completed. The important of study this project is also included to …Then, the appropriate system architecture is researched and defined. In order to ensure the efficiency and effectiveness of the development process, the correct system development methodology is identified. Finally, define the scope of the project to ensure that the project provides specific solutions to the identified problems within the specified time frame, so as to prepare for the subsequent development and implementation stages.

3.2.1.2 Requirement Gathering and Elicitation

Once the project proposal is finished, the next step is to gather important data and information thought research on the existing systems' review. This stage of research is dedicated to examining comparable systems or technologies that are currently available on the market or being used by the target users. By analysing these existing systems, the strengths, weaknesses, features, and technologies can be recognized. These data are very important for guiding the design and development of the project, so that the features and functions that the system should provide can meet the expectations of users and solve the problems found in the existing system. Therefore, the requirements specifications can be defined based on the analysis.

3.2.1.3 Project Scheduling

The last task to be done in planning phase is project scheduling. Once the project scheduling begins, a Work Breakdown Structure (WBS) is developed to outline all the tasks and subtasks needed to complete the project. Each task is broken

down into smaller components to better organize and allocate resources throughout the Software Development Life Cycle (SDLC). It helps ensuring all the tasks are identified and will not miss out and better manage the time.

Besides, Gantt chart is created to visually represent the project schedule, displaying tasks, timelines, dependencies, and milestones. Gantt chart provides a clear timeline for the start and end of each task plan, which helps the team to track and monitor the progress effectively.

3.2.2 Analysis and Design

Once the planning phase is completed, moving to the analysis and design phase. During this phase, the objectives, project scope, and requirements specification will be analysed. After that, use case diagram, interface flow chart and prototypes will be generated. Use case diagram has come out to show the interactions between parents and children with the application. Each use case is detailed in use case descriptions to describe the process and interactions for the features such as time management, activity reports, website filtering, and location tracking. These documents help ensure that the system effectively fulfils user requirements.

In addition, the interface flow chart shows how parents and children navigate in the application. Once all this is clear, prototypes are developed to provide a functional application interface and basic features without coding. It has shown the overview idea of the application before implementing.

3.2.3 Implementation and Testing

After the analysis and design phase is completed, the process will go on to implementing and testing phase. This phase will be categorized into four system version. Set up connection and develop login and sign in will be the system version 1. Followed by the develop parents' application in system version 2 with modify profile, time management feature and activity summary report feature and system version 3 will develop the content feature and location tracking. In the last system version, system version 4 is develop the application for children with the feature view dashboard, SOS feature, and rewards.

3.2.3.1 System Version 1

In the first system version, the login and sign in account and user profile feature will be implement first. However, before beginning, the development tools need to be set up first. The things that need to do is create the repository, configure server and database, and connect the application to server and database. The connection needs to be tested to make sure that there is no failure. After sure that all is perform well, features can start to implement. Login and sign in feature for both parents' application and children's application is come first to develop. The user interface is created, followed to implement the feature and test the algorithm.

3.2.3.2 System Version 2

Moving to this version, the application for parent is started. The user interface of user profile will be created based on the requirements. The algorithm of the sub features such as edit user profile, add device, and remove device are then implemented and tested. Once there is no bug on this version, the next version can proceed. the features that will be implemented are time management and activity summary report feature. For time management feature, the implementing phase carries out to implement the sub function which are set and edit daily and weekly screen time, lock the device and receive notification. The testing phase will test each of functionality to make sure that can run without bug. For activity summary report feature, the implementing phase conducted to implement the view daily or weekly usage of screen time and application algorithms. After completing with the implementation, the testing phase will test on the functionalities to pass the process to next system version while the feature run correctly.

3.2.3.3 System Version 3

Same goes as previously, after creating the user interface for content filtering feature and location tracking the implementing phase will start to implement the sub feature for each. The sub feature of content filtering feature will be implemented and proceed to testing phase for testing purpose. During the implementing phase of location tracking feature, the sub feature of location tracking feature like view location, save certain place, delete certain saved place,

and receive notification will start implementing for algorithms. After implementing phase, testing phase will carry out to test each functionality to ensure that no error during implementing phase. Once there is no bug on this version, the next version can proceed.

3.2.3.4 System Version 4

In system version 4, the children's application is developed. The user interface of dashboard will be created based on the requirements. The algorithm of the view dashboard and add device is implemented and tested. After that, the SOS feature will be implemented after the user interfaces are created. During the implementing phase of SOS feature, the sub feature including send and stop SOS will start implementing for algorithms. After implementing phase, testing phase will carry out to ensure that no error during implementing phase. The user interface of rewards will then be created. The algorithms of view rewards and request claim rewards will be implemented, and each algorithm of sub feature will conduct the testing phase. Last but not least, as it is the final system version, testing will be conducted to test all the functions in the application to make sure that the application can run without errors.

3.2.4 Closing

After implementing and testing phase, the process now goes to closing phase. It also includes the final test of the entire application to ensure that all features work prefect without errors or defects. After testing, the user acceptance testing and system documentation are finalised. Therefore, the final project is ready to be shown.

3.3 Software Development Tools

3.3.1 Axure RP10

Axure RP10 is a powerful prototyping and wireframe tool for software development to create interactive prototypes and digital interface models. It provides animation, interaction and other functions to make it suitable for designing the designing complex and interactive mobile applications. It allows to create the functional prototypes without writing code. Therefore, a view of the interfaces can be implemented using Axure RP10.

3.3.2 Visual Studio Code

Visual Studio Code is one of the favorite code editor among the developer because it supports a numerous programming languages including JavaScript, PHP, CSS, HTML, and more. It can also use for coding with Dart programming language often associated with Flutter application development. It enables the developers to write and debug code efficiently. It also supports various extensions and plug-ins, allowing developers to customize their coding environment to meet their specific needs. In short, Visual Studio Code is a universal and efficient tool for Dart coding, which provides a smooth and efficient development experience.

3.3.3 Android Studio

Android Studio is a useful integrated development environment (IDE) used for developing mobile applications designed for Android devices, but also for iOS devices. It provides some tools, including visual layout editor, code editing function with syntax highlighting and auto-completion, debugging tools and built-in emulator for testing applications. In the project, Android Studio is used as a emulator that allows developers to simulate Android and iOS devices on the computer for testing and debugging.

3.3.4 Flutter

In the project, Flutter is used to develop cross-platform mobile applications. It acts as the main development framework, that enable to develop the mobile applications with a single codebase that can run on both Android and iOS devices. It offers some of pre-built widget that enable developers to easily create attractive and interactive user interfaces for the front-end. The hot reload feature enables developers to view real-time changes during coding, speeding up the development process and increasing efficiency.

3.3.5 Firebase

Firebase is used for back-end development and database for storing and managing application data. It provides features such as real-time database, authentication, web hosting, cloud storage, and data analysis, letting developers

to focus on creating engaging user interfaces without the need to manage server resources. Its real-time database helps in storing the data across devices. Overall, Firebase simplifies back-end development and provides an efficient tool for building excellent applications.

3.4 Work Breakdown Structure

0.0 Remote Parental Control Mobile Application

1.0 Planning

- 1.1 Analyse the project title
- 1.2 Study the background of problems
- 1.3 Define problems statement
- 1.4 Define project objectives
- 1.5 Define proposed solution
- 1.6 Define proposed approach
  - 1.6.1 Define development approach
- 1.7 Define project scope
  - 1.7.1 Identify the system scope
  - 1.7.2 Identify target users
  - 1.7.3 Identify the application features
- 1.8 Requirement gathering
  - 1.8.1 Review similar existing system
    - 1.8.1.1 Review Qustodio
    - 1.8.1.2 Review Mobicip
    - 1.8.1.3 Review Google Family Link
    - 1.8.1.4 Review Bark
  - 1.8.2 Compare similar existing system
    - 1.8.2.1 Compare the pros and cons
    - 1.8.2.2 Compare features
- 1.9 Literature review
  - 1.9.1 Research on evaluation of parental control
  - 1.9.2 Research on enhancement of parental control
  - 1.9.3 Research on similar existing system review
- 1.10 Requirement elicitation
  - 1.10.1 Choose the recommended features

1.10.2 Define the functional and non-functional requirements

1.10.3 Refine the functional and non-functional requirements

1.10.4 Finalize the functional and non-functional requirements

1.11 Project scheduling

1.11.1 Create Work Breakdown Structure

1.11.1.1 Identify main activities

1.11.1.2 Breakdown the activities into smaller tasks

1.11.2 Create Gantt Chart

1.11.2.1 Determine task dependency

1.11.2.2 Estimate duration of task

1.11.2.3 Draft Gantt chart

1.11.2.4 Refine Gantt chart

1.11.2.5 Finalize Gantt chart

2.0 Analysis and Design

2.1 Design use case diagram

2.2 Create use case diagram

2.3 Design interface flow diagram

2.4 Design prototype

3.0 Implementing and testing software version 1

3.1 Set up connection

3.1.1 Create Repository

3.1.2 Configure server and database

3.1.3 Connect the application to server and database

3.2 Test connection

3.3 Create application framework

3.4 Develop sign up and login features

3.4.1 Implementing

3.4.1.1 Create sign up and login user interface

3.4.1.2 Implement sign up and login algorithm

3.4.2 Testing

3.4.2.1 Test sign up and login features

4.0 Implementing and testing software version 2 (for parent application)

4.1 Develop user profile feature

4.1.1 Implementing

4.1.1.1 Create user profile user interface

4.1.1.2 Implement edit user information algorithm

4.1.1.3 Implement add devices algorithm

4.1.1.4 Implement remove devices algorithm

4.1.2 Testing

4.1.2.1 Test edit user information algorithm

4.1.2.2 Test add devices algorithm

4.1.2.3 Test remove devices algorithm

4.2 Develop time management feature

4.2.1 Implementing

4.2.1.1 Create time management user interface

4.2.1.2 Implement set daily and screen time algorithm

4.2.1.3 Implement edit daily and screen time algorithm

4.2.1.4 Implement lock device algorithm

4.2.1.5 Implement notification algorithm

4.2.2 Testing

4.2.2.1 Test set daily and screen time algorithm

4.2.2.2 Test edit daily and screen time algorithm

4.2.2.3 Test lock device algorithm

4.2.2.4 Test notification algorithm

4.3 Develop activity summary report feature

4.3.1 Implementing

4.3.1.1 Create activity summary report

4.3.1.2 Implement view daily or weekly usage of screen time algorithm

4.3.2 Testing

4.3.2.1 Test view daily or weekly usage of screen time algorithm

5.0 Implementing and testing software version 3 (for parents)

5.1 Develop content filtering feature

5.1.1 Implementing

5.1.1.1 Create content filtering user interface

5.1.1.2 Implement view categories to filter content algorithm

5.1.1.3 Implement block specific websites algorithm

5.1.1.4 Implement notifications receive algorithm

5.1.2 Testing

5.1.2.1 Test view categories to filter content algorithm

5.1.2.2 Test block specific websites algorithm

5.1.2.3 Test notifications receive algorithm

5.2 Develop location tracking feature

5.2.1 Implementing

5.2.1.1 Create location user interface

5.2.1.2 Implement view children's location algorithm

5.2.1.3 Implement save certain place algorithm

5.2.1.4 Implement delete certain saved place algorithm

5.2.1.5 Implement notifications algorithm

5.2.2 Testing

5.2.2.1 Test view children's location algorithm

5.2.2.2 Test save certain place algorithm

5.2.2.3 Test delete certain saved place algorithm

5.2.2.4 Test notifications algorithm

6.0 Implementing and testing software version 4 (for children application)

6.1 Develop dashboard feature

6.1.1 Implementing

6.1.1.1 Create dashboard user interface

6.1.1.2 Implement view dashboard algorithm

6.1.1.3 Implement add device algorithm

6.1.2 Testing

6.1.2.1 Test view dashboard algorithm

6.1.2.2 Test add device algorithm

6.2 Develop SOS feature

6.2.1 Implementing

6.2.1.1 Create SOS user interface

6.2.1.2 Implement send SOS algorithm

6.2.1.3 Implement stop SOS algorithm

6.2.2 Testing

6.2.2.1 Test send SOS algorithm

6.2.2.2 Test stop SOS algorithm

6.3 Develop rewards feature

6.3.1 Implementing

6.3.1.1 Create rewards user interface

6.3.1.2 Implement view rewards algorithm

6.3.1.3 Implement request claim rewards algorithm

6.3.2 Testing

6.3.2.1 Test view rewards algorithm

6.3.2.2 Test request claim rewards algorithm

7.0 Closing

7.1 Conduct user acceptance test

7.2 Create system documentation

7.3 Finalize the project documentation

3.5 Gantt Chart

3.5.1 The Project Schedule's Overview

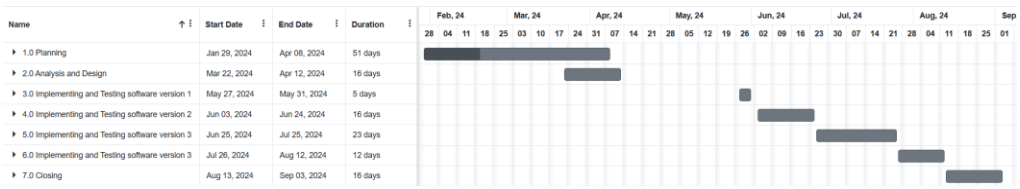
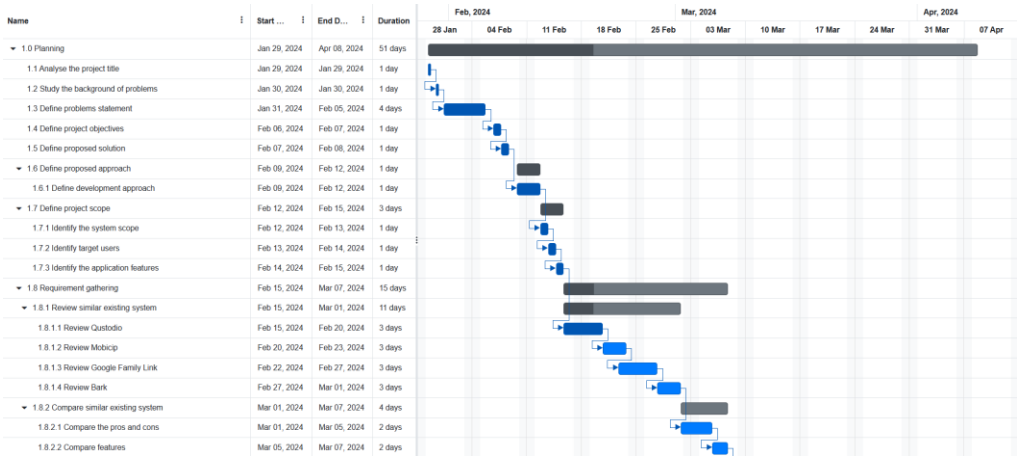


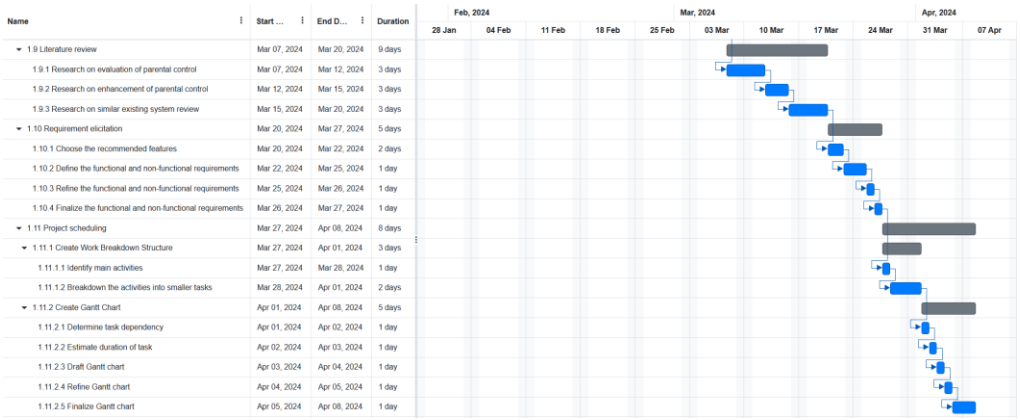
Figure 3.2: The Project Schdule's Overview

3.5.2 Planning Phase



Name	Start ...	End D...	Duration
1.0 Planning	Jan 29, 2024	Apr 08, 2024	51 days
1.1 Analyse the project title	Jan 29, 2024	Jan 29, 2024	1 day
1.2 Study the background of problems	Jan 30, 2024	Jan 30, 2024	1 day
1.3 Define problems statement	Jan 31, 2024	Feb 05, 2024	4 days
1.4 Define project objectives	Feb 06, 2024	Feb 07, 2024	1 day
1.5 Define proposed solution	Feb 07, 2024	Feb 08, 2024	1 day
1.6 Define proposed approach	Feb 09, 2024	Feb 12, 2024	1 day
1.6.1 Define development approach	Feb 09, 2024	Feb 12, 2024	1 day
1.7 Define project scope	Feb 12, 2024	Feb 15, 2024	3 days
1.7.1 Identify the system scope	Feb 12, 2024	Feb 13, 2024	1 day
1.7.2 Identify target users	Feb 13, 2024	Feb 14, 2024	1 day
1.7.3 Identify the application features	Feb 14, 2024	Feb 15, 2024	1 day
1.8 Requirement gathering	Feb 15, 2024	Mar 07, 2024	15 days
1.8.1 Review similar existing system	Feb 15, 2024	Mar 01, 2024	11 days
1.8.1.1 Review Qustodio	Feb 15, 2024	Feb 20, 2024	3 days
1.8.1.2 Review Mobicip	Feb 20, 2024	Feb 23, 2024	3 days
1.8.1.3 Review Google Family Link	Feb 22, 2024	Feb 27, 2024	3 days
1.8.1.4 Review Bark	Feb 27, 2024	Mar 01, 2024	3 days
1.8.2 Compare similar existing system	Mar 01, 2024	Mar 07, 2024	4 days
1.8.2.1 Compare the pros and cons	Mar 01, 2024	Mar 05, 2024	2 days
1.8.2.2 Compare features	Mar 05, 2024	Mar 07, 2024	2 days

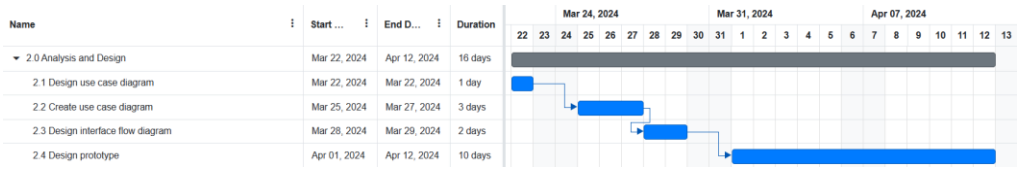
Figure 3.3: Schedule of Planning Phase



Name	Start ...	End D...	Duration
1.9 Literature review	Mar 07, 2024	Mar 20, 2024	9 days
1.9.1 Research on evaluation of parental control	Mar 07, 2024	Mar 12, 2024	3 days
1.9.2 Research on enhancement of parental control	Mar 12, 2024	Mar 15, 2024	3 days
1.9.3 Research on similar existing system review	Mar 15, 2024	Mar 20, 2024	3 days
1.10 Requirement elicitation	Mar 20, 2024	Mar 27, 2024	5 days
1.10.1 Choose the recommended features	Mar 20, 2024	Mar 22, 2024	2 days
1.10.2 Define the functional and non-functional requirements	Mar 22, 2024	Mar 25, 2024	1 day
1.10.3 Refine the functional and non-functional requirements	Mar 25, 2024	Mar 26, 2024	1 day
1.10.4 Finalize the functional and non-functional requirements	Mar 26, 2024	Mar 27, 2024	1 day
1.11 Project scheduling	Mar 27, 2024	Apr 08, 2024	8 days
1.11.1 Create Work Breakdown Structure	Mar 27, 2024	Apr 01, 2024	3 days
1.11.1.1 Identify main activities	Mar 27, 2024	Mar 28, 2024	1 day
1.11.1.2 Breakdown the activities into smaller tasks	Mar 28, 2024	Apr 01, 2024	2 days
1.11.2 Create Gantt Chart	Apr 01, 2024	Apr 08, 2024	5 days
1.11.2.1 Determine task dependency	Apr 01, 2024	Apr 02, 2024	1 day
1.11.2.2 Estimate duration of task	Apr 02, 2024	Apr 03, 2024	1 day
1.11.2.3 Draft Gantt chart	Apr 03, 2024	Apr 04, 2024	1 day
1.11.2.4 Refine Gantt chart	Apr 04, 2024	Apr 05, 2024	1 day
1.11.2.5 Finalize Gantt chart	Apr 05, 2024	Apr 08, 2024	1 day

Figure 3.4: Schedule of Planning Phase (cont.)

3.5.3 Anaylsis and Design Phase



Name	Start ...	End D...	Duration
2.0 Analysis and Design	Mar 22, 2024	Apr 12, 2024	16 days
2.1 Design use case diagram	Mar 22, 2024	Mar 22, 2024	1 day
2.2 Create use case diagram	Mar 25, 2024	Mar 27, 2024	3 days
2.3 Design interface flow diagram	Mar 28, 2024	Mar 29, 2024	2 days
2.4 Design prototype	Apr 01, 2024	Apr 12, 2024	10 days

Figure 3.5: Schedule of Analysis and Design Phase

3.5.4 Implementing and Testing Phase

3.5.4.1 Software Version 1

Name	Start Date	End Date	Duration
3.0 Implementing and Testing software version 1	May 27, 2024	May 31, 2024	5 days
3.1 Set up connection	May 27, 2024	May 28, 2024	2 days
3.1.1 Create Repository	May 27, 2024	May 27, 2024	1 day
3.1.2 Configure server and database	May 27, 2024	May 27, 2024	1 day
3.1.3 Connect the application to server and database	May 28, 2024	May 28, 2024	1 day
3.2 Test connection	May 28, 2024	May 28, 2024	1 day
3.3 Create application framework	May 29, 2024	May 29, 2024	1 day
3.4 Develop sign up and login features	May 30, 2024	May 31, 2024	2 days
3.4.1 Implementing	May 30, 2024	May 31, 2024	2 days
3.4.1.1 Create sign up and login user interface	May 30, 2024	May 30, 2024	1 day
3.4.1.2 Implement sign in and login algorithm	May 31, 2024	May 31, 2024	1 day
3.4.2 Testing	May 31, 2024	May 31, 2024	1 day
3.4.2.1 Test sign in and login features	May 31, 2024	May 31, 2024	1 day

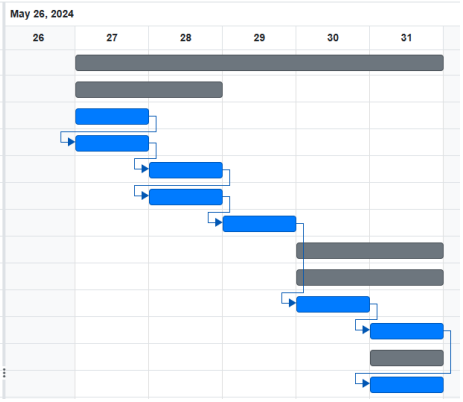


Figure 3.6: Schedule of Implementing and Testing Software Version 1 Phase

3.5.4.2 Software Version 2

Name	Start Date	End Date	Duration
4.0 Implementing and Testing software version 2	Jun 03, 2024	Jun 24, 2024	16 days
4.1 Develop user profile feature	Jun 03, 2024	Jun 07, 2024	5 days
4.1.1 Implementing	Jun 03, 2024	Jun 05, 2024	3 days
4.1.1.1 Create user profile user interface	Jun 03, 2024	Jun 03, 2024	1 day
4.1.1.2 Implement edit user information algorithm	Jun 04, 2024	Jun 04, 2024	1 day
4.1.1.3 Implement add devices algorithm	Jun 05, 2024	Jun 05, 2024	1 day
4.1.1.4 Implement remove devices algorithm	Jun 05, 2024	Jun 05, 2024	1 day
4.1.2 Testing	Jun 06, 2024	Jun 07, 2024	2 days
4.1.2.1 Test edit user information algorithm	Jun 06, 2024	Jun 07, 2024	2 days
4.1.2.2 Test add devices algorithm	Jun 06, 2024	Jun 06, 2024	1 day
4.1.2.3 Test remove devices algorithm	Jun 06, 2024	Jun 06, 2024	1 day
4.2 Develop time management feature	Jun 07, 2024	Jun 19, 2024	9 days
4.2.1 Implementing	Jun 07, 2024	Jun 17, 2024	7 days
4.2.1.1Create time management user interface	Jun 07, 2024	Jun 07, 2024	1 day
4.2.1.2 Implement set daily and screen time algorithm	Jun 10, 2024	Jun 11, 2024	2 days
4.2.1.3 Implement edit daily and screen time algorithm	Jun 12, 2024	Jun 12, 2024	1 day
4.2.1.4 Implement lock device algorithm	Jun 13, 2024	Jun 14, 2024	2 days
4.2.1.5 Implement notification algorithm	Jun 17, 2024	Jun 17, 2024	1 day

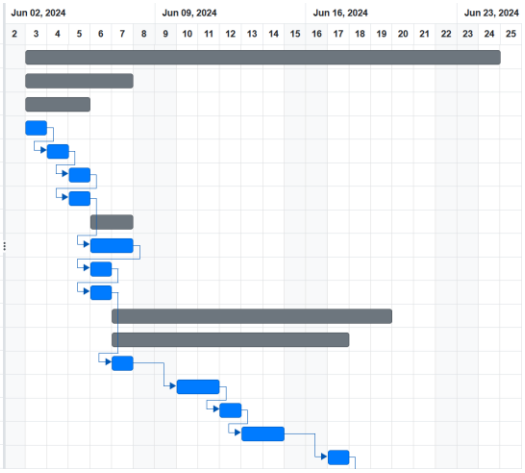


Figure 3.7: Schedule of Implementing and Testing Software Version 2 Phase

4.2.2 Testing	Jun 18, 2024	Jun 19, 2024	2 days
4.2.2.1 Test set daily and screen time algorithm	Jun 18, 2024	Jun 18, 2024	1 day
4.2.2.2 Test edit daily and screen time algorithm	Jun 18, 2024	Jun 18, 2024	1 day
4.2.2.3 Test lock device algorithm	Jun 19, 2024	Jun 19, 2024	1 day
4.2.2.4 Test notification algorithm	Jun 19, 2024	Jun 19, 2024	1 day
4.3 Develop activity summary report feature	Jun 20, 2024	Jun 24, 2024	3 days
4.3.1 Implementing	Jun 20, 2024	Jun 21, 2024	2 days
4.3.1.1 Create activity summary report	Jun 20, 2024	Jun 20, 2024	1 day
4.3.1.2 Implement view daily or weekly usage of scr...	Jun 21, 2024	Jun 21, 2024	1 day
4.3.2 Testing	Jun 24, 2024	Jun 24, 2024	1 day
4.3.2.1 Test view daily or weekly usage of screen tim...	Jun 24, 2024	Jun 24, 2024	1 day

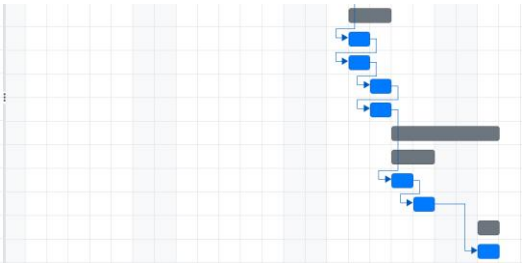
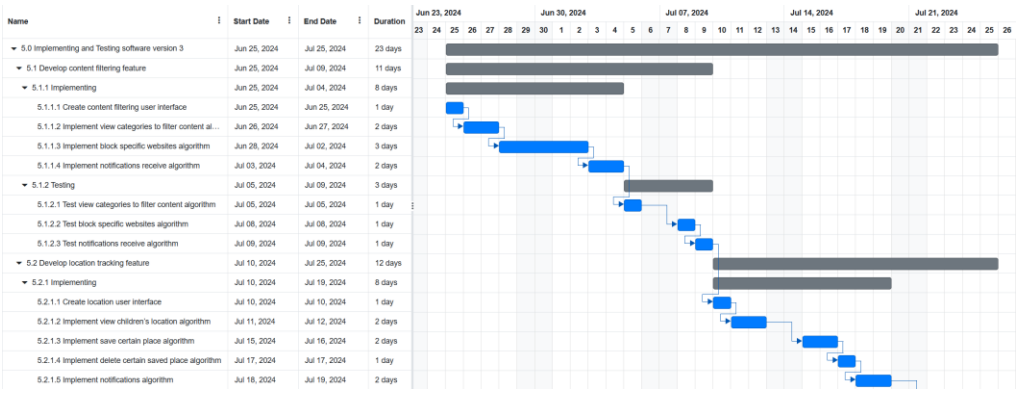


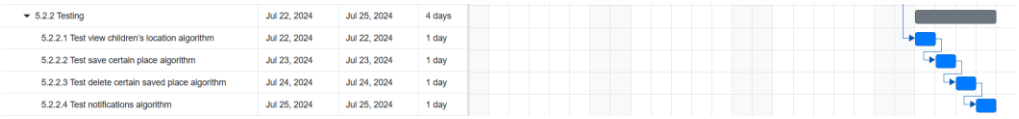
Figure 3.8: Schedule of Implementing and Testing Software Version 2 Phase (cont.)

3.5.4.3 Software Version 3



Name	Start Date	End Date	Duration
5.0 Implementing and Testing software version 3	Jun 25, 2024	Jul 25, 2024	23 days
5.1 Develop content filtering feature	Jun 25, 2024	Jul 09, 2024	11 days
5.1.1 Implementing	Jun 25, 2024	Jul 04, 2024	8 days
5.1.1.1 Create content filtering user interface	Jun 25, 2024	Jun 25, 2024	1 day
5.1.1.2 Implement view categories to filter content al...	Jun 26, 2024	Jun 27, 2024	2 days
5.1.1.3 Implement block specific websites algorithm	Jun 28, 2024	Jul 02, 2024	3 days
5.1.1.4 Implement notifications receive algorithm	Jul 03, 2024	Jul 04, 2024	2 days
5.1.2 Testing	Jul 05, 2024	Jul 09, 2024	3 days
5.1.2.1 Test view categories to filter content algorithm	Jul 05, 2024	Jul 05, 2024	1 day
5.1.2.2 Test block specific websites algorithm	Jul 08, 2024	Jul 08, 2024	1 day
5.1.2.3 Test notifications receive algorithm	Jul 09, 2024	Jul 09, 2024	1 day
5.2 Develop location tracking feature	Jul 10, 2024	Jul 25, 2024	12 days
5.2.1 Implementing	Jul 10, 2024	Jul 19, 2024	8 days
5.2.1.1 Create location user interface	Jul 10, 2024	Jul 10, 2024	1 day
5.2.1.2 Implement view children's location algorithm	Jul 11, 2024	Jul 12, 2024	2 days
5.2.1.3 Implement save certain place algorithm	Jul 15, 2024	Jul 16, 2024	2 days
5.2.1.4 Implement delete certain saved place algorithm	Jul 17, 2024	Jul 17, 2024	1 day
5.2.1.5 Implement notifications algorithm	Jul 18, 2024	Jul 19, 2024	2 days

Figure 3.9: Schedule of Implementation and Testing Software Version 3 Phase



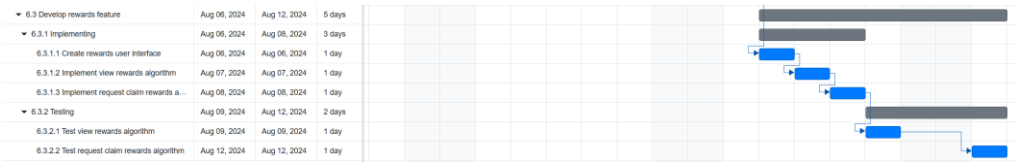
5.2.2 Testing	Jul 22, 2024	Jul 25, 2024	4 days
5.2.2.1 Test view children's location algorithm	Jul 22, 2024	Jul 22, 2024	1 day
5.2.2.2 Test save certain place algorithm	Jul 23, 2024	Jul 23, 2024	1 day
5.2.2.3 Test delete certain saved place algorithm	Jul 24, 2024	Jul 24, 2024	1 day
5.2.2.4 Test notifications algorithm	Jul 25, 2024	Jul 25, 2024	1 day

Figure 3.10: Schedule of Implementation and Testing Software Version 3 Phase (cont.)

3.5.4.4 Software Version 4

Name	Start D...	End Date	Duration
6.0 Implementing and Testing software version 3	Jul 26, 2024	Aug 12, 2024	12 days
6.1 Develop dashboard feature	Jul 26, 2024	Jul 30, 2024	3 days
6.1.1 Implementing	Jul 26, 2024	Jul 29, 2024	2 days
6.1.1.1 Create dashboard user interface	Jul 26, 2024	Jul 26, 2024	1 day
6.1.1.2 Implement view dashboard algorithm	Jul 29, 2024	Jul 29, 2024	1 day
6.1.1.3 Implement add device algorithm	Jul 29, 2024	Jul 29, 2024	1 day
6.1.2 Testing	Jul 30, 2024	Jul 30, 2024	1 day
6.1.2.1 Test view dashboard algorithm	Jul 30, 2024	Jul 30, 2024	1 day
6.1.2.2 Test add device algorithm	Jul 30, 2024	Jul 30, 2024	1 day
6.2 Develop SOS feature	Jul 31, 2024	Aug 05, 2024	4 days
6.2.1 Implementing	Jul 31, 2024	Aug 02, 2024	3 days
6.2.1.1 Create SOS user interface	Jul 31, 2024	Jul 31, 2024	1 day
6.2.1.2 Implement send SOS algorithm	Aug 01, 2024	Aug 01, 2024	1 day
6.2.1.3 Implement stop SOS algorithm	Aug 02, 2024	Aug 02, 2024	1 day
6.2.2 Testing	Aug 05, 2024	Aug 05, 2024	1 day
6.2.2.1 Test send SOS algorithm	Aug 05, 2024	Aug 05, 2024	1 day
6.2.2.2 Test stop SOS algorithm	Aug 05, 2024	Aug 05, 2024	1 day

Figure 3.11: Schedule of Implementation and Testing Software Version 4 Phase



6.3 Develop rewards feature	Aug 06, 2024	Aug 12, 2024	5 days
6.3.1 Implementing	Aug 06, 2024	Aug 08, 2024	3 days
6.3.1.1 Create rewards user interface	Aug 06, 2024	Aug 06, 2024	1 day
6.3.1.2 Implement view rewards algorithm	Aug 07, 2024	Aug 07, 2024	1 day
6.3.1.3 Implement request claim rewards a...	Aug 08, 2024	Aug 08, 2024	1 day
6.3.2 Testing	Aug 09, 2024	Aug 12, 2024	2 days
6.3.2.1 Test view rewards algorithm	Aug 09, 2024	Aug 09, 2024	1 day
6.3.2.2 Test request claim rewards algorithm	Aug 12, 2024	Aug 12, 2024	1 day

Figure 3.12: Schedule of Implementation and Testing Software Version 4 Phase (cont.)

3.5.5 Closing Phase

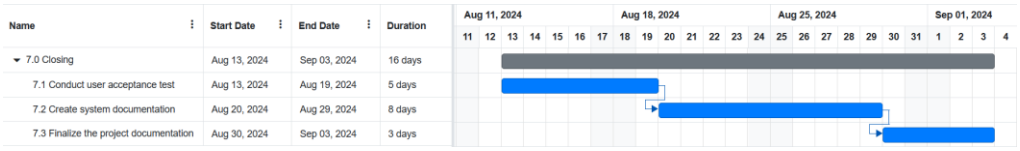


Figure 3.13: Schedule of Closing Phase

3.6 Summary

The chosen software development methodology is phased development methodology. In the planning phase, project proposal containing the important information, including problem statement, objectives, and project scope, requirement gathering from research on the review similar existing system, and project schedule with work breakdown structure was included. This has provided an overview of project that including important information. During the analysis and design phase, the use case diagram is along with the design of the interface flow diagram and prototype which show the layout of how the interface of application will look like.

Moving to the next phase, four system versions will carry out to implement and test the features. After system version 1 is completed then can follow to the next version. Lastly, in the closure of the project, a complete documentation such as user acceptance test and system documentation will be created.

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

This chapter includes the requirements specification that details the functional requirements and non-functional requirements of the application. The use case diagram and use case description were also completed to define the processes of the application. Lastly, the prototypes will be shown for the design and development of the application.

4.2 Requirements Specification

The functional requirements and non-functional requirements are involved in the requirements specification. The functional requirements specify what the project must do, and what features that must have in order to meet the expectations of the users. Whereas non-functional requirements outline the overall characteristics of the system. For instance, the non-functional requirements can be usability, security, availability, and performance.

4.2.1 Functional Requirement

FR01	The application shall allow the parents and children to login their account.
FR02	The application shall able the parents and children to modify the profile.
FR03	This application shall generate an activity summary report for parents and children to view.
FR04	The application should able the parents to set daily or weekly screen time limits.
FR05	The application should able the parents to block the specific websites.
FR06	The application should allow parents to track their children's real-time location.
FR07	This application shall allow parents to save a marked as saved place.

FR08	This application should able the parents to receive notification.
FR09	The application should the able children to view the time that they can use.
FR10	The application should allow children to send SOS to parents.
FR11	The application shall able children to claim rewards to request for extra time.

4.2.2 Non-functional Requirement

1.0 Usability

1.1 The application must be user-friendly simple and clear user interface, and easy to navigate which makes it simple for users to interact with.

1.2 The application should provide clear error messages and intuitive error handling methods to assist users in case of input errors or system issues.

2.0 Performance

2.1 The application shall response within 5 seconds.

2.2 The application shall send the alert to parents' devices within 5 seconds.

3.0 Security

3.1 The application should verify the users with a correct email and password before logging in.

4.0 Availability

4.1 The application shall be available to users at all the time, 24/7.

5.0 Compatibility

5.1 The application should be able to run seamlessly on Android and iOS devices with different resolutions and screen sizes.

4.3 Use Case Modelling

4.3.1 Use Case Diagram

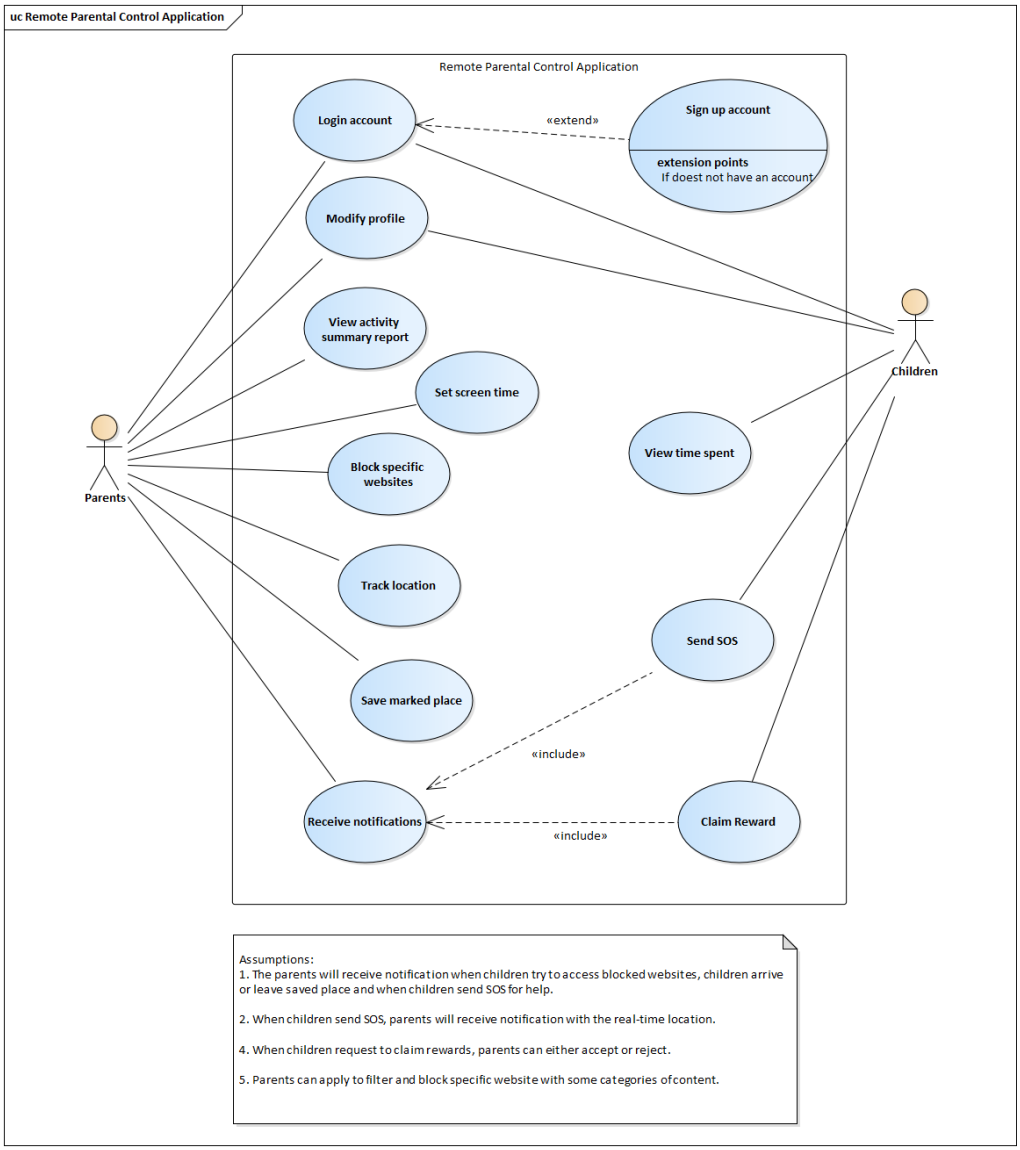


Figure 4.1: Use Case Diagram of The Proposed Project.

4.3.2 Use Case Description

Table 4.1: User Case Description of Login Account

Use Case Name: Login account	ID: 1	Importance Level: High
Primary Actor: Parents and Children	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – wants to login into their account and access the application. Children – wants to login into their account and access the application.		

Brief Description: This use case describes how parents and children login into their account to access the application.

Trigger: Parents and children want to login into their account and access the application.

Relationships:

Association : Parents and children
Include : -
Extend : Sign up account (ID: 2)
Generalization : -

Normal Flow of Events:

1. Parents and children launch the remote parental control application.
2. The Onboarding Screen shows in the application for parents and children to choose either get start as parents or children.
3. Parents choose for start as parents and children choose for start as children.
4. The application shows the Login Screen for parents, S1: Parent Login is performed
5. The application shows the Login Screen for children, S1: Children Login is performed.
6. Parents and children login the application successfully.
7. The application shows the Dashboard Screen.

Sub-flows:

S1: Parent Login

1. Parents enter the email and password.
2. The application checks account status.
    a. If the email address or password is invalid, an error message will pop out to ask for reenter.
    b. If the submit in empty field either for email address or password, an error message will pop out to ask for reenter.

S2: Children Login

1. Children enter their name and the child ID. 2. The application checks account status. a. If the username or child ID is invalid, an error message will pop out to ask for reenter. b. If the submit in empty field either for username or child ID, an error message will pop out to ask for reenter.
Alternate/Exceptional Flows: - 2a: If the parents or children do not have any account, "Sign up account" use case (ID: 2) is performed. - If parents or children enter invalid field, the application will pop out an error message and they can re-enter for the field.

Table 4.2: Use Case Description of Sign up Account

Use Case Name: Sign up account	ID: 2	Importance Level: High
Primary Actor: Parents and Children	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to sign in for register a new account to access the application. Children – want to sign in for link the new account to parent's application.		
Brief Description: This use case describes how parents and children sign in for register a new account to access the application.		
Trigger: Parents and children want to sign in for register a new account to access the application.		
Relationships: Association : Login account Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Parents and children launch the remote parental control application.		

2. The Onboarding Screen shows in the application for parents and children to choose either get start as parents or children.
3. Parents choose for start as parents and children choose for start as children.
4. The application shows the Login Screen.
5. Parents choose “Sign up” for register a new account, S1: Parent Sign up is performed.
6. Children click on the “Does not link yet” to link connection with parent, Child Device Connect is performed.
7. Parents and children direct navigate to the homepage of application.

Sub-flows:

S1: Paren Sign up

1. Parents enter their username, email and password with confirm password.
2. The application validates the data input.
    a. If the submit in empty field either for username, email address or password, an error message will pop out to ask for reenter.
    b. If the email address is not in the right format with “@”, or password is not more than 8 character, or confirmation password is not match with password, an error message will pop out to ask for reenter.
3. Parents confirm to create a new account.

S2: Child Device Connect

4. Children enter their username, the code generated by parent and age.
5. The application validates the data input.
    a. If the submit in empty field either for username, code or age, an error message will pop out to ask for reenter.
    b. If the code entered is invalid the error message will pop out.
6. Children confirm for login the application.

Alternate/Exceptional Flows: - If parents enter invalid email address and password, the application will pop out an error message and they can re-enter the email address and password.

Table 4.3: Use Case Description of Modify Profile.

Use Case Name: Modify profile	ID: 3	Importance Level: High
Primary Actor: Parents and children	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to edit their personal information with password and add and remove device. Children – want to edit their personal information with and age.		
Brief Description: This use case describes how parents modify their profile by editing the personal information such as username and email address or add and remove device and children want to modify personal information by username and email address.		
Trigger: Parents and children want to modify their personal information.		
Relationships: Association : Parents Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Parents and children select profile icon on the right bottom of the screen. 2. The application shows the personal information including, username, and email address for parent and username, id and age for children. 3. Parents who want to modify profile, S1: Parents modify profile is performed. 4. Children who want to modify profile, S2: Children modify profile is performed.		

5. Parents and children will back to the Profile Screen.
Sub-flows: S1: Parents modify profile. 1. Parents view their personal information with username, email address, and children. 2. Parents click the “change password” for changing the password. 3. Parents enter the current password, and new password with confirm password for successfully changed. 4. Parents click on the “add device” button. 5. The system shows a code for parents to link with children’s application. 6. Parents enter the code provided on the children’ device. 7. Parents select the device for removed by clicking the “remove” icon. S2: Children modify profile 1. Children view their personal information with username, ID and age. 2. Children can direct edit on the age at the field. 3. Children click “save” button to save the changes.
Alternate/Exceptional Flows: - If parents enter incorrect current password or the new password and confirm password are not match, the application will pop out an error message and they can re-enter the password.

Table 4.4: Use Case Description of View Activity Summary Report.

Use Case Name: View activity summary report	ID: 4	Importance Level: High
Primary Actor: Parents, Children	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to view the report that provide the overview of their children's online activities.		

Brief Description: This use case describes how parents view the report that provide the overview of their children's online activities.
Trigger: Parents want to view the report that provide the overview of their children's online activities.
Relationships: Association : Parents Include : - Extend : - Generalization : -
Normal Flow of Events: 1. Parents select home icon on the left bottom of the screen. 2. Parents select the children user from the profile. 3. The system will display the dashboard of a summary for children's online activities, including screen time limit today and the application usage of the children.
Sub-flows:-
Alternate/Exceptional Flows: -

Table 4.5: Use Case Description of Set Limit Time.

Use Case Name: Set Screen Time	ID: 5	Importance Level: High
Primary Actor: Parents	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to set a daily or weekly screen time limit for their children.		
Brief Description: This use case describes how parents set daily or weekly screen time limit for their children.		
Trigger: Parents want to set a daily or weekly screen time limit for their children.		

Relationships: Association : Parents Include : - Extend : - Generalization : -
Normal Flow of Events: 1. Parents select home icon on the left bottom of the screen. 2. Parents select the children user from the profile. 3. The system will display the children's dashboard. 4. Parents click for the set time button below the dashboard. 5. The application displays the daily screen time limit screen which allow the parents to view the daily and weekly time set. 6. Parents can select for the day from the dropbox. 7. The application shows the time set for parents to set. After setting, press done button to save it. 8. Parents press the back button to navigate to Dashboard Screen.
Sub-flows:-
Alternate/Exceptional Flows: - Parents are not allowed to set the time more than 3 hours.

Table 4.6: Use Case Description of Block Specific Websites

Use Case Name: Block specific websites	ID: 6	Importance Level: High
Primary Actor: Parents	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to filter and block the access for specific websites.		
Brief Description: This use case describes how parents apply filter to websites' content and block the access for specific websites.		

Trigger: Parents want to apply filter to websites' content block the access for specific websites.
Relationships: Association : Parents Include : - Extend : - Generalization : -
Normal Flow of Events: 1. Parents select home icon on the left bottom of the screen. 2. Parents select the children user from the profile. 3. The system will display the children's dashboard. 4. Parents click for the "filter" button below the dashboard. 5. The application navigates to Content Filtering Screen that shows the categories of websites. 6. Parents can enable the switch for apply filtering that help parents to filter the content of the website and direct block the access to that website when the inappropriate content is detected. 7. Parents can also block the category of websites manually by enabling the switch according to the categories of websites.
Sub-flows:-
Alternate/Exceptional Flows: - Enabling the switch and make it got colour then the websites are filtered and blocked

Table 4.7: Use Case Description of Track Location

Use Case Name: Track location	ID: 7	Importance Level: High
Primary Actor: Parents	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to track their children real-time location.		

Brief Description: This use case describes how parents track their children real-time location.

Trigger: Parents want to track their children real-time location.

Relationships:

Association : Parents

Include : -

Extend : -

Generalization : -

Normal Flow of Events:

1. Parents select location icon on the bottom of the screen.
2. The application shows the children's location with icon in the map.
3. Parents click on the icon and the detail location will be shown.
4. The application shows the detail of the children such as the children's name, and location.
5. Parents press back button to navigate back to the Dashboard screen.
6. Parents view the real-time location of children even the children are moving.
7. Parents mark a location place to save the place when they long press on the maps, U1: Save place.

Sub-flows:

U1: Saved place

1. Parents mark a place, and the detail of the location will be shown.
2. If parents want to add the marked place, U1.1 performed.
3. If the place is marked, and parent want to modify, U1.2 is performed.

U1.1: Add place

1. Parents click the add icon for saved the marked the place.
2. Parents remark the place with the name such as school and home.

U1.2: Modify place

1. If parents want to edit the name of the saved place, parents on tap on the marked place and the detail will show with the place name, radius and colour. 2. Parents are allow the save when press the “Save” button and they can also delete the marked place when press the “delete” button at the same dialog.
Alternate/Exceptional Flows: -

Table 4.8: Use Case Description of Save Marked Place.

Use Case Name: Save marked place	ID: 8	Importance Level: High
Primary Actor: Parents	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – want to save the marked place to know when children arrive or leave the saved place.		
Brief Description: This use case describes how parents save the marked place to know when children arrive or leave the saved place.		
Trigger: Parents want to save the marked place to know when children arrive or leave the saved place.		
Relationships: Association : Parents Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Parents select location icon on the bottom of the screen. 2. The application shows the children’s location with icon in the map. 3. Parents click on the icon and the detail location will be shown. 4. The application shows the detail of the children such as the children’s name, and location.		

5. Parents long press on the map and a dialog will display. 6. Parents mark the place with the name such as school and home, radius and set the colour. 7. Parents press save button to navigate back to the Location Screen.
Sub-flows: -
Alternate/Exceptional Flows: -

Table 4.9: Use Case Description of Receive Notification.

Use Case Name: Receive notifications	ID: 9	Importance Level: High
Primary Actor: Parents	Use Case Type: Detail and real	
Stakeholders and Interests: Parents – can receive notification when the children send request or SOS.		
Brief Description: This use case describes how parents receive notification when the children send request or SOS.		
Trigger: When children send SOS or request to claim the rewards and want parents to manage their request by accepting or rejecting.		
Relationships: Association : Parents Include : Send SOS (ID: 11), Claim rewards (ID: 12) Extend : - Generalization : -		
Normal Flow of Events: 1. Parents navigate to Notification Screen. 2. Parents view the list of all the notifications in the screen.		

3. Parents click the notification for view the details, S1: View notifications is performed.
Sub-flows: S1: View notifications 1. If parents click on the notification that is show about the SOS message send from children, U1: Send SOS is performed. 2. If parents click on the notification that is show children request to claim the rewards, U2: Manage request is performed. U1: Send SOS 1. Parents click on the SOS message's notification such as children send SOS. 2. The application shows a message box that displays the real-time location of the children. 3. Parents can scroll left to delete after read. U2: Manage request 1. Parents click on the request notification such as children request to claim rewards. 2. The application shows a message box that describe the detail of the task that children request to claim rewards. 3. Parents can choose for accept or scroll left as reject. 4. The message box will disappear and still in the Notification Screen.
Alternate/Exceptional Flows: -

Table 4.10: Use Case Description of View Time Spent.

Use Case Name: View time spent	ID: 10	Importance Level: High
Primary Actor: Children	Use Case Type: Detail and real	
Stakeholders and Interests: Children – want to view the time that they are allowed to use the device.		

Brief Description: This use case describes how children want to view the time that they are allowed to use the device.
Trigger: Children want to view the time that they are allowed to use the device.
Relationships: Association : Children Include : - Extend : - Generalization : -
Normal Flow of Events: 1. After children login to the children application, the system shows the Home Screen. 2. Home Screen direct displays the time that children have spent.
Sub-flows: -
Alternate/Exceptional Flows: -

Table 4.11: Use Case Description of Send SOS.

Use Case Name: Send SOS	ID: 11	Importance Level: High
Primary Actor: Children	Use Case Type: Detail and real	
Stakeholders and Interests: Children – can send SOS to parents when they are in trouble.		
Brief Description: This use case describes children send and SOS to parents when they are in trouble.		
Trigger: When are in trouble and ask for help, children can send and stop SOS to parents when they.		
Relationships: Association : Children Include : - Extend : - Generalization : -		

Normal Flow of Events: 1. The application shows the Home Screen. 2. Children navigate to the SOS Screen from the navigation bar. 3. Children can press the "SOS" circle to send message to parent. 4. An alert will pop out to show that the SOS message is sent. 5. The application will send the SOS to parent's device, which mean that parents will receive the notification, "Receive notification" in use case (ID: 9) is performed. 6. The application displays "Stop" for children to stop it when they are safety now or accidently press send. 7. Children press back button to navigate back to Home Screen.
Sub-flows: -
Alternate/Exceptional Flows: -

Table 4.12: Use Case Description of Claim Rewards

Use Case Name: Claim rewards	ID: 12	Importance Level: High
Primary Actor: Children	Use Case Type: Detail and real	
Stakeholders and Interests: Children – want to claim the rewards for extra screen time by request from parents.		
Brief Description: This use case describes how children claim the rewards for extra screen time by request from parents.		
Trigger: Children want to claim the rewards for extra screen time by request from parents when they have finished the tasks.		
Relationships: Association : Children Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. The application shows the Home Screen.		

2. Children navigate to Rewards Screen from the navigation bar. 3. Children choose to show the tabs which are Tasks and Rewards, S1: Manage rewards is performed.
Sub-flows: S1: Manage rewards 1. If children want to view the Tasks tab, S1.1: View task is performed. 2. If children want to view the Rewards tab, S1.2: View rewards is performed. S1.1: View task 1. Children click for the Tasks tab. 2. The application shows the tasks that the children can do for claim. 3. Children claim for the task that they had done, the request will then send to the parents for approve, "Receive notifications" in use case (ID:9) is performed. 4. After claiming, the task will not be able to claim in a day. S1.2 View rewards 1. Children click on the Rewards tab. 2. The application shows the tasks that is approved to the children. 3. Children click the "Add" button to add the time to their time that parents set to.
Alternate/Exceptional Flows: - If the task is claimed, it will disable for children to claim any more within a day. - If the extra time is added, it will disable for children to add to their time.

4.4 Prototype interface

A user interface prototype for the application is designed to visualize the idea of this project.

1. Onboarding

Figure 4.5 show the onboarding screen of the application for first launch. Parents can click on the Get Started as Parents for enter to parents' application while children can click on the Get Started as Children for enter to children application.

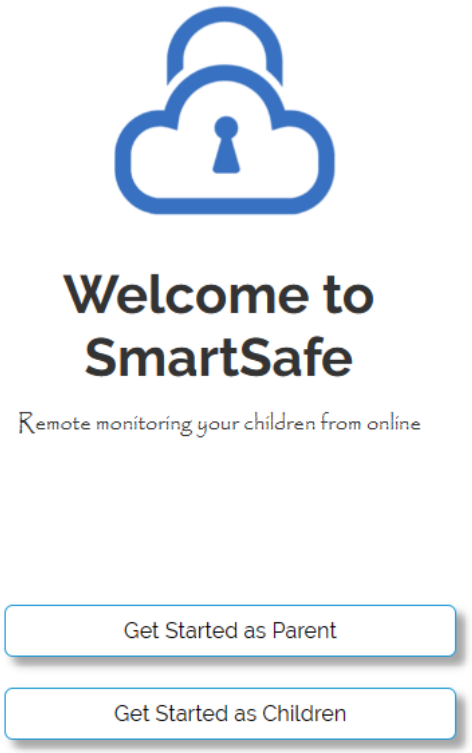


Figure 4.2: Prototype of Onboarding Screen.

2. Parent's application

2.1 Login Screen

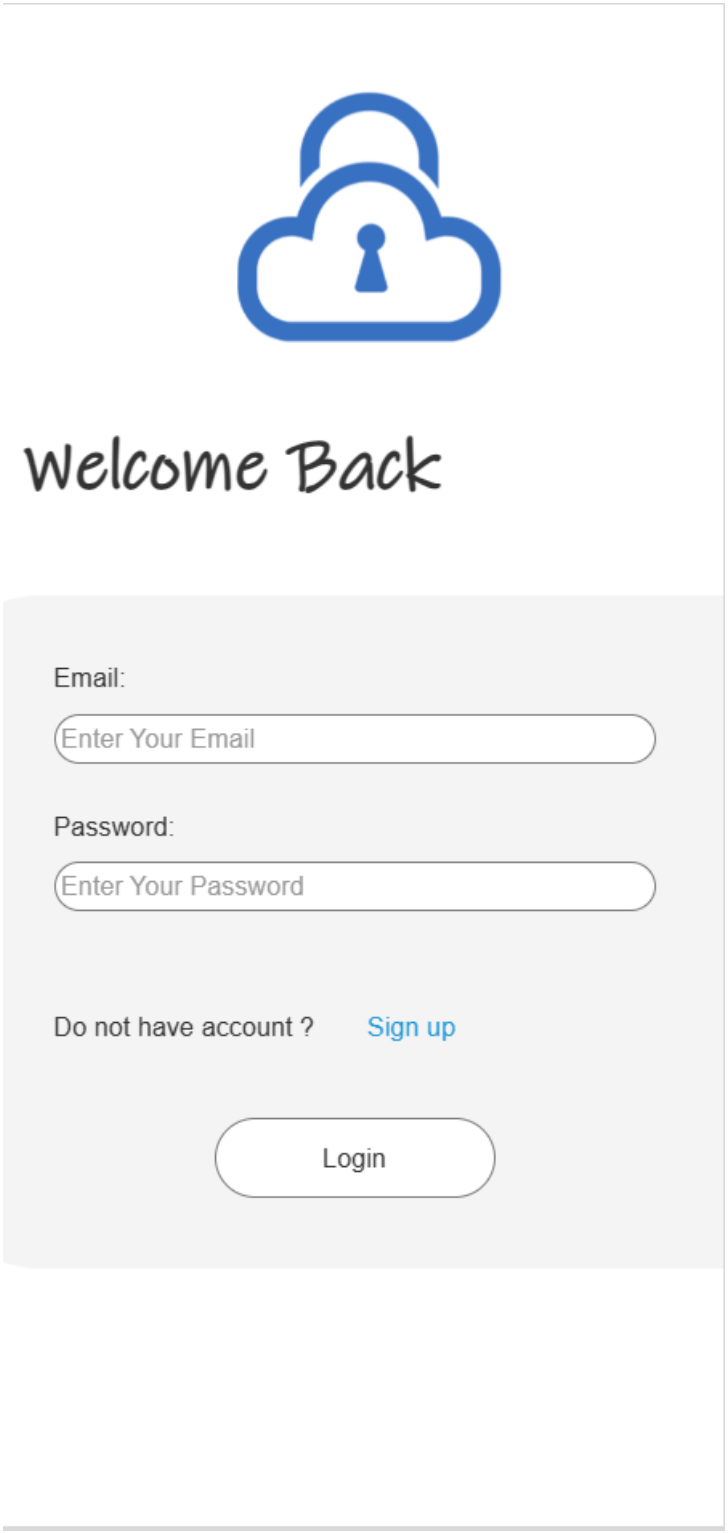


Figure 4.3: Prototype of Parents Login Screen.

2.2 Sign up Screen

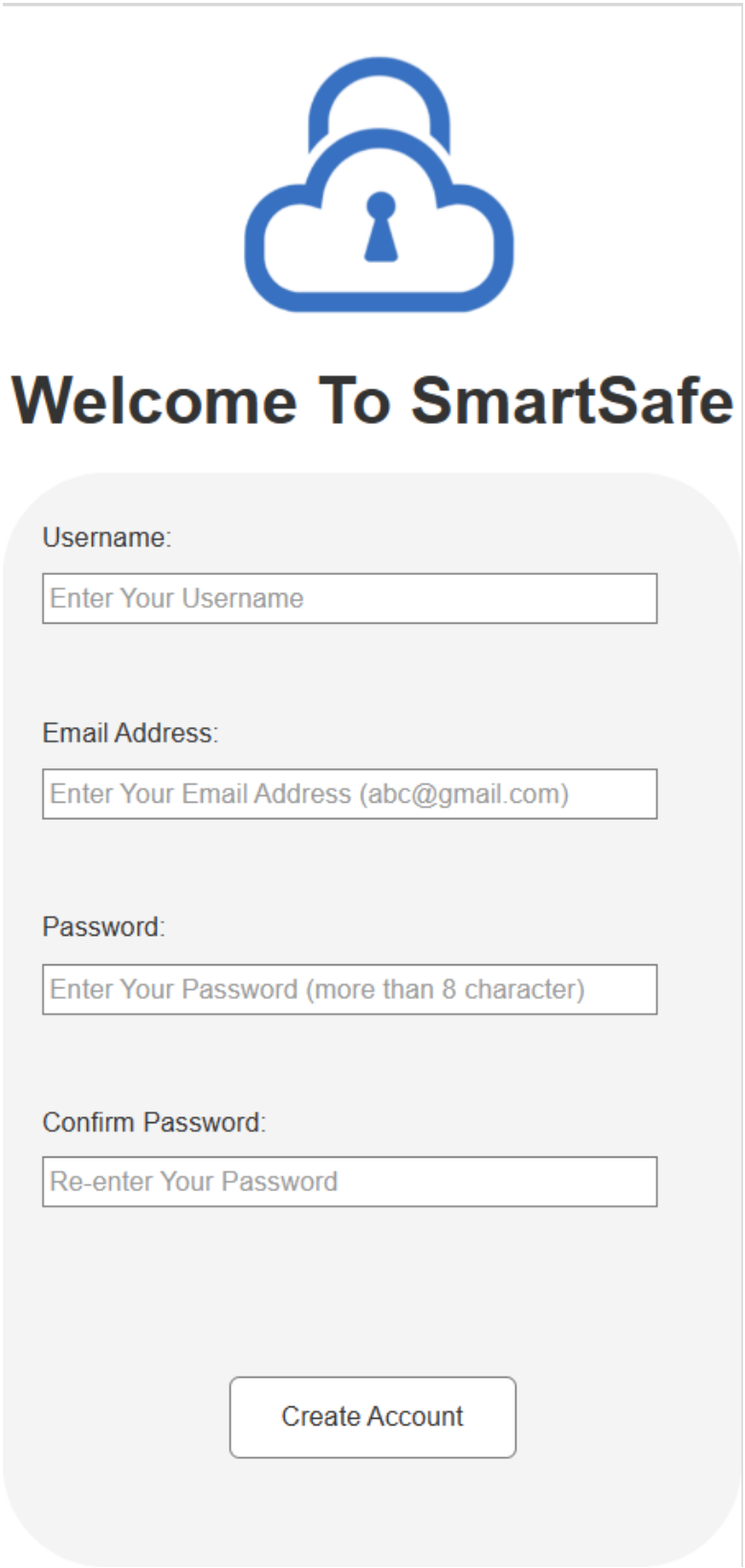


Figure 4.4: Prototype of Parent Sign up Screen.

2.3 Parent Home Screen

Home screen provides the navigation bar at the bottom with Home, Location, Notification, and Profile. Parents able to click to navigate to that screen. Home screen displays the children with the time spend chart that provide parents with the overview. Parents able to click on the children that they want to know, then the dashboard of the children will be shown. Besides, two features are provided, which are set time limit and content filter.

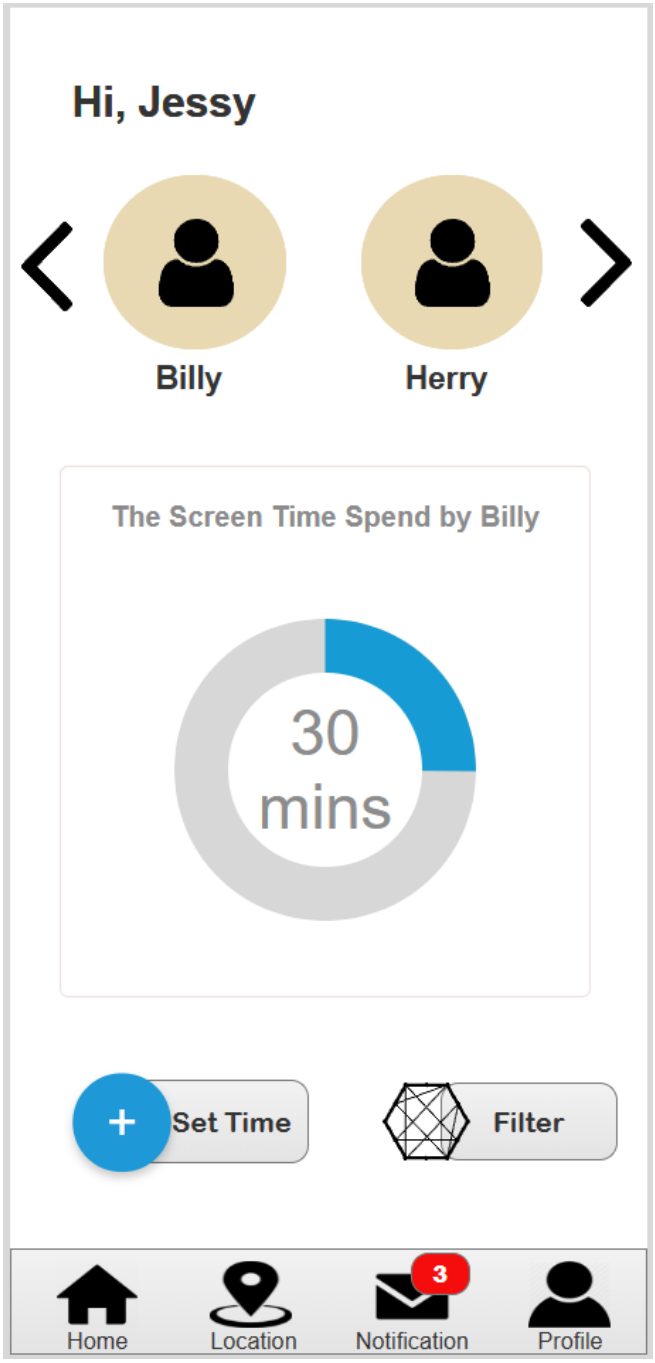


Figure 4.5: Prototype of Parents Home Screen.

2.3.1 Activity Summary Report Screen

Parents able to view the detail of the children's online activity summary report through click on the dashboard. After clicking, the application will navigate to the activity summary report screen. The activity summary report shows the screen with the screen time spend by the children and the time spent for application. Parents able to go back to Home Screen by clicking the left arrow.

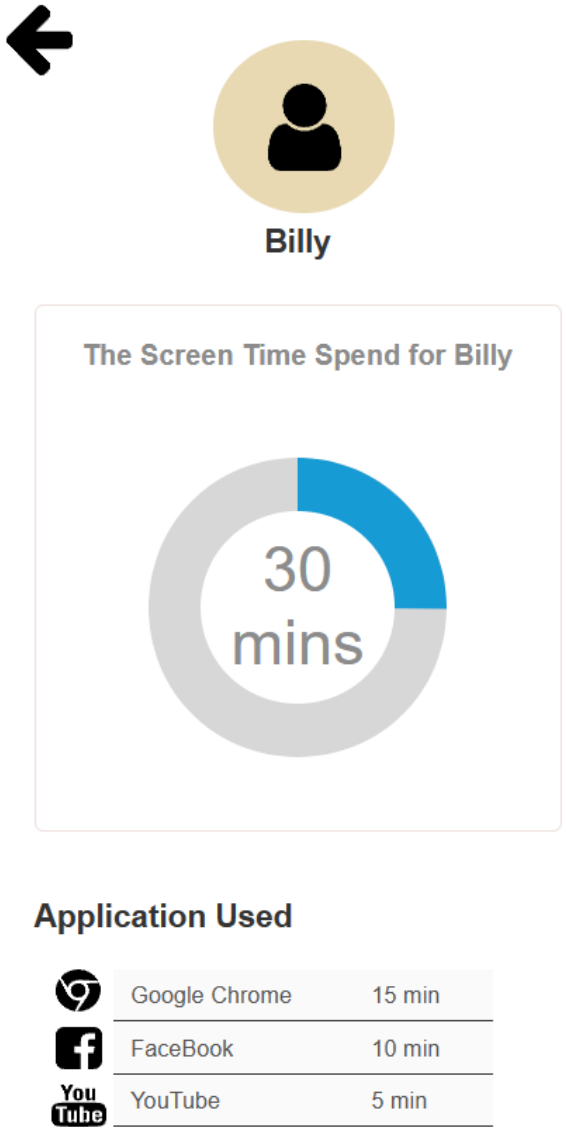


Figure 4.6: Prototype of Activity Summary Report Screen.

2.3.2 Set Time Limit Screen

In order to set the screen time limit, parents click on the “set time” below the dashboard at Home Screen. It will navigate to the set daily screen time limit screen. It shows a week of time and parents can set the time for each.

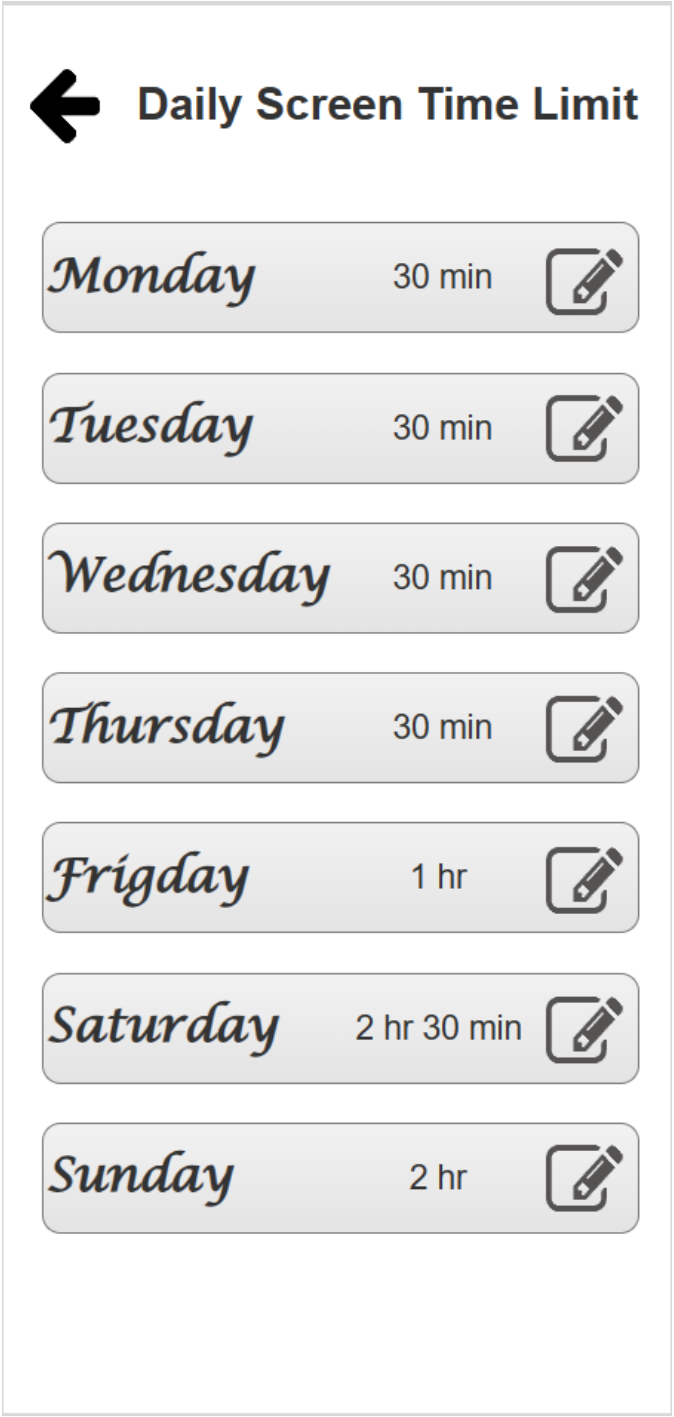


Figure 4.7: Prototype of Set Time Limit Screen.

2.3.3 Content Filtering Screen

For parents to apply the filtering feature, parents are able to click on the content for apply it, then the application will navigate to the content filtering screen. There is a switch for parents to enable for apply the filtering feature. In the screen also shows the categories of websites that parents can enable to block to the websites based on the categories.

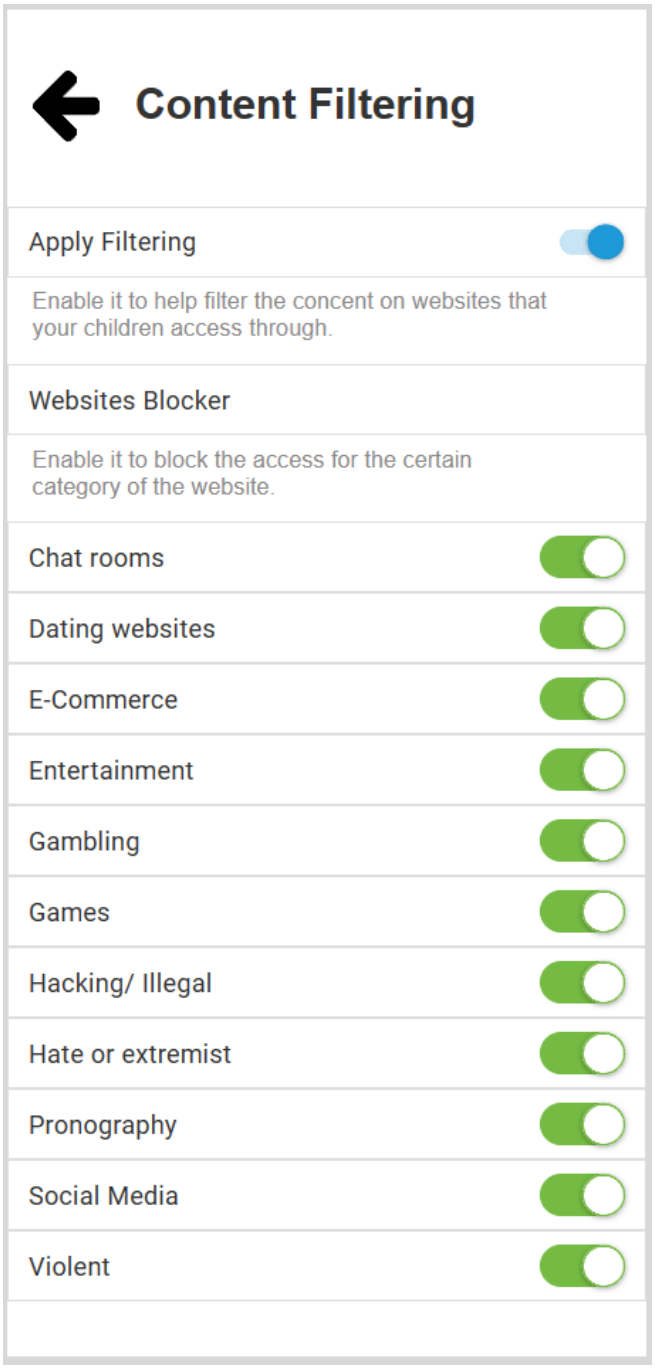


Figure 4.8: Prototype of Content Filtering Screen.

2.4 Location Screen

Parents are allowed to view their children's real-time location when they navigate to the location screen by the second icon of the navigation bar. Parents can click on the location of the children and view the detail on it with children's name, place, and location. Parents are also allowed to save the marked place with the love icon on the right top of the detail of the location.

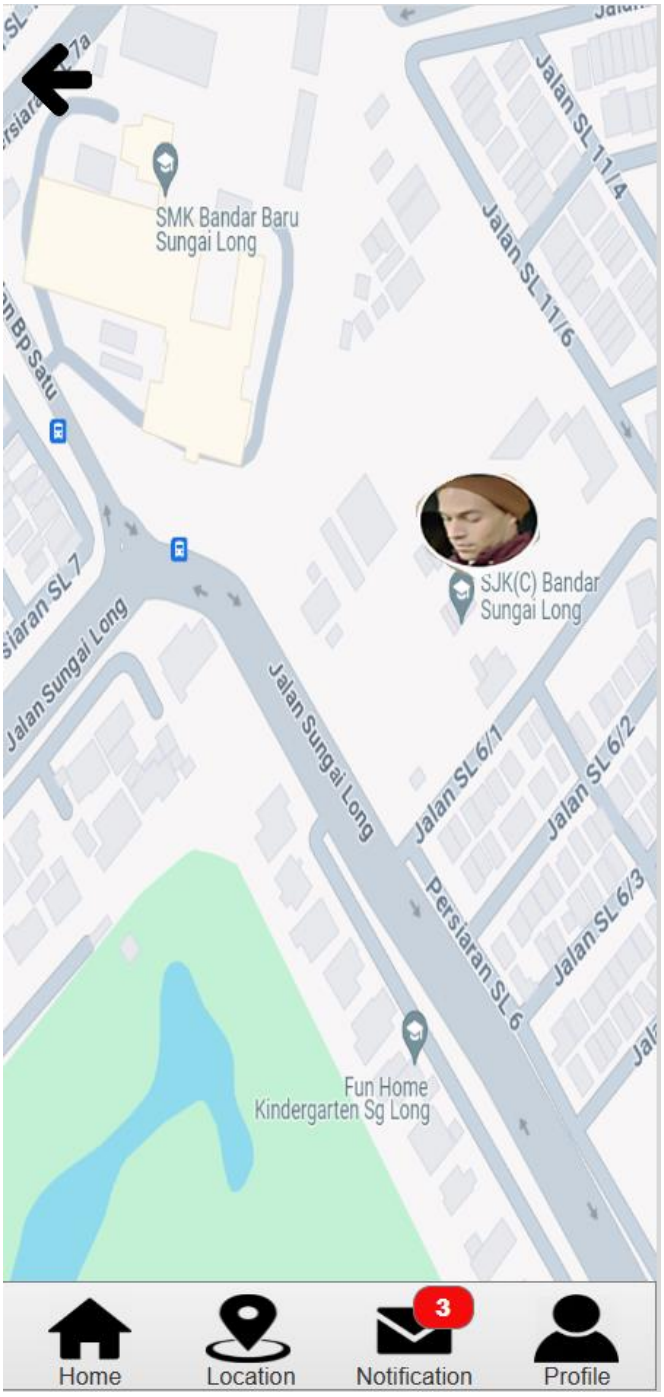


Figure 4.9: Prototype of Location Screen.

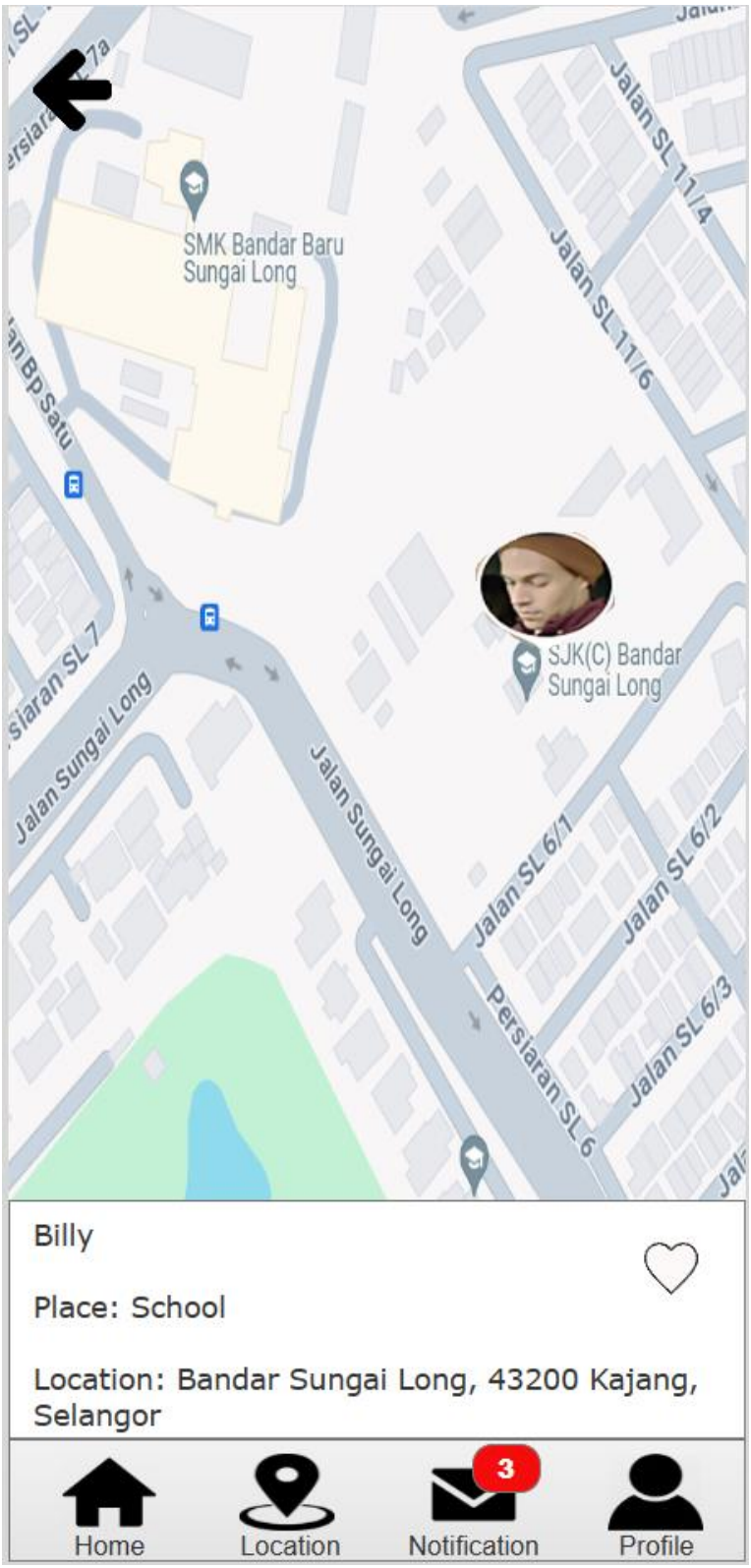


Figure 4.10: Prototype of Location Marked Screen.

2.5 Notification Screen

By navigating to notification screen, parents view the request for claim the rewards from the children, children try to access the blocked, children arrive or leave a saved place, and when children send SOS to them. Parents can click to the message to view the detail of the notification. For instance, parents can accept or reject for the request from the children.

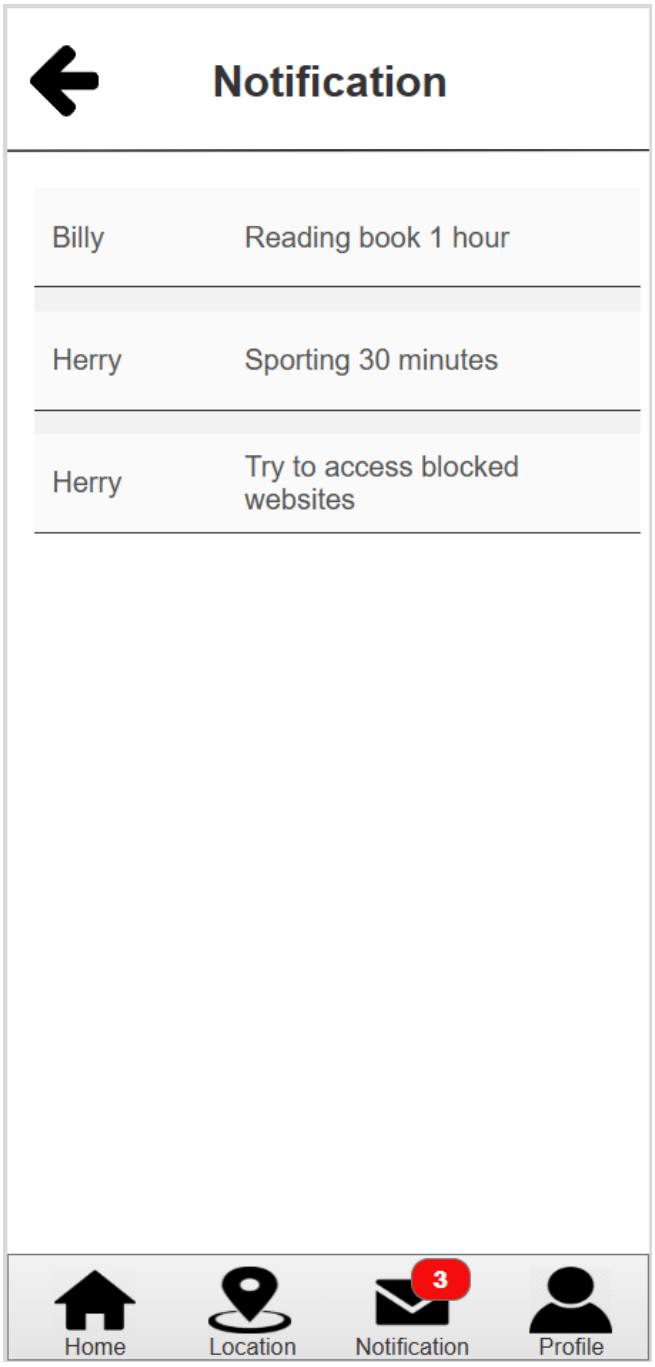


Figure 4.11: Prototype of Notification Screen.

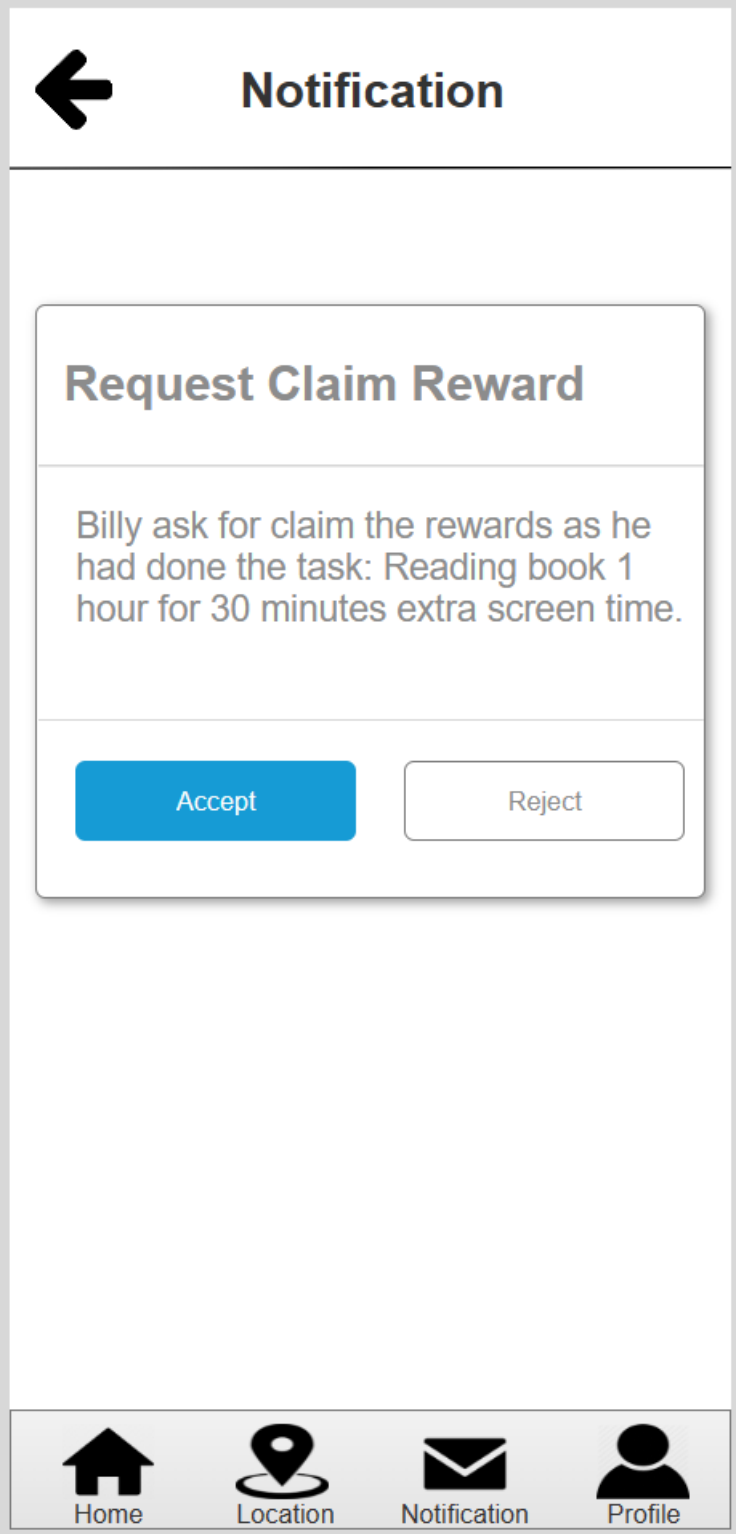


Figure 4.12: Prototype of Notification Screen (cont.).

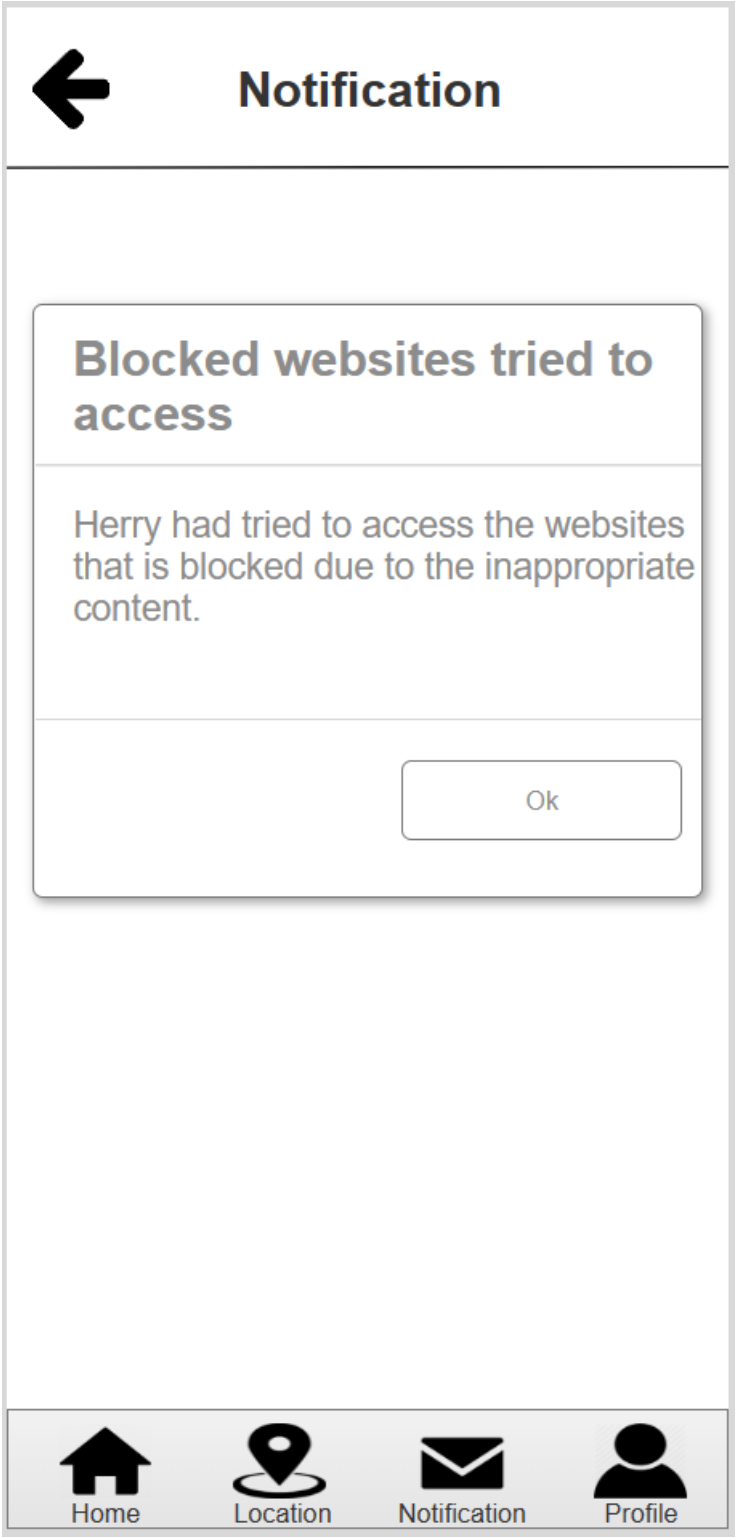


Figure 4.13: Prototype of Notification of Blocked Access Screen.

2.6 Parent User Profile

Parents able to view and edit the user profile by clicking the profile icon at the bottom right of the navigation bar. Parents can view their username, email

address, and children connected. Parents can also edit their personal information and add or remove children device through “edit”. For adding device, the application will provide a code for parents to enter to children’s device.

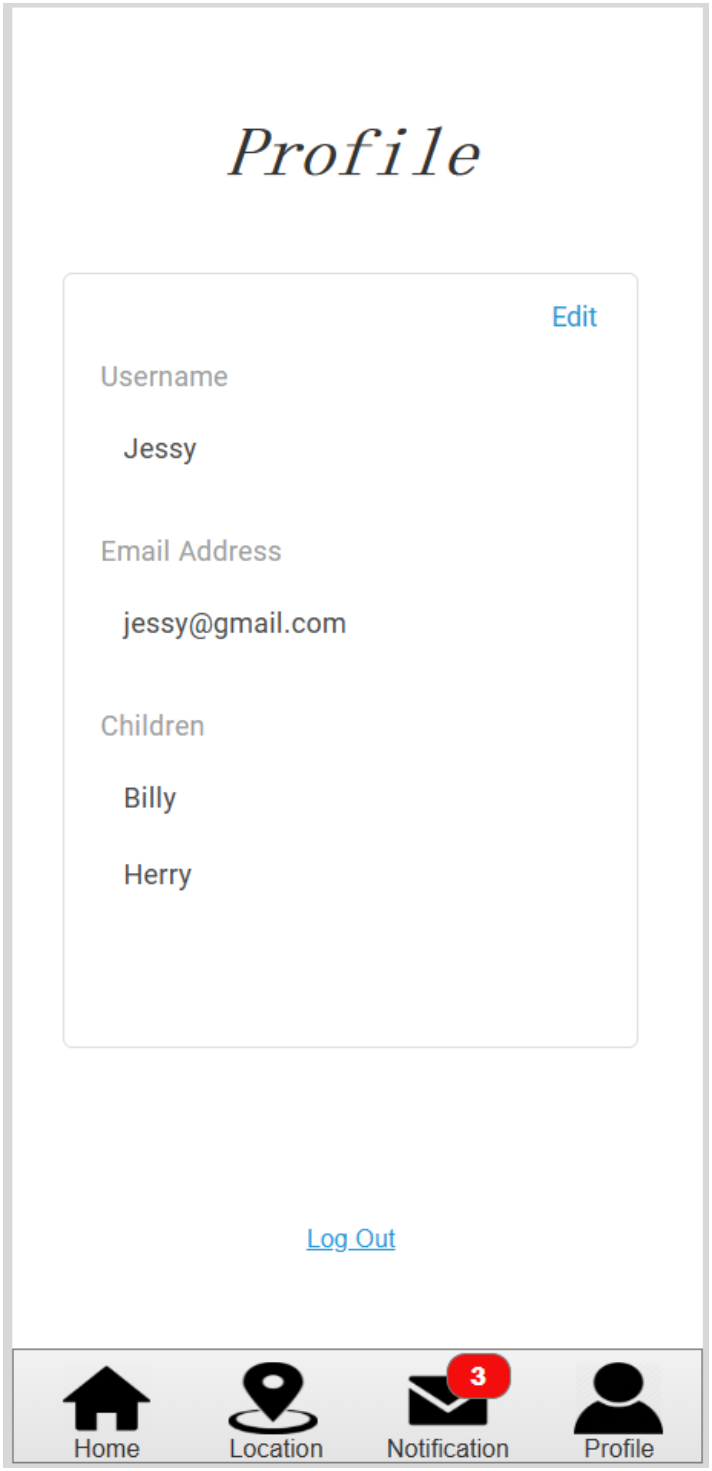


Figure 4.14: Prototype of Parents User Profile Screen.

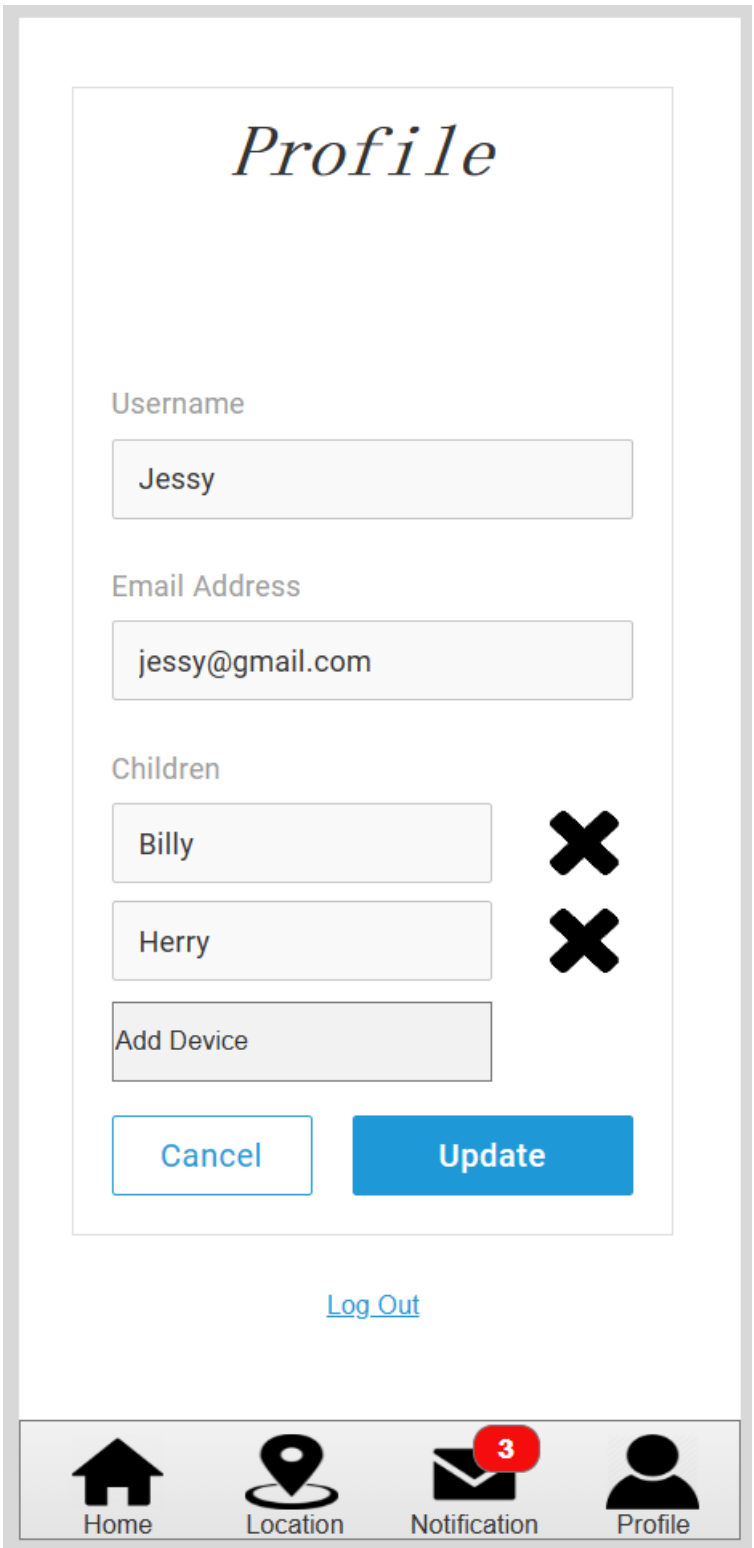


Figure 4.15: Prototype of Modify Parents User Profile Screen.

3. Children Application

3.1 Children Login Screen

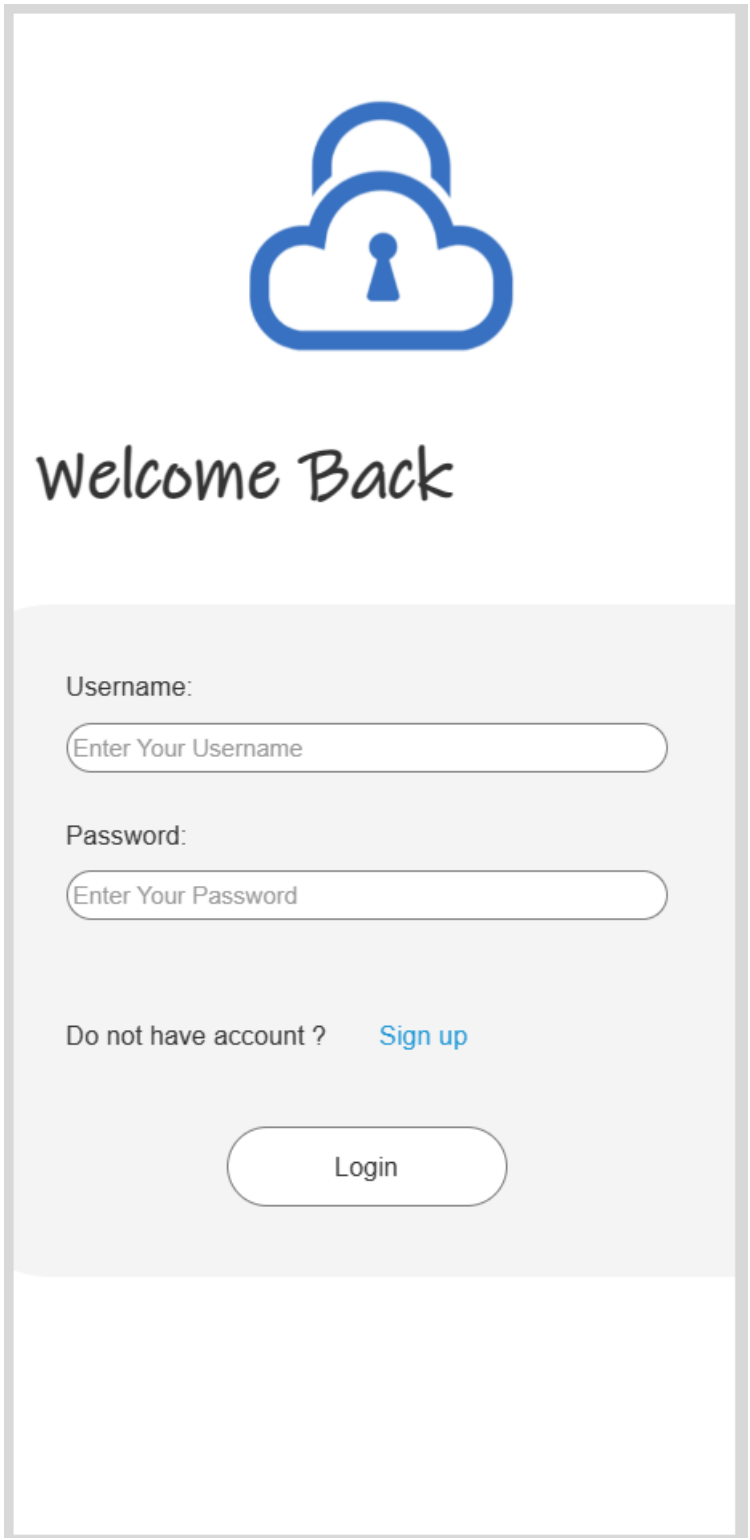


Figure 4.16: Prototype of Children Login Screen.

3.2 Children Sign Up Screen

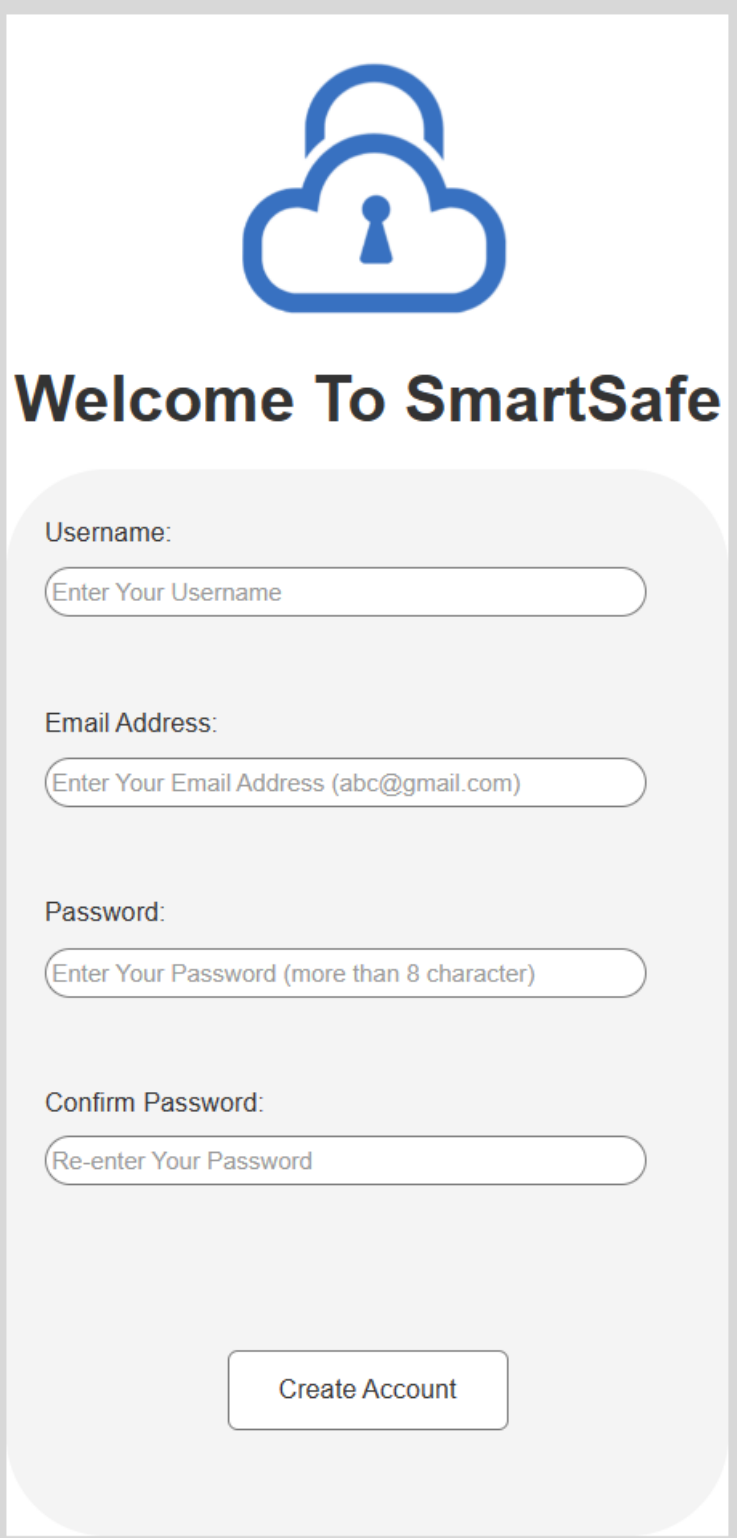


Figure 4.17: Prototype of Children Sign Up Screen.

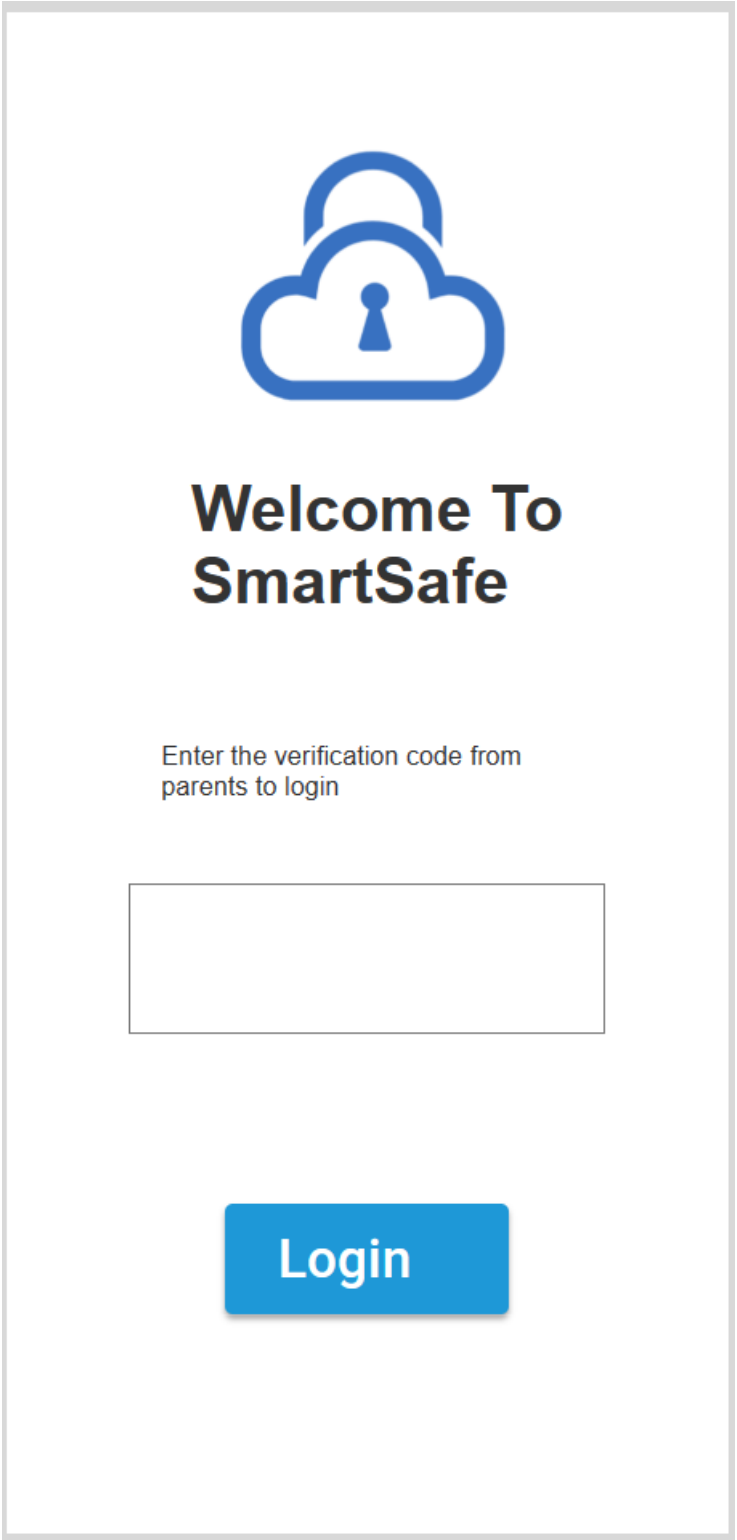


Figure 4.18: Prototype of Children Sign Up Screen (cont.).

3.3 Children Home Screen

At Home screen of the children's application, there are also a navigation bar for navigate to Home, SOS, Rewards, and Profile. Children able to view the screen time for them at the Home screen.

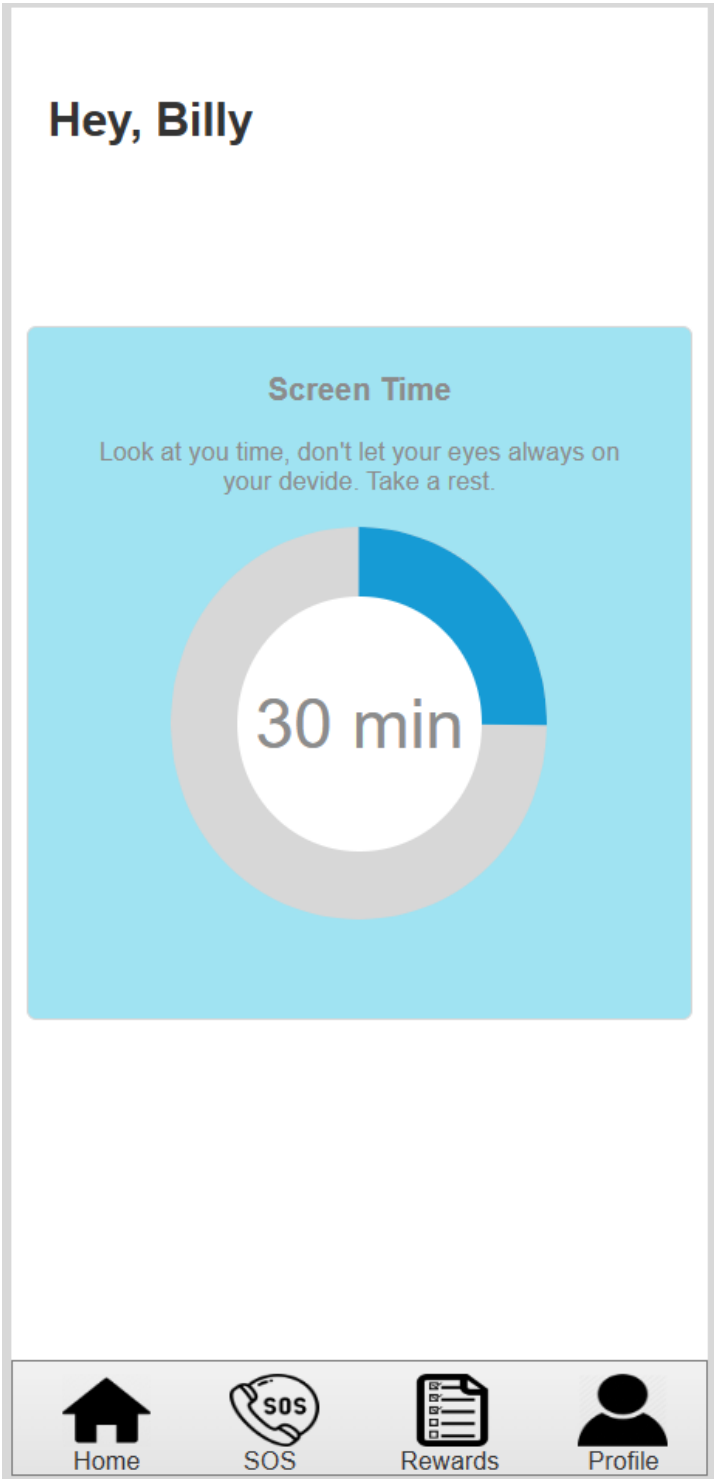


Figure 4.19: Prototype of Children Home Screen.

3.4 SOS Screen

After navigating to SOS screen, children are allowed to send and stop the SOS sending to parents' device. This use when they are in trouble.

Figure 4.20: Prototype of Send SOS Screen.

Figure 4.21: Prototype of Stop SOS Screen.

3.5 Rewards Screen

Children can view the daily task at the task tab that they can claim for extra screen. After doing the task, children can request to claim the rewards from parents. After parents approved, it will be shown in the rewards tab.

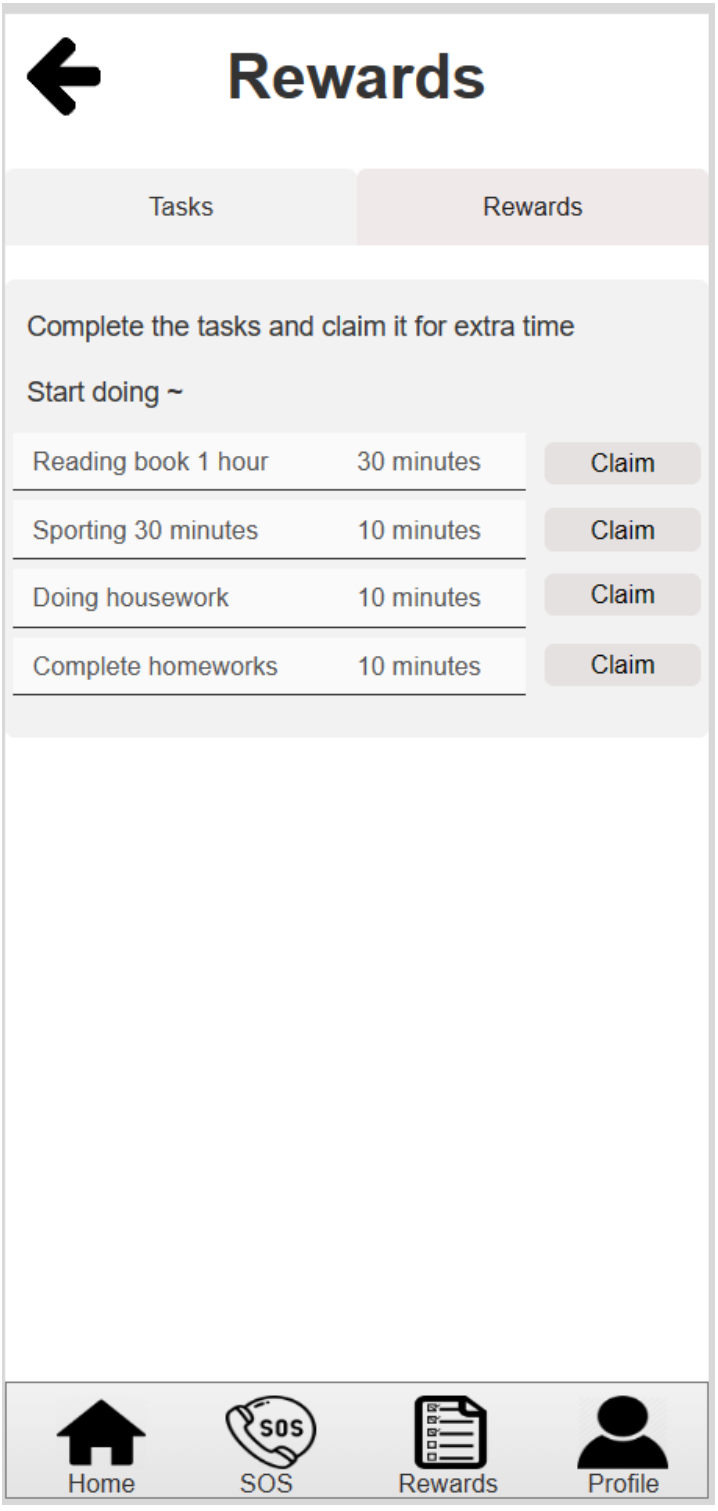


Figure 4.22: Prototype of Tasks of Rewards Screen.

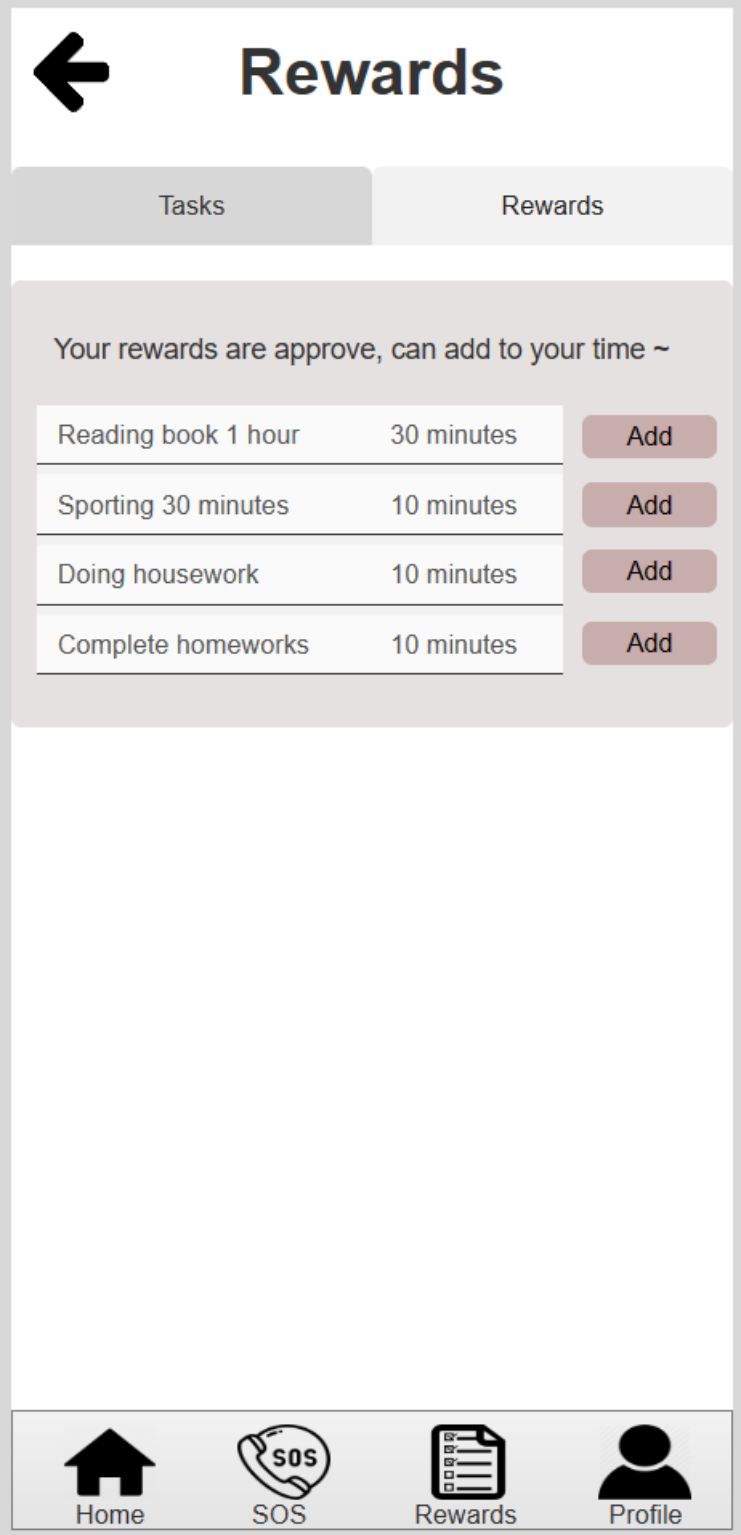


Figure 4.23: Prototype of Rewards of Rewards Screen.

3.6 Children User Profile

Children are also view and edit their profile at the profile icon. Children can edit for their username, email address, and the age of them.

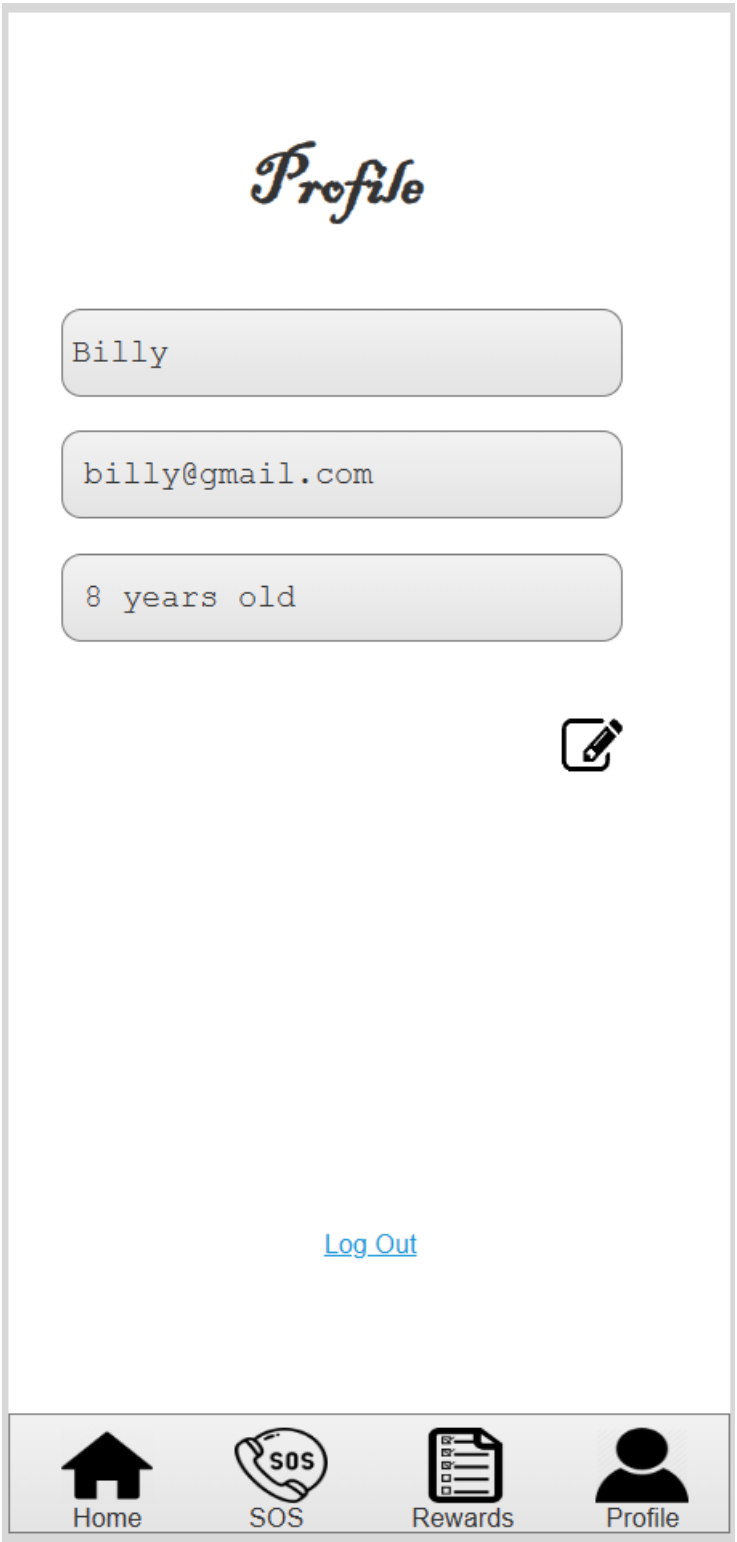


Figure 4.24: Prototype of Children Profile Screen.

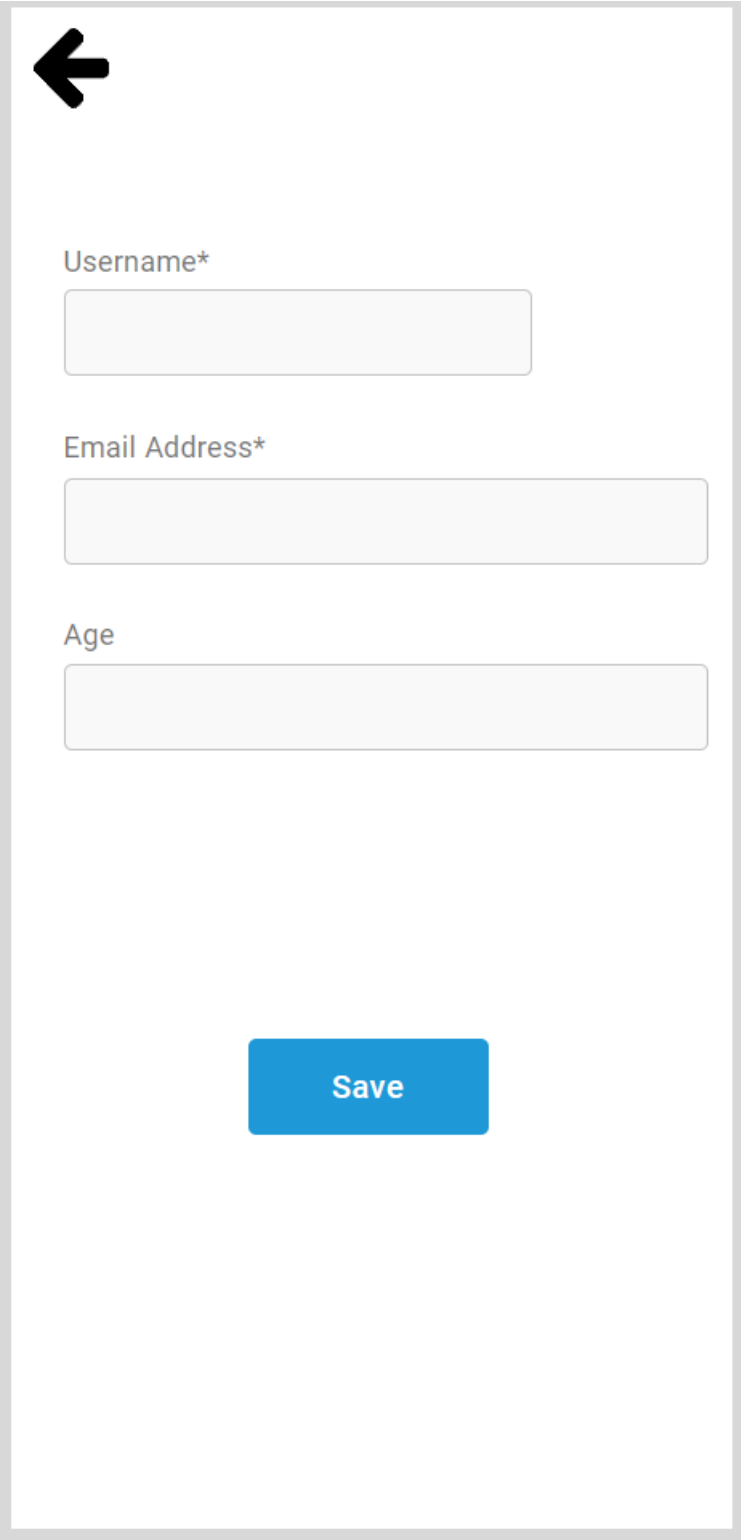


Figure 4.25: Prototype of Children Modify Profile Screen.

4.5 Summary

In summary, this chapter has detailed the requirements specification with both functional requirements and non-functional requirement. Functional

requirements have outlined the specific requirements in the application, such as screen time management, content filtering, parental alert and location tracking whereas non-functional requirements focus on the ensuring the application achieve success in some aspects such as performance, usability, availability, security, and compatibility. In this project, 11 functional requirements and 7 non-functional requirements in 5 aspects to make sure the project is developed to meet user's need.

Use case diagram and use case description are also presented to show the interaction of parents and children with application and describe the user action with the application response. Lastly, the user interface prototype is also included to show the layout and the design of the application to develop a user-friendly interface.

CHAPTER 5
DESIGN AND IMPLEMENTATION

5.1 Diagrams

5.1.1 Entity Relationship Diagram (ERD)

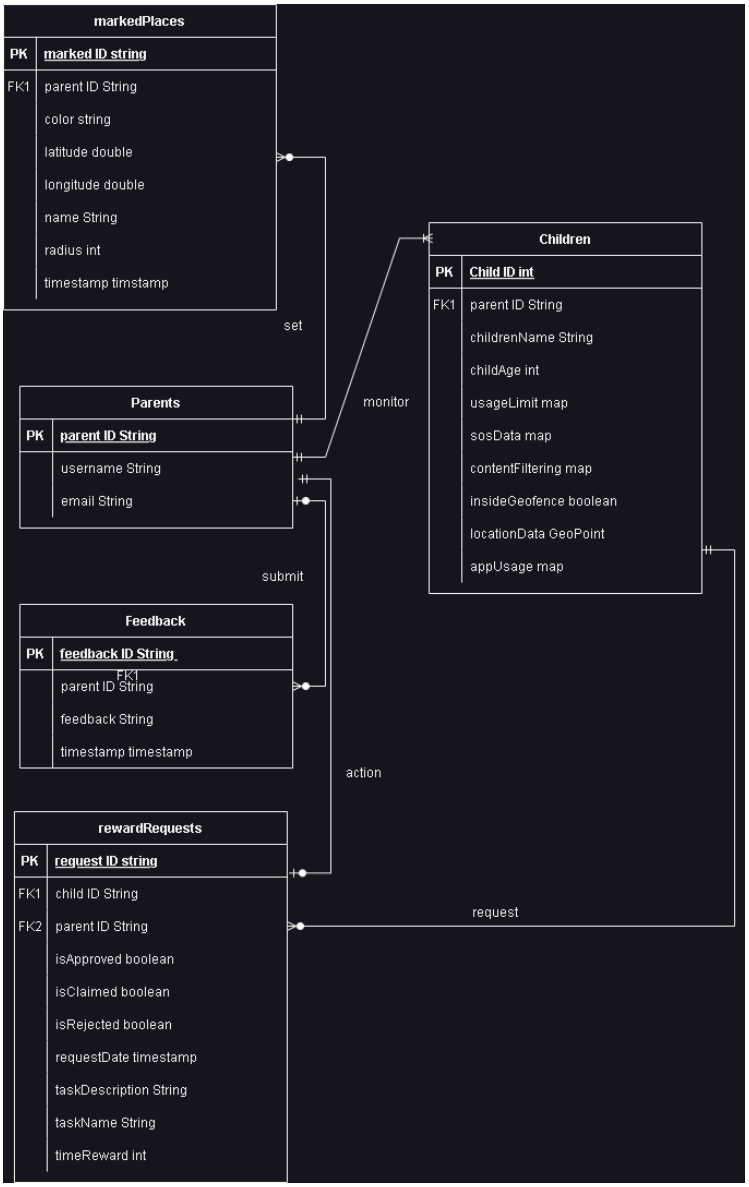


Figure 5.1: Entity Relationship Diagram

The ERD (Entity Relationship Diagram) shown in the Figure 5.1 represents the relationships between parents, children, reward requests, marked places and feedback in the application. These relationships show that how parents monitor their children, send the feedback, set the marked places and action the reward request, and how children request for the rewards.

For the relationship of parents and children shows that a parent can have more children as it forms a One-to-Many relationship. This is mandatory on both site, which means that a parent must have at least one child and a child must be linked to a parent to exist. Each parent is uniquely identified by the parentID, while each child in a parent's subcollection is identified by the childID. Thus, a parent should monitor at least one child.

Besides, the relationship between parent and marked places forms an Optional One-to-Many relationships. This is because a parent can set zero or many marked places, but it is not necessary to mark any place. A parent can also have multiple marked places, such as parent set school and playground for tracking their children movement and store the marked place as subcollection under parent with unique marked ID. In short, a parent can set zero or many places.

Moreover, One-to-Many Optional relationships is forms for parent and feedback relationships. Not all parents need to give feedback, but every parent can give more than one feedback entry. Feedback is recognized through its feedbackID and is connected to the parent through the parentID of the parent. Feedback entries have timestamps and include details such as the feedback's title and the time it was generated.

Next, a parent may receive multiple reward requests from their children, which it has form a One-to-Many relationships between them. The relationship means that each reward request is associated with a parent, but a parent can receive zero or many requests, so that it is an Optional relationship for parent. The unique rewardID is used to identify the reward request, which is connected to the parentID. This ensures that each request for a reward is action by the parent.

5.1.2 Class Diagram

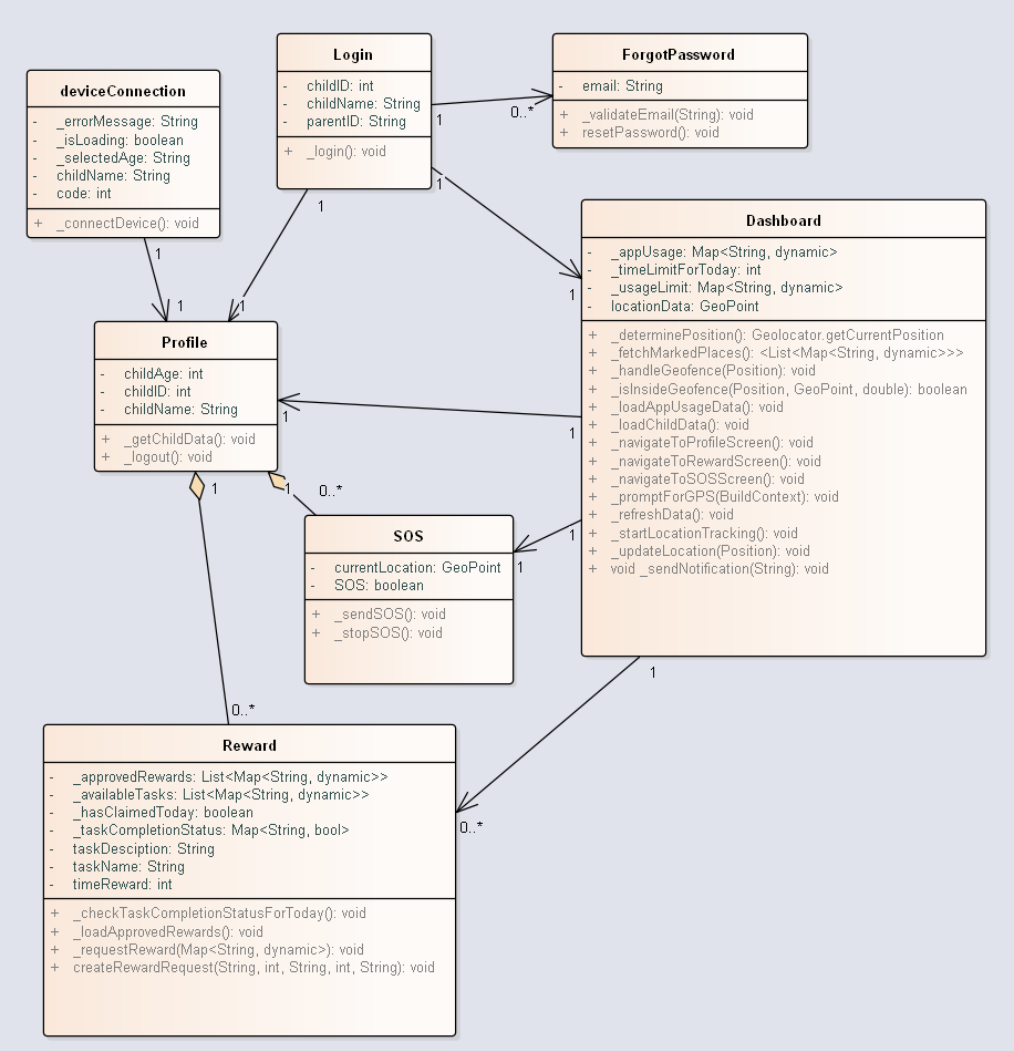


Figure 5.2: Children Class Diagram

In the child's class diagram shown in Figure 5.2, the relationship between each class based on how they interact is defined.

Table 5.1: Child's Class Diagram Description

Relationship	Multiplicity	Explanation
Login -> Profile	1 -> 1	Each login is associated with one user, each profile can only be accessed by one user.

Login -> ForgetPassword	1 -> 0..*	Each login can have zero or more forget password, and each forgetPasword process is associated with one login.
Login -> Dashboard	1 -> 1	Each login will show the dashboard, and the dashboard is accessed by one login.
Dashboard -> SOS	1 -> 1	A dashboard provide access for one SOS feature, and each SOS feature is accessible by one dashboard.
Dashboard -> Rewards	1 -> 0..*	A dashboard provides multiple reward options, but each rewards is link to dashboard.
Dashboard -> Profile	1 -> 1	A dashboard provide access for one user Profile feature, and each profile feature is accessible by one dashboard.
SOS -> Profile	0..* -> 1	The profile stores the SOS data. Zero or more SOS is sent or stop from one user, and each profile track zero or more SOS data.
Rewards -> Profile	1 -> 0..*	The profile stores the rewards data. Zero or more requests are request from one user, and each profile track zero or more rewards data.
deviceConnection	1 -> 1	Each connection is associated with one user, each profile can only be accessed by one connection.

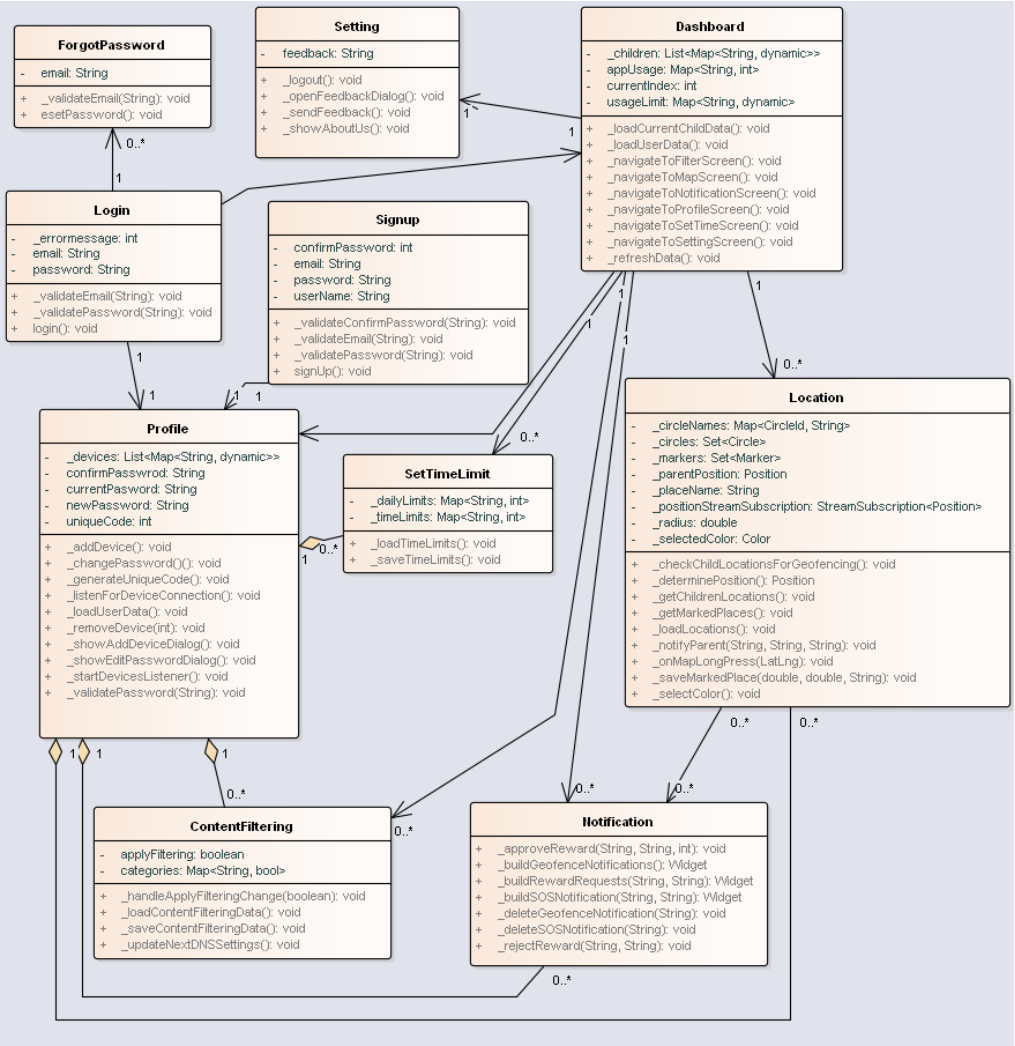


Figure 5.3: Parent Class Diagram

In the parent's class diagram shown in Figure 5.3, the relationship between each class based on how they interact is defined.

Table 5.2: Parent's Class Diagram Description

Relationship	Multiplicity	Explanation
Login -> Profile	1 -> 1	Each login is associated with one user, each profile can only be accessed by one user.
Login -> ForgetPassword	1 -> 0..*	Each login can have zero or more forget password, and each forgetPasword process is associated with one login.

Login -> Dashboard	1 -> 1	Each login will show the dashboard, and the dashboard is accessed by one login.
SignUp -> Profile	1 -> 1	Each signup creates one user profile, and each user profile is created from one signup.
Profile -> Contentfiltering	1 -> 0..*	The profile holds filtering options, but filtering can exist independently. A profile can have zero or more filtering content options but each is associated with one user profile.
Profile -> SetTimeLimit	1 -> 0..*	The profile stores time limit settings but exists independently. A profile can have zero or more time limits but each is associated with one user profile.
Profile -> Location	1 -> 0..*	Location data is stored under the profile but exists independently. A profile can track zero or more locations, but each is associated with one user profile.
Profile -> Notification	1 -> 0..*	Notification data is stored under the profile but exists independently. A profile can view zero or more notifications, but each is associated with one user profile.
Dashboard -> Contentfiltering	1 -> 0..*	A dashboard can display zero or more filtering content options but each is associated with one user profile.
Dashboard -> SetTimeLimit	1 -> 0..*	A dashboard can display zero or more daily time limit but each is associated with one user profile.
Dashboard -> Location	1 -> 0..*	A dashboard can display zero or more locations but each is associated with one user profile.

Dashboard -> Notification	1 -> 0..*	A dashboard can display zero or more notifications but each is associated with one user profile.
Dashboard -> Setting	1 -> 0..*	A dashboard provides access to settings, and each settings access by one dashboard.

5.1.3 Interface Flow Diagram

The interface flow diagram shown in Figure 5.4 represents the navigation of parents and children using the application with the features.

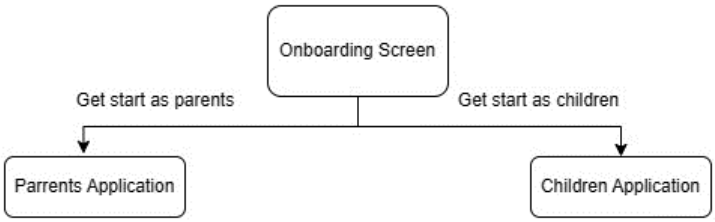


Figure 5.4: Interface Flow Diagram of Access the Application.

First, the users will access to the Onboarding Screen and select as parent or children. If the parents' application is chosen, it will navigate to the parent's application which the interface flow diagram shown in Figure 5.5, whereas, if choose as children, it will navigate to the children's application which the interface flow diagram shown in Figure 5.6

5.1.3.1 Parents Application

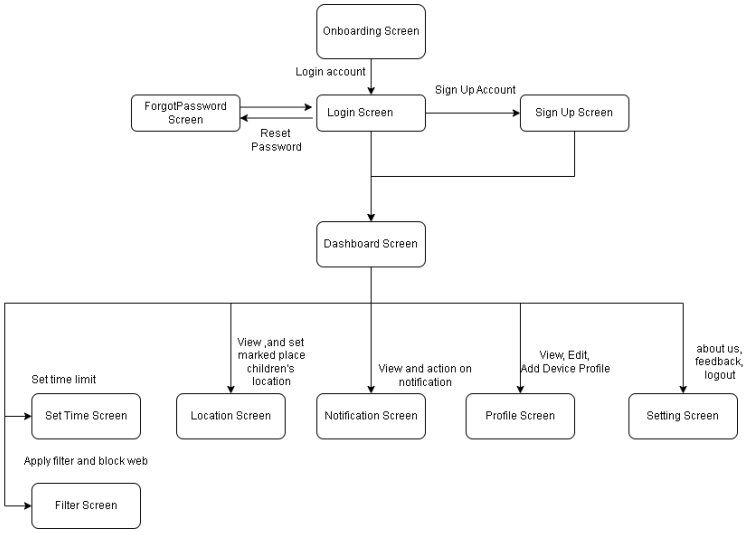


Figure 5.5: Interface Flow Diagram of Parents Application.

After the users select “Get Started as Parents”, it will display the Login Screen for registered parents to log into the application. If not, parents can sign up by navigating to Signup Screen. After signing up, it will route to the dashboard (Home Screen). However, if parents forgot their password, they could also reset the pass if click on forget password because the application will navigate to Forgot Password Screen.

The Dashboard Screen provide the navigation bar for parents to navigate to the specific screen. There are Location Screen, Notification Screen, and Profile Screen to navigate from the navigation bar for the specific feature. However, parent can also view the summary online activities of the children at the Dashboard Screen with the child’s today time limit and app usage. From Dashboard Screen, parents are allowed to navigate to the Set Time Limit Screen to set and update the daily time limit and navigate to the Content Filtering Screen to apply blocking. When the parents route to the Setting Screen, they can view the about us, provide the feedback and logout from the application.

4.1.3.2 Children Application

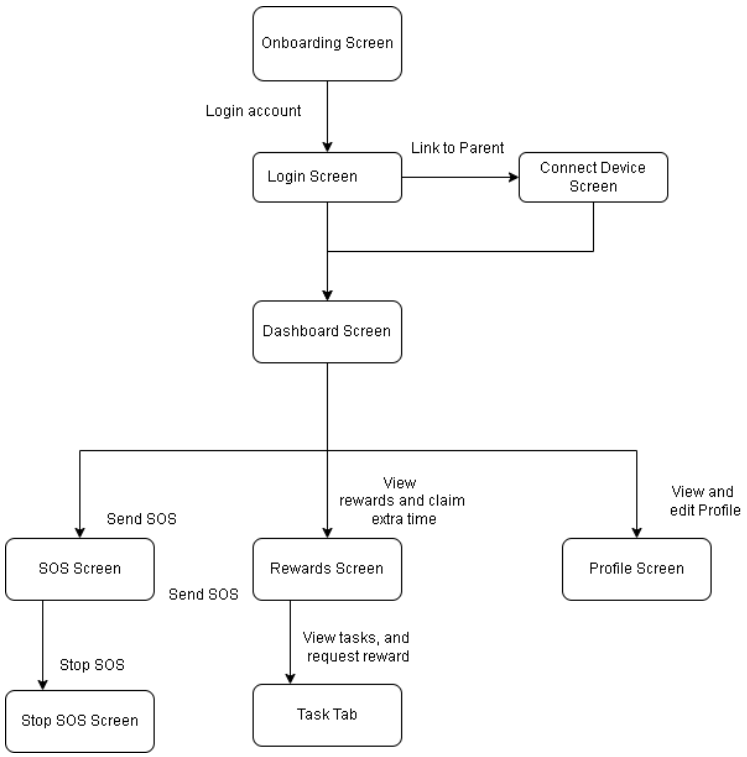


Figure 5.6: Interface Flow Diagram of Children Application.

When the users select as children, they will access to the children's application. Children are also need to login their account at the Login Screen. If the children have not yet to link to parent, they can access to the Connect Device Screen to setup the connection with parent with the unique code generated by the parents. After that, children will on the Dashboard Screen. The Dashboard Screen will show the today time limit and the app usage that used by today. Besides, the navigation bar at bottom of the screen gives the access to other screens. For example, click on the "SOS", the application will bring the children to the SOS screen as they can send and stop the SOS. Children can also will the rewards at the Reward Screen and profile from Profile Screen.

5.2 Connection Setup

5.2.1 Software Setup

5.2.1.1 Setup Flutter

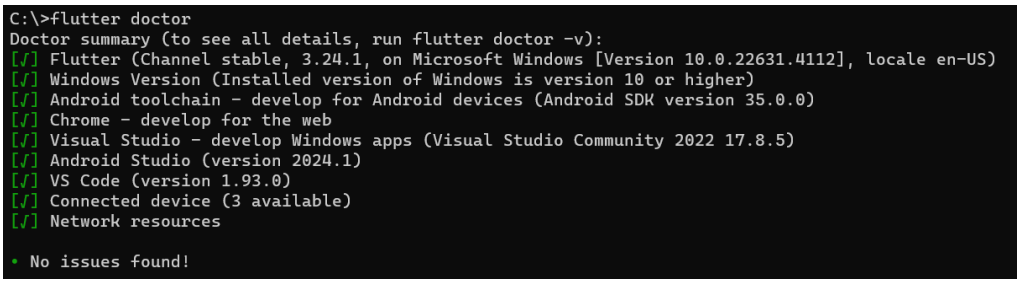


Figure 5.7: Setup Flutter

Flutter is the framework selected to implement the application using Dart Language. Setup the Flutter to confirm the connection with Visual Studio, Android Studio and connected device. Figure 5.7 shown the setup of Flutter which all the resources are connected.

5.2.1.2 Setup Android Studio

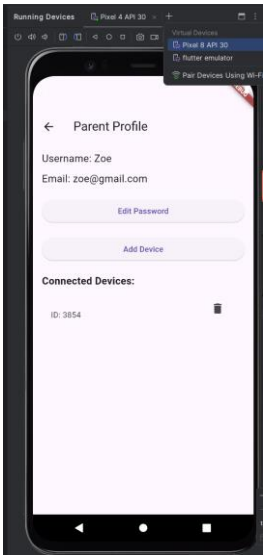


Figure 5.8: Setup Android Studio

Android Studio is the tools that provide the emulator for show the output. After setup, the emulator can run at the Visual Studio. Figure 5.8 shown the emulator to debugging the system.

5.2.1.3 Setup Visual Studio

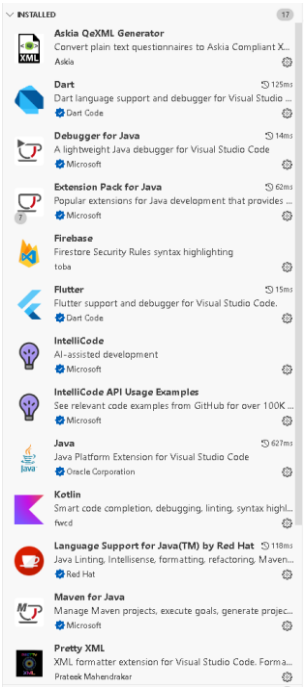


Figure 5.9: Setup Visual Studio

Visual Studio Code is the IDE that used in this project. Therefore, the extensions need to install to support the service as shown in Figure 5.9. For example, Flutter and Dart is the programming language used in the project and the Firebase is the Back-end that store the data.

5.2.2 Third-party Setup

5.2.2.1 Setup the Firebase

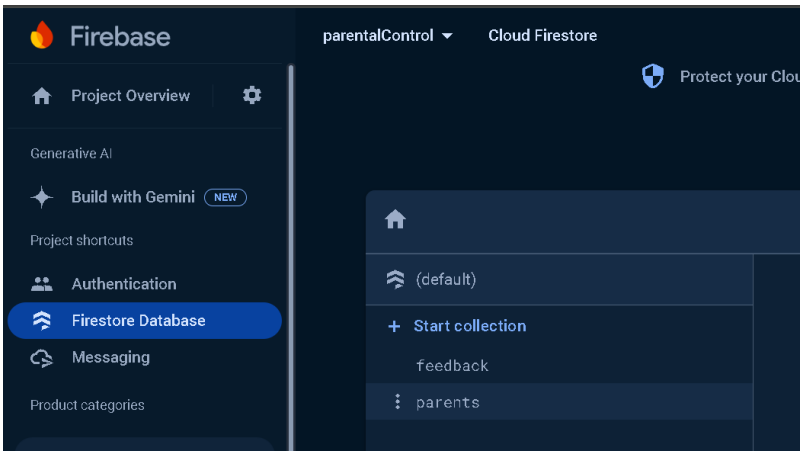


Figure 5.10: Setup Firebase

```
C:\Users\Admin\parental_control>firebase projects:list
✓ Preparing the list of your Firebase projects
```

Project Display Name	Project ID	Project Number	Resource Location ID
flutter ParentalControl	flutter-parentalcontrol	575968657544	[Not specified]
parentalControl	**parentalcontrol-5db2a (current)**	416000587844	[Not specified]
parentalControl	parentalcontrol-965da	670900468079	[Not specified]
remoteParentalControl	remoteparentalcontrol	806391087396	[Not specified]
remoteParentalControl	remoteparentalcontrol-3fb74	88944840767	[Not specified]
test	test-3dd69	966573835110	[Not specified]
testing	testing-c08f0	257225474322	[Not specified]

```
7 project(s) total.
```

Figure 5.11: Setup Firebase in Visual Studio

Firebase is used as the back-end which help to verify the authentication and store the data. Next, the Firebase need to set up for further use with command ‘flutterfire configure’. Therefore, the successfully connected, help in use for storing in further function. The Figure 5.10 and Figure 5.11 shown that the Firebase is settled up an connected to the system.

5.2.2.2 Setup NextDNS

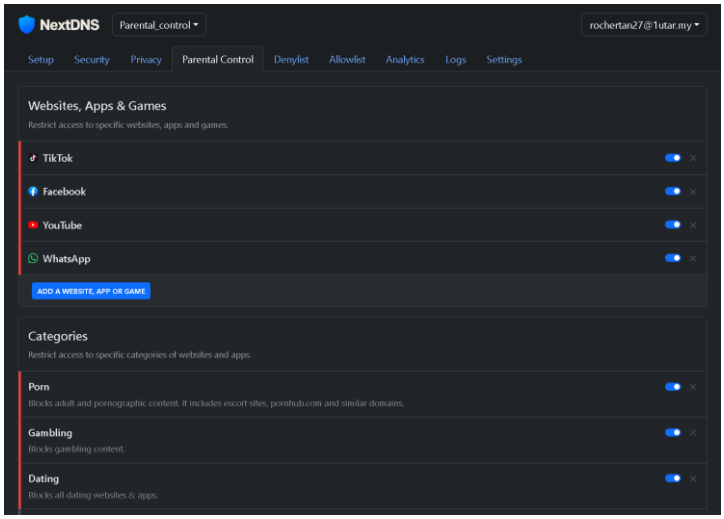


Figure 5.12: Setup NextDNS

Figure 5.12 shown the setup of NextDNS, which is the server that provide the content filtering feature. It allows parents to control and restrict the categories of websites and online content accessible from their child's device. Filtering rules can be customized for categories like porn, gaming, social media, and dating after setting up NextDNS. Typically, setting up involves making a NextDNS account, setting up a specific DNS profile and it will provide an API to integrate with.

5.2.2.3 Setup Google Cloud

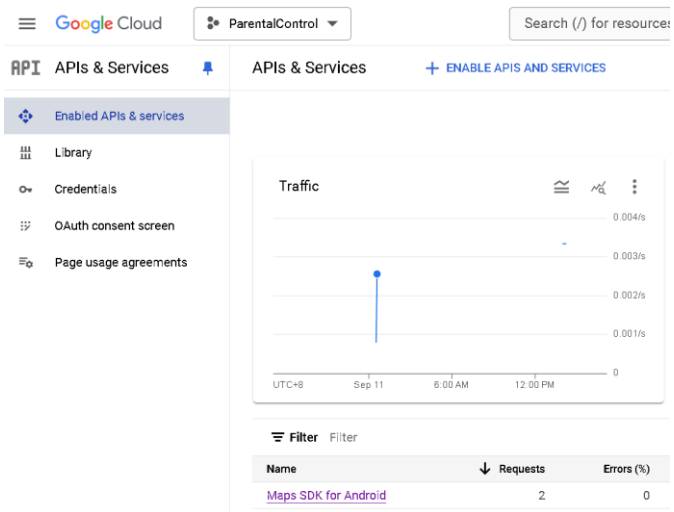


Figure 5.13: Setup Google Cloud

Figure 5.12 shown the setup Google Cloud, which is used to enable the Maps SDK for Android, allowing the integration of Google Maps into the application. By enabling this service, the maps, markers, geolocation features, and geofencing can display to interactive within the application. In order to configure the Maps SDK, creating an project, enabling the Maps for Android API and generate the API key. The API key is then added to the configuration file named “AndroidManifest.xml”.

5.3 Screenshot and Code Snippet of Application

5.3.1 Parents’ application

5. 3.1.1 Onboarding Screen

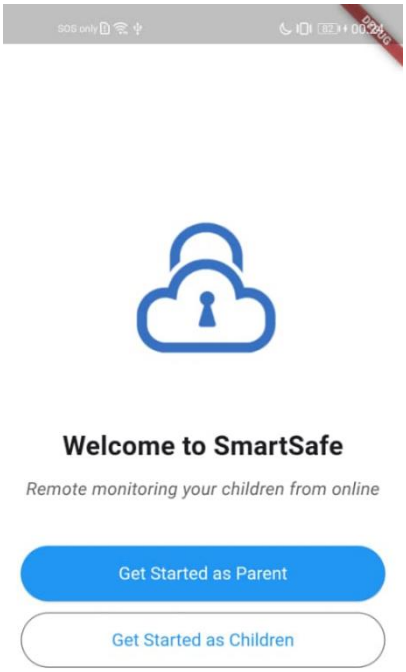


Figure 5.14: Onboarding Screen

```
ElevatedButton(
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ParentLoginScreen()),
    );
  },
  child: Text('Get Started as Parent'),
  style: ElevatedButton.styleFrom(
    foregroundColor: Colors.white, backgroundColor: Colors.blue,
    minimumSize: Size(double.infinity, 50),
    textStyle: TextStyle(fontSize: 16),
  ),
), // ElevatedButton
SizedBox(height: 10),
OutlinedButton(
  onPressed: () {
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => ChildLoginScreen()),

    );
  },
  child: Text('Get Started as Children'),
  style: OutlinedButton.styleFrom(
    foregroundColor: Colors.blue, minimumSize: Size(double.infinity, 50),
    textStyle: TextStyle(fontSize: 16),
  ),
), // OutlinedButton
```

Figure 5.15: Code of Onboarding Screen

This is the onboarding screen which users can select as parent or children as user. It will navigate to parent login screen when pressing “Get Started as Parent”, while pressing “Get Started as Children”, it will navigate to children login screen. The code provided in Figure 5.2 demonstrates how buttons are created and how *Navigator.push* is used to route to the appropriate screen when each button is pressed. Therefore, parents need to access as parent.

5.3.1.2 Parent Login Screen

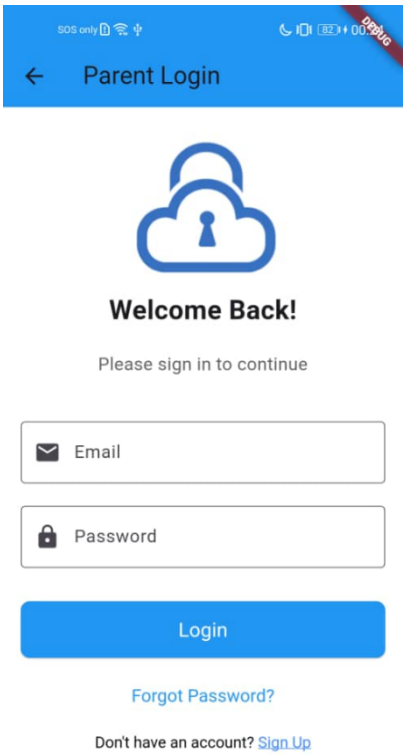


Figure 5.16: Parent Login Screen

```
void login(BuildContext context) async {
  setState(() {
    _emailError = null;
    _passwordError = null;
  });

  if (_formKey.currentState!.validate()) {
    try {
      // Attempt to sign in with the provided email and password
      UserCredential userCredential = await FirebaseAuth.instance.signInWithEmailAndPassword(
        email: emailController.text,
        password: passwordController.text,
      );

      // Navigate to the dashboard screen when successful login
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => ParentMainScreen()),
      );
    } on FirebaseAuthException catch (e) {
      setState(() {
        if (e.code == 'user-not-found') {
          _emailError = 'User is not registered yet.';
        } else if (e.code == 'wrong-password') {
          _passwordError = 'Please enter a correct email or password.';
        } else {
          _emailError = 'Please enter a correct email or password.';
        }
      });
    }
  }
}
```

Figure 5.17: Code of Parent Login Screen

In Parent Login Screen, parents are required to enter their email address and password as registered. After the authentication, parents are successfully login and navigate to the dashboard screen. As the code shows the function handle the login process of the parents by using Firebase Authentication. Firstly, it will check if the form validation passes, ensuring that the input data meets the required format using *_formKey* as using a valid email and password. When the form is valid, the application attempts to sign in using the user's email and password through the Firebase's *signInWithEmailAndPassword* method. If the login failure, the error message will display. For example, when the parents are not registered, it will show the message "user-not-found". "Please enter a correct email and password." will show when the email or password or both are not correct. It helps parents to handle easily.

However, if parents forgot their password, they could reset their password by clicking the link "Forgot Password?", it will bring them to the reset password. Whereas, when parents have not registered an account, they could Sign Up by press on the "Sign Up".

5.3.1.3 Parent Sign Up Screen

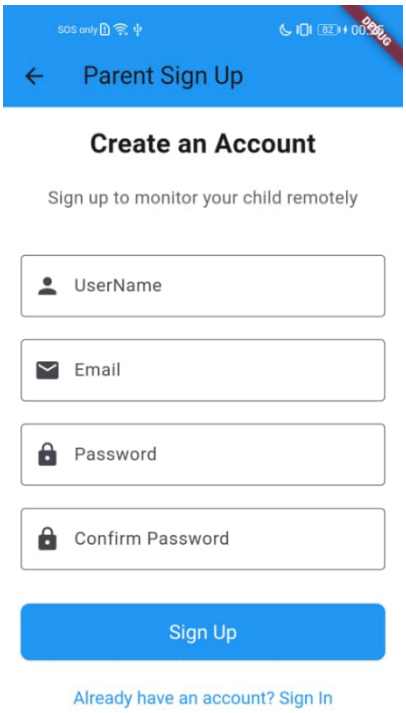


Figure 5.18: Parent Sign Up Screen

```
void signUp() async {
  if (_formKey.currentState!.validate()) {
    if (passwordController.text != confirmPasswordController.text) {
      setState(() {
        _confirmPasswordError = 'Passwords do not match';
      });
      return;
    }

    try {
      // Create user with Firebase Authentication
      UserCredential userCredential = await FirebaseAuth.instance.createUserWithEmailAndPassword(
        email: emailController.text,
        password: passwordController.text,
      );

      // Store user data in Firestore
      await FirebaseFirestore.instance.collection('parents').doc(userCredential.user?.uid).set({
        'username': userNameController.text,
        'email': emailController.text,
      });

      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text('Account created for ${userCredential.user?.email}')),
      );

      // Navigate to the dashboard screen after successful sign-up
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => ParentMainScreen()),
      );
    } on FirebaseAuthException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text('Failed to sign up: ${e.message}')),
      );
    }
  }
}
```

Figure 5.19: Code of Parent Sign Up Screen

If the parent is a new user and wants to register a new account, they can create one by navigating to the Sign Up screen. On the screen, they will be prompted for details as they need to enter their username, email, and password. Once the required fields are completed, the application will validate the input and use Firebase Authentication to create a new account. However, if the confirm password is not match with the password entered, an error message will display. The Firestore database stores the parent's username and email under 'parent' collection, with a random generated user ID as the document ID. After sign up successfully, the application is navigated to the Parent Dashboard screen.

5.3.1.4 Parent Forget Password Screen

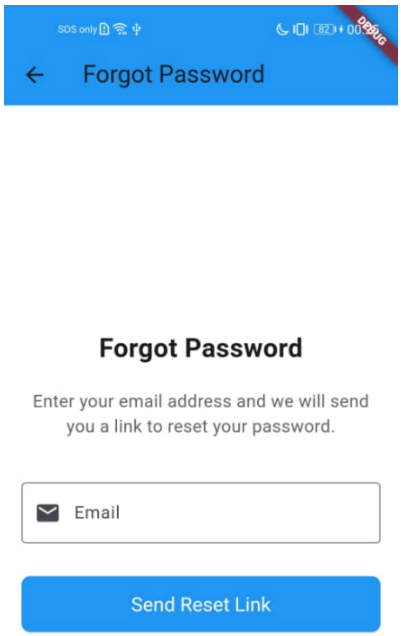


Figure 5.20: Forgot Password Screen

```
void resetPassword(BuildContext context) async {
  if (_emailError == null && _formKey.currentState!.validate()) {
    try {
      await FirebaseAuth.instance.sendPasswordResetEmail(email: emailController.text);
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text('Password reset email sent! Please check your inbox.')),
      );
      Navigator.pop(context); // Go back to login screen after sending the email
    } on FirebaseAuthException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text('Error: ${e.message}')),
      );
    }
  }
}
```

Figure 5.21: Code of Forgot Password Screen

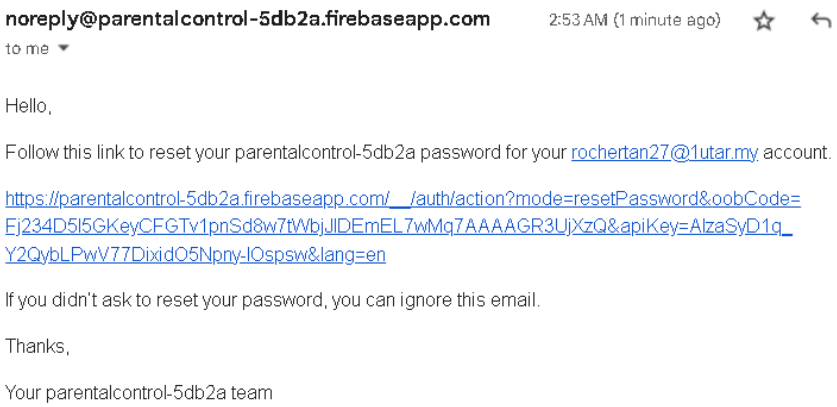


Figure 5.22: Reset Password

Even in login or signup screen, parent can reset their password if they do not remember it by pressing the “Forgot Password?” link. When this link is pressed, the application will run the *resetPassword* function. The code validates the inputted email using the form's state. If the email is valid, it sends a password reset link via email using Firebase Authentication's *sendPasswordResetEmail* method. A confirmation message is displayed using a *SnackBar*, which informing the parent that the reset email has been sent. Therefore, parents can check their inbox to follow the link to reset the password.

5.3.1.5 Parent User Profile Screen

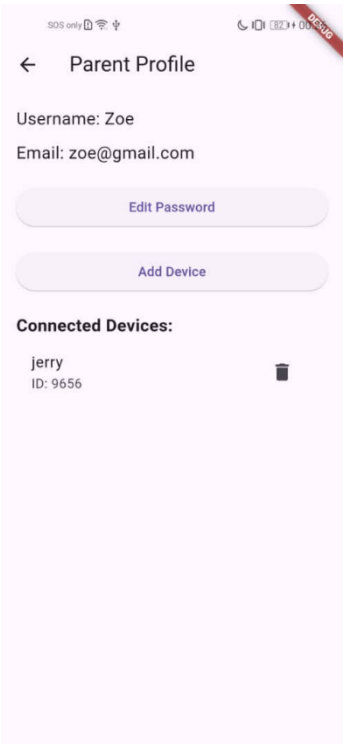


Figure 5.23: Parent User Profile Screen

When parents navigate to the user profile screen, the application will display their username, email address and connected devices. The data of the information is gained from the Firestore database.

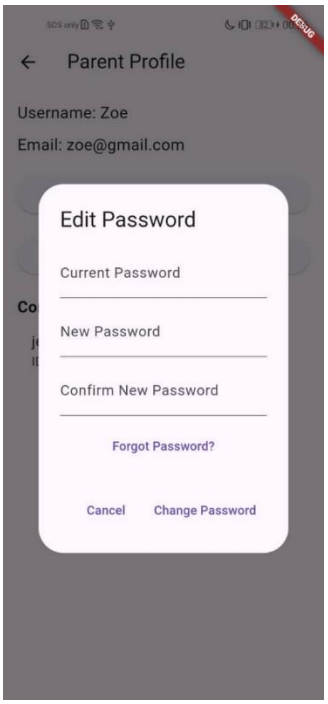


Figure 5.24: Parent Change Password Screen

Parents are allowed to change their password using their current password. They are required to enter the correct current password then only can successfully change. They can also click on the “Forgot Password” to get the reset link.

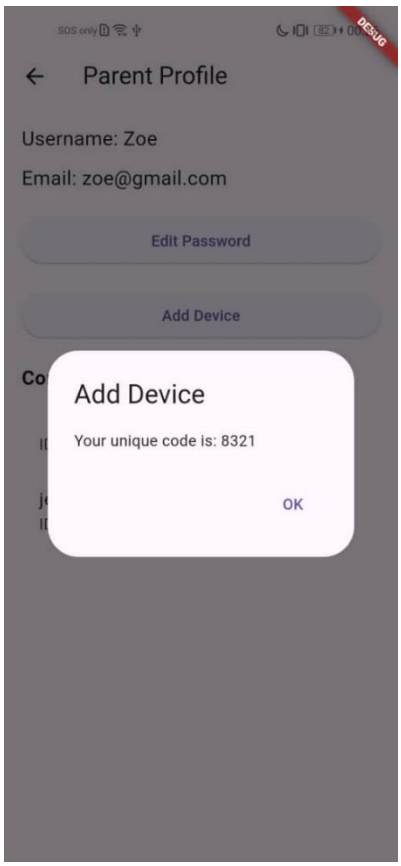


Figure 5.25: Parent Add Device Screen

When parents want to add a child, the application will generate a code and display to them. The unique code is use in children connect device as their child ID. As the implemented code shown, the *_addDevice* function is an asynchronous method used to add a new child device to a parent's account in a Firestore database. Then, it creates a new collection name ‘children’ with the generated code as document ID under parent. The stored data includes default values for the child's name, age, daily usage limits, content filtering settings, and location data for further features.

5.3.1.6 Parent Dashboard Screen

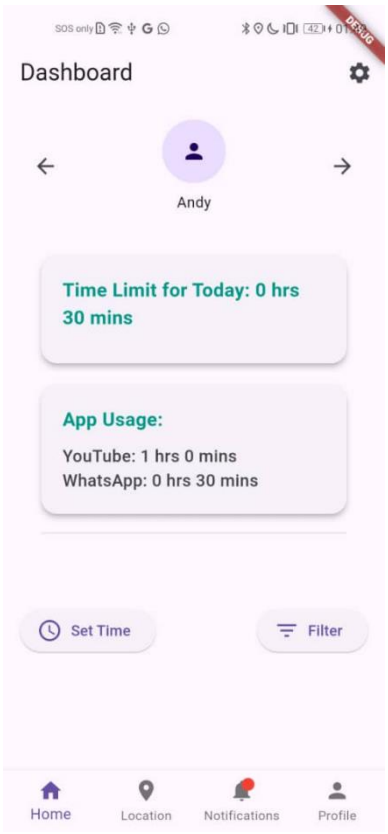


Figure 5.26: Parent Dashboard Screen

Parent Dashboard Screen is a main screen after logged in. Parents can select the connected child at the top that shown the child's name. Press left and right to switch to other children. Then, for each selected child, their name, time limit and app usage will show as the data retrieve from database. If parents click on the "Set Time" button, it will navigate to Set Time Screen for set the daily time for the selected child. Same as well for the "Filter" button, the screen will navigate to Content Filtering Screen as also set for the selected child. The navigator bar is provided at the bottom which let the parents to navigate screen to the specific screen. As the "Home" button represent the dashboard, "Location" will route to Location Tracking Screen and "Profile" is for parent to view and edit their profile. The "Setting" icon is also provided for navigation to Setting Screen which able the parent to logout from the application.

5.3.1.7 Parent Set Time Limit Screen

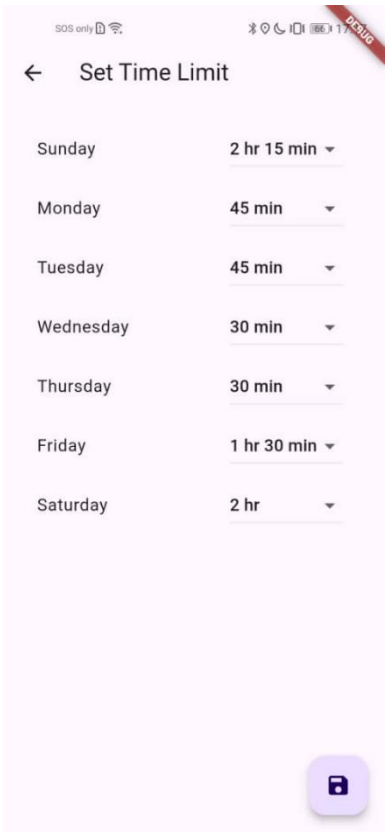


Figure 5.27: Parent Set Time Limit Screen

Parents are able to set the daily screen time for children by navigating to the Set Time Screen from the Dashboard Screen of the “Set Time” button. They are allowed to schedule the daily screen time with the drop list of each day. After saving, the data will update to the Firestore database. Parents are also can reset the time by update again to the application.

```
Future<void> _saveTimeLimits() async {
if (_auth.currentUser == null || widget.childId == null) return;

await _firestore
    .collection('parents')
    .doc(_auth.currentUser!.uid)
    .collection('children')
    .doc(widget.childId)
    .set(
  {
    'usageLimit': {
      'dailyUsageLimits': _dailyLimits,
      'lastUsageUpdate': Timestamp.now(), // Current timestamp
    },
  },
  SetOptions(merge: true),
);
}
```

Figure 5.28: Parent Set Time Limit Screen

From the figure above, the *_saveTimeLimits* is implemented to save and update the daily time limit for children in Firestore. This code snippet simply show that after user is authenticated, the data of the list of 'dailyUsageLimits' will update to the Firestore under the 'usageLimit' field in the children collection.

5.3.1.8 Parent Content Filtering Screen

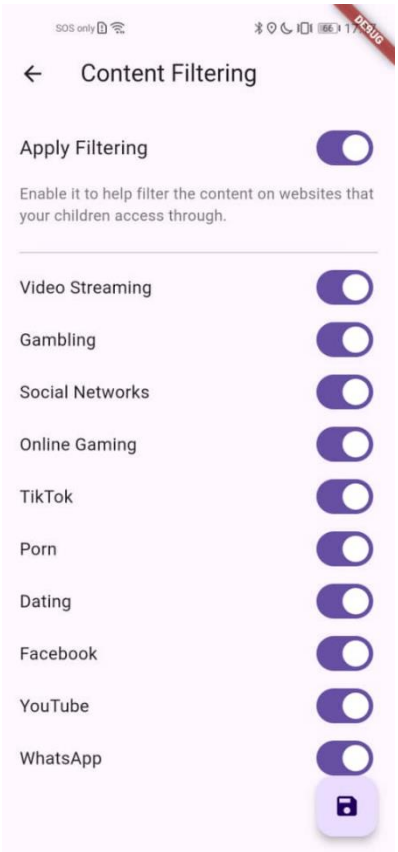


Figure 5.29: Parent Content Filtering Screen

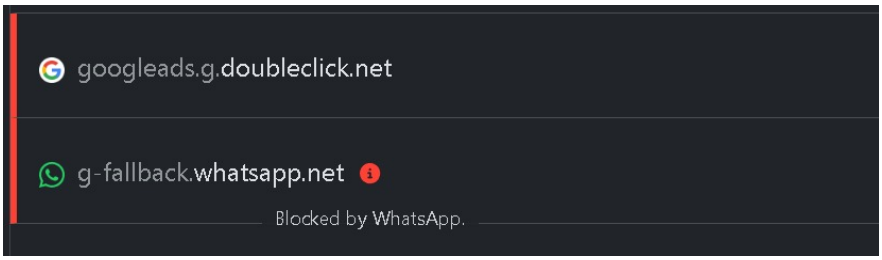


Figure 5.30: Log of the Blocked Content

In this Content Filtering Screen, parent can view the categories of the content and block the access. Turn the switch of the 'Apply Filtering' will enable all the categories of the content. For example, when turn the switch of the WhatsApp, the WhatsApp application will be blocked by the NextDNS server as shown in the figure 5.23.

```
Future<void> _updateNextDNSSettings() async {
final profileId = '362d98';
final url = Uri.parse('https://api.nextdns.io/profiles/$profileId/parentalcontrol');
final headers = {
  'x-api-key': '318f29dd5d4cd5c24f380abb7a7d86f4ccef6e48',
  'Content-Type': 'application/json',
};

final body = jsonEncode({
  'safeSearch': false,
  'youtubeRestrictedMode': false,
  'blockBypass': false, // Block bypass if filtering is applied
  'services': [
    {'id': 'tiktok', 'active': categories['TikTok'] ?? false},
    {'id': 'facebook', 'active': categories['Facebook'] ?? false},
    {'id': 'youtube', 'active': categories['YouTube'] ?? false},
    {'id': 'whatsapp', 'active': categories['WhatsApp'] ?? false},
  ],
  'categories': [
    {'id': 'porn', 'active': categories['Porn'] ?? false},
    {'id': 'social-networks', 'active': categories['Social Networks'] ?? false},
    {'id': 'gambling', 'active': categories['Gambling'] ?? false},
    {'id': 'gaming', 'active': categories['Online Gaming'] ?? false},
    {'id': 'video-streaming', 'active': categories['Video Streaming'] ?? false},
    {'id': 'dating', 'active': categories['Dating'] ?? false},
  ],
});

try {
  final response = await http.patch(url, headers: headers, body: body);

  if (response.statusCode == 200) {
    print('NextDNS settings updated successfully.');
  } else {
    print('Failed to update NextDNS settings: ${response.statusCode}');
    print('Response body: ${response.body}');
  }
} catch (e) {
  print('Error updating NextDNS settings: $e');
}
}
```

Figure 5.31: Parent Content Filtering Feature

This code snippet describes the content filtering integrate with NextDNS server via an API request to control over the content filtering feature. In order to integrate with NextDNS, a document is provided on guiding to integrate. First, the server needs to be set up as in chapter 5.2.2.2 with setting the parental control setting. Then, it constructs the URL for the API endpoint using a predefined profileId (362d98). The HTTP headers are set including the API key for authentication and content type of ‘application/json’. The function builds the request body by forming a JSON object that includes different parental control options like categories and services. The function sends an HTTP PATCH request to update these settings on the NextDNS server. If the request is successful, it will update on the server.

5.3.1.9 Parent Location Tracking Screen

Figure 5.32: Parent Location Tracking Screen

Parents can view their children's real-time location by navigating to Location Tracking Screen while open the Location Service with GPS. At the screen, parents are able to see their children location, and marked place with the colour. When parents click on the children location, the detail of the location will display with the latitude and longitude. The 'my location' icon on the right top corner, can return back to the to the parent's current location.

```
Future<void> _getChildrenLocations() async {
  User? currentUser = _auth.currentUser;
  if (currentUser == null) {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('No user is logged in')),
    );
    return;
  }

  QuerySnapshot childrenSnapshot = await _firestore
      .collection('parents')
      .doc(currentUser.uid)
      .collection('children')
      .get();

  setState(() {
    _markers.addAll(
      childrenSnapshot.docs.map((doc) {
        var data = doc.data() as Map<String, dynamic>;
        var locationData = data['locationData'];
        if (locationData is GeoPoint) {
          return Marker(
            markerId: MarkerId(doc.id),
            position: LatLng(locationData.latitude, locationData.longitude),
            infoWindow: InfoWindow(
              title: data['childrenName'],
              snippet: 'Tap for more info',
            ), // InfoWindow
            onTap: () {
              _showLocationDetails(
                doc.id,
                data['childrenName'],
                locationData.latitude,
                locationData.longitude,
              );
            },
          ); // Marker
        } else {
          print('Location data is not a GeoPoint');
          return null;
        }
      }).whereType<Marker>(),
    );
  });
}
```

Figure 5.33: Code Get Child Location

The *_getChildLocations* function retrieves and shows the real-time location of every child connected to the parent who is current logged in. Firstly, the user authentication will need to be verify. If the parent is authenticated, the function retrieves the list of children linked to the parent's account by searching the Firestore database within the parent's document. It goes into the children subcollection, fetching the document of each child. The function gets the

locationData field for each child document, which should contain a GeoPoint that indicate the latitude and longitude of the child. If the locationData is valid, the children's location is display on the map with creating a marker which include the children's name. When parents tap on the marker, it triggers the *_showLocationDetails* function which provide the details about the child's location.

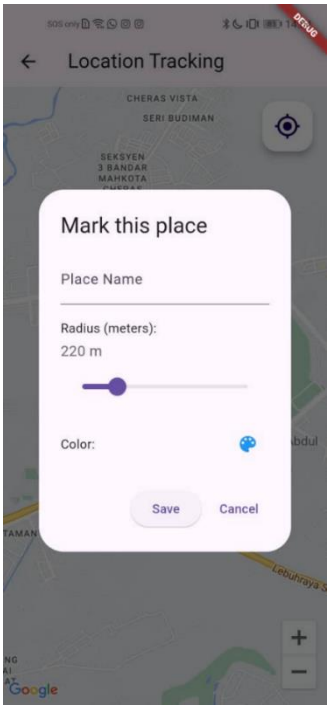


Figure 5.34: Parent Mark a Place Feature

```
Future<void> _saveMarkedPlace(double lat, double lng, [String? id]) async {
  User? currentUser = _auth.currentUser;
  if (currentUser == null) {
    return;
  }

  try {
    DocumentReference placeRef;
    if (id == null) {
      // Create new document if id is null
      placeRef = _firestore
          .collection('parents')
          .doc(currentUser.uid)
          .collection('markedPlaces')
          .doc(); // Creates a new document with a unique ID
    } else {
      // Update existing document
      placeRef = _firestore
          .collection('parents')
          .doc(currentUser.uid)
          .collection('markedPlaces')
          .doc(id);
    }

    Map<String, dynamic> placeData = {
      'name': _placeNameController.text,
      'latitude': lat,
      'longitude': lng,
      'radius': _radius,
      'color': _selectedColor.value.toString(),
      'timestamp': FieldValue.serverTimestamp(),
    };

    await placeRef.set(placeData);

    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text(id == null ? 'Place saved successfully!' : 'Place updated successfully!')),
    );

    setState(() {
      _circles.removeWhere((circle) => circle.circleId.value == placeRef.id);
      _circles.add(
        Circle(
          circleId: CircleId(placeRef.id),
          //name:_placeNameController.text,
          center: LatLng(lat, lng),
          radius: _radius,
          fillColor: _selectedColor.withOpacity(0.2),
          strokeColor: _selectedColor,
          strokeWidth: 2,

        ), // Circle
      );
    });
  } catch (e) {
    print('Error saving marked place: $e');
  }
}
```

Figure 5.35: Parent Mark a Place Function

When parents long press on the map, they are allowed to set a place as marked place with geofencing function. While long press, the dialog has appeared for parents to set the place name, place size with radius and colour. The “Save” button trigger the *_saveMarkedPlace* function which allow the parents to save and update the marked place on the map, storing its details in the Firestore database displaying it visually on the map with a circle. Firstly, the user authentication needs to be verified, then check if the marked ID exists. If not (ID is null), the function will take it as a new place and create a new document in the markedPlaces subcollection of the current parent's document in the Firestore. The map named placeData stores information about the location, including name, latitude, longitude, radius, colour, and timestamp. The data is stored in Firestore using the *set()* method, which either creates a new document or updates an existing one. This function allows users to create a new marked place.

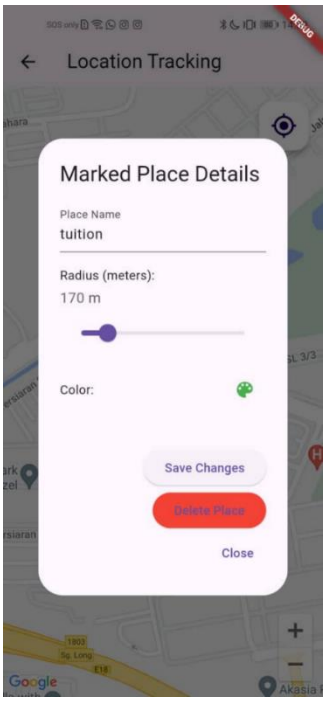


Figure 5.36: Parent View, Edit and Delete Marked Place Feature

```
Future<void> _getMarkedPlaces() async {
  User? currentUser = _auth.currentUser;
  if (currentUser == null) {
    return;
  }

  try {
    QuerySnapshot markedPlacesSnapshot = await _firestore
        .collection('parents')
        .doc(currentUser.uid)
        .collection('markedPlaces')
        .get();

    setState(() {
      _circles.addAll(
        markedPlacesSnapshot.docs.map((doc) {
          var data = doc.data() as Map<String, dynamic>;
          double lat = data['latitude'];
          double lng = data['longitude'];
          double radius = data['radius'];
          Color color = Color(int.parse(data['color']));
          String placeName = data['name'];

          CircleId circleId = CircleId(doc.id);
          _circleNames[circleId] = placeName;

          return Circle(
            circleId: CircleId(doc.id),
            //name: placeName,
            center: LatLng(lat, lng),
            radius: radius,
            fillColor: color.withOpacity(0.2),
            strokeColor: color,
            strokeWidth: 2,

          ); // Circle
        }),
      );
    });
  } catch (e) {
    print('Error loading marked places: $e');
  }
}
```

Figure 5.37: Parent View Marked Place Function

```
Future<void> _deleteMarkedPlace(String placeId) async {
  User? currentUser = _auth.currentUser;
  if (currentUser == null) {
    return;
  }

  try {
    await _firestore
        .collection('parents')
        .doc(currentUser.uid)
        .collection('markedPlaces')
        .doc(placeId)
        .delete();

    setState(() {
      _circles.removeWhere((circle) => circle.circleId.value == placeId);
      _circleNames.removeWhere((key, value) => key.value == placeId);
    });

    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Marked place deleted successfully!')),
    );
  } catch (e) {
    print('Error deleting marked place: $e');
  }
}
```

Figure 5.38: Parent Delete Marked Place Function

By on tap on the marked place, parents are allowed to view, update and delete the marked place. The dialog will appear and display the detail of the place with name, radius and colour, parents are also able to change it and save it or just delete the place.

Based on the figure 5.18, the *_getMarkedPlaces* function retrieves the logged-in user's (parent) marked places from database 'markedPlaces' in Firestore and shows them on the map as circles. The places that are retrieved are mapped to the Circle objects that represent areas on the map. It fetches the data of latitude, longitude, radius, colour, and name for every location from Firestore. Then, the data will display on the screen.

The save function is same as the create marked place in figure 5.16. Besides, the *_deleteMarkedPlace* function allows parents to delete a marked place by specifying its 'placeId'. After verifying that the user is logged in, it gets the 'markedPlaces' collection for the current user in Firestore and removes the document that matches the provided 'placeId'. After the place is deleted, the

function updates the map by removing the matching circle from '_circle' which use the 'placeID' to find it.

5.3.1.10 Parent Notification Screen

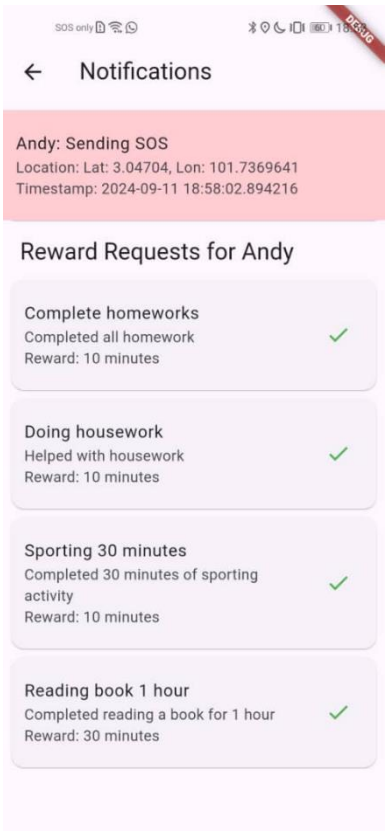


Figure 5.39: Parent Notification Screen

In the Notification Screen, the notifications of reward request, SOS, and geofencing will be displayed for parents. Parents can take action on it like scroll left to delete the notification or reject the request and approve for the request. It will also show the geofencing notification like if children leave a marked place, then the notification will send to Notification Screen to alert parents that the children is enter or leave the marked place.

```
Widget _buildSOSNotification(String childID, String childName) {
  return StreamBuilder<DocumentSnapshot>(
    stream: _firestore
        .collection('parents')
        .doc(widget.parentID)
        .collection('children')
        .doc(childID)
        .snapshots(),
    builder: (context, snapshot) {
      if (!snapshot.hasData) {
        return SizedBox.shrink(); // No data yet, return an empty widget
      }

      var childData = snapshot.data!.data() as Map<String, dynamic>?;

      if (childData == null) {
        return SizedBox.shrink(); // No data in the document
      }

      var sosData = childData['sosData'] as Map<String, dynamic>?;

      if (sosData == null) {
        return SizedBox.shrink(); // No SOS data in the document
      }

      bool isSOSActive = sosData['isSOSActive'] ?? false;
      Timestamp? sosTimestamp = sosData['sosTimestamp'] as Timestamp?;
      GeoPoint? sosLocation = sosData['sosLocation'] as GeoPoint?;

      String sosStatus = isSOSActive ? 'Sending SOS' : 'Stopped SOS';
      String locationText = sosLocation != null ? 'Lat: ${sosLocation.latitude}, Lon: ${sosLocation.longitude}' : 'No location data';
      String timestampText = sosTimestamp != null ? sosTimestamp.toDate().toLocal().toString() : '';

      return Dismissible(
        key: Key('sos-${childID}'),
        direction: DismissDirection.endToStart,
        onDismissed: (direction) {
          _deleteSOSNotification(childID);
        },
        background: Container(
          color: Colors.red,
          child: Align(
            alignment: Alignment.centerRight,
            child: Padding(
              padding: EdgeInsets.symmetric(horizontal: 20),
              child: Icon(Icons.delete, color: Colors.white),
            ), // Padding
          ), // Align
        ), // Container
        child: Card(
          margin: EdgeInsets.symmetric(vertical: 8),
          child: ListTile(
            contentPadding: EdgeInsets.all(12),
            title: Text('$childName: $sosStatus'),
            subtitle: Text('Location: $locationText\nTimestamp: $timestampText'),
            tileColor: isSOSActive ? Colors.red.shade100 : Colors.grey.shade100,
          ), // ListTile
        ), // Card
      ); // Dismissible
    },
  ); // StreamBuilder
}
```

Figure 5.40: Parent SOS Notification Feature

The *_buidldSOSNotification* function generates a widget that monitors real-time SOS alerts from a specific child using a *StreamBuilder* connected to Firestore. Then, it checks the Firestore document for the child's SOS data, and if it is available, it extracts and displays the SOS status, timestamp, and location. The SOS status is shown as either "Sending SOS" or "Stopped SOS", along with the child's latitude and longitude and if the SOS is active, the notification card is coloured in red.

```dart
Widget _buildRewardRequests(String childID, String childName) {
  return StreamBuilder<QuerySnapshot>(
    stream: _firestore
        .collection('parents')
        .doc(widget.parentID)
        .collection('children')
        .doc(childID)
        .collection('rewardRequests')
        .where('isApproved', isEqualTo: false)
        .snapshots(),
    builder: (context, snapshot) {
      if (!snapshot.hasData) {
        return Center(child: CircularProgressIndicator());
      }

      List<QueryDocumentSnapshot> requestDocs = snapshot.data!.docs;

      if (requestDocs.isEmpty) {
        return ListTile(
          title: Text(''),
        ); // ListTile
      }

      return Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Padding(
            padding: const EdgeInsets.symmetric(vertical: 8.0, horizontal: 16.0),
            child: Text(
              'Reward Requests for $childName',
              style: Theme.of(context).textTheme.titleLarge,
            ), // Text
          ), // Padding
          ...requestDocs.map((doc) {
            String requestID = doc.id;
            String taskName = doc['taskName'];
            String taskDescription = doc['taskDescription'] ?? '';
            int timeReward = doc['timeReward'];
```

```dart
            int timeReward = doc['timeReward'];

            return Dismissible(
              key: Key('reward-${requestID}'),
              direction: DismissDirection.endToStart,
              onDismissed: (direction) {
                _rejectReward(childID, requestID);
              },
              background: Container(
                color: Colors.red,
                child: Align(
                  alignment: Alignment.centerRight,
                  child: Padding(
                    padding: EdgeInsets.symmetric(horizontal: 20),
                    child: Icon(Icons.delete, color: Colors.white),
                  ), // Padding
                ), // Align
              ), // Container
              child: Card(
                margin: EdgeInsets.symmetric(vertical: 4, horizontal: 8),
                child: ListTile(
                  contentPadding: EdgeInsets.all(12),
                  title: Text(taskName),
                  subtitle: Text('$taskDescription\nReward: $timeReward minutes'),
                  trailing: IconButton(
                    icon: Icon(Icons.check, color: Colors.green),
                    onPressed: () => _approveReward(childID, requestID, timeReward),
                  ), // IconButton
                ), // ListTile
              ), // Card
            ); // Dismissible
          }).toList(),
        ],
      ); // Column
    },
  ); // StreamBuilder
}

Widget _buildGeofenceNotifications() {
```

Figure 5.41: Parent Request Notification Feature

The *_buildRewardRequest* function also creates a widget that monitors the real-time reward request. It retrieves all the pending requests where ‘isApproved is false’ form the FIrestore database. When there are requests, each request is displayed in a card format, and showing with the task name, description, and the reward extra time in minutes. Parents can click “✓” as approve and scroll to the left to delete the card as reject.

5.3.1.11 Parent Setting Screen

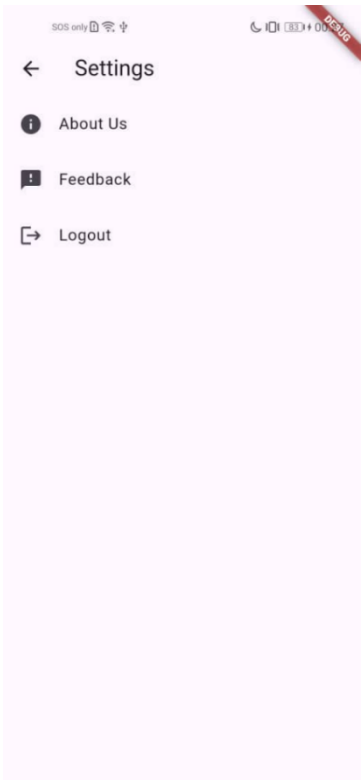


Figure 5.42: Parent Setting Screen

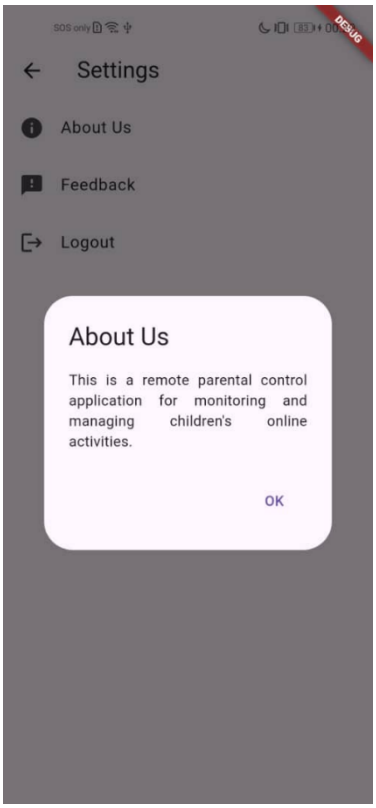


Figure 5.43: Parent About Us Feature

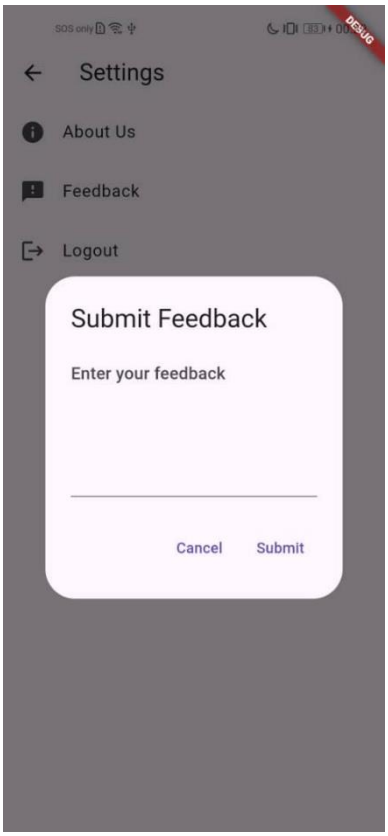


Figure 5.44: Parent Feedback Feature

```dart
void _logout() async {
  _stopDeviceListener();
  try {
    await _auth.signOut();
    Navigator.pushAndRemoveUntil(
      context,
      MaterialPageRoute(builder: (context) => OnboardingScreen()),
      (Route<dynamic> route) => false,
    );
  } catch (e) {
    print('Error during logout: $e');
    // Optionally, show an error message to the user
  }
}
```

Figure 5.45: Parent Logout Feature

In order to navigate to the setting screen, parents can press on the setting icon on the right top corner of the screen. The setting screen will show the about us, feedback from parents, and logout. The figures above show the dialog message of the application's information and the field to enter feedback and send to the developers' team. However, for the logout feature, the function of *_logout* is carried out. Firstly, the application will stop the listening process to ensure that

the application is stopped updating device data when the user logs out using the_*stopDeviceListener*. Then, the function tries to log out the current user by using *await _auth.signOut()*, which interacts with Firebase Authentication to close the user's session. After logging out successfully, the function utilizes *Navigator.pushAndRemoveUntil* to move to the Onboarding Screen and clear all previous routes from the navigation stack.

5.3.2 Children's Application

5.3.2.1 Login Screen

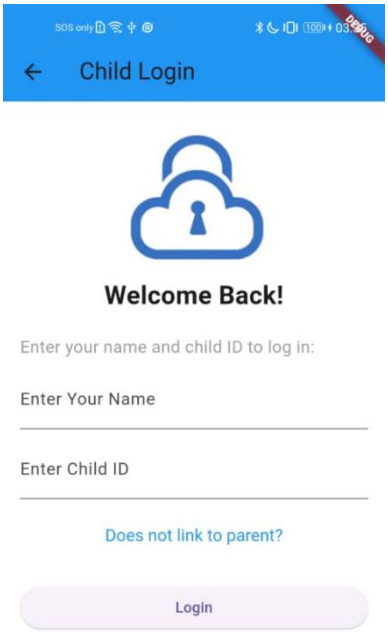


Figure 5.46: Child Login Screen

Once the children press the "Get Started as Children" in Onboarding Screen same as parent, then the application will navigate to the Login screen, where children can log in by providing their child ID and name. The code snippet shows that the *_login* function handle the login process of children. If the function finds that both fields are inputted, it will search the Firestore database for all parent documents. It loops through the subcollection of each parent's children to find a match for the given child ID and name. When a match is

detected, the function shows a success message and moves to the ChildDashboardScreen while sending the child's name, ID, and parent ID. Otherwise, an error message will be displayed to the user. If children click on the blue link “Does not link to parent?”, it will navigate to Child Connect Device Screen

5.3.2.2 Child Connect Device Sereen

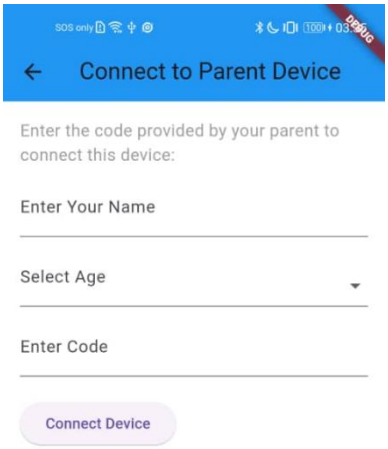
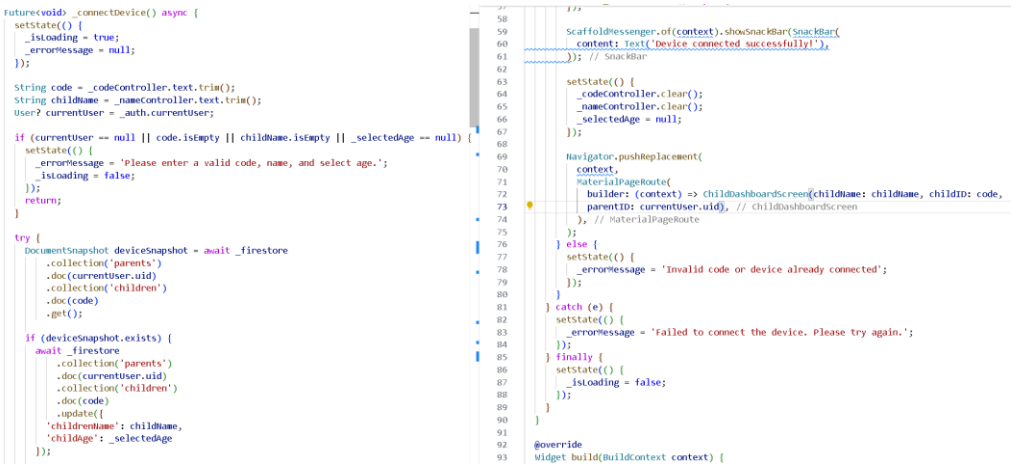


Figure 5.47: Child Connect Device Screen



```
Future<void> _connectDevice() async {
  setState(() {
    _isLoading = true;
    _errorMessage = null;
  });

  String code = _codeController.text.trim();
  String childName = _nameController.text.trim();
  User? currentUser = _auth.currentUser;

  if (currentUser == null || code.isEmpty || childName.isEmpty || _selectedAge == null) {
    setState(() {
      _errorMessage = 'Please enter a valid code, name, and select age.';
      _isLoading = false;
    });
    return;
  }

  try {
    DocumentSnapshot deviceSnapshot = await _firestore
        .collection('parents')
        .doc(currentUser.uid)
        .collection('children')
        .doc(code)
        .get();

    if (deviceSnapshot.exists) {
      await _firestore
          .collection('parents')
          .doc(currentUser.uid)
          .collection('children')
          .doc(code)
          .update({
        'childrenName': childName,
        'childAge': _selectedAge
      });
```

```
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(
        content: Text('Device connected successfully!'),
      )); // SnackBar

      setState(() {
        _codeController.clear();
        _nameController.clear();
        _selectedAge = null;
      });

      Navigator.pushReplacement(
        context,
        MaterialPageRoute(
          builder: (context) => ChildDashboardScreen(childName: childName, childID: code,
          parentID: currentUser.uid), // ChildDashboardScreen
        ), // MaterialPageRoute
      );
    } else {
      setState(() {
        _errorMessage = 'Invalid code or device already connected';
      });
    }
  } catch (e) {
    setState(() {
      _errorMessage = 'Failed to connect the device. Please try again.';
    });
  } finally {
    setState(() {
      _isLoading = false;
    });
  }
}

@override
Widget build(BuildContext context) {
```

Figure 5.48: Code of Child Connect Device Screen

If the children have not link to parent yet, they need to connect the link with parent device using the code generated in parent's device, this feature is also similar as sign up feature. The *_connectDevice* function facilitates this process by retrieving the connection code, child's name, and checks if the current user is logged in. The next step is to search Firestore to validate if the given code corresponds to a record in the children collection of the parent document. If a match is detected, it will then update the document with the child's name and age. After successfully linked, it will navigate to the Child Dashboard Screen by passing together with child's name, IS and parent ID.

5.3.2.3 Child User Profile Screen

Figure 5.49: Child User Profile Screen

The Child User Profile Screen shows the children information, including name, ID and age which retrieved from the database after they logged in. Children are only allowed to change their age and logout their account by pressing the logout icon on the right top corner.

5.3.2.4 Child Dashboard Screen

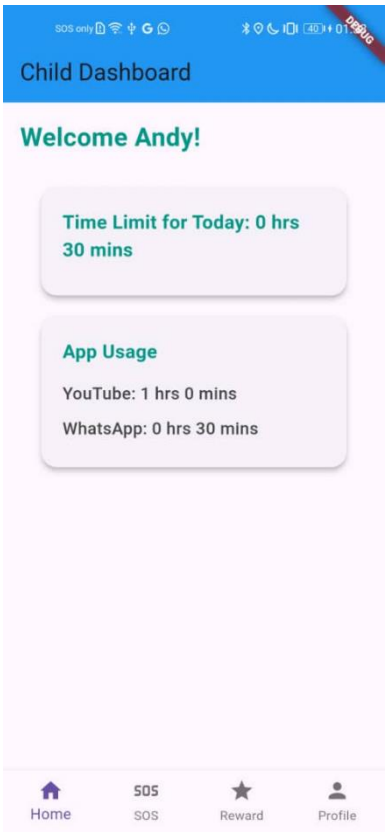


Figure 5.50: Child Dashboard Screen

After children login, the time limit for today and their app usage will be show in the dashboard. It helps the children to know about their time and usage on the device. The navigation bar also provided for children application. “Home” icon indicates the dashboard, “SOS” allow children to navigate to SOS screen for ask for help, “Reward” route to the Reward Screen for claim the extra time, and “Profile” allow the children to navigate to Child Profile Screen for view their data and logout.

5.3.2.5 Child SOS Screen

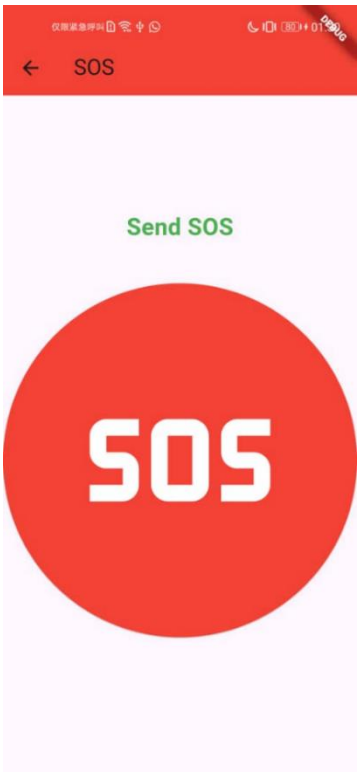
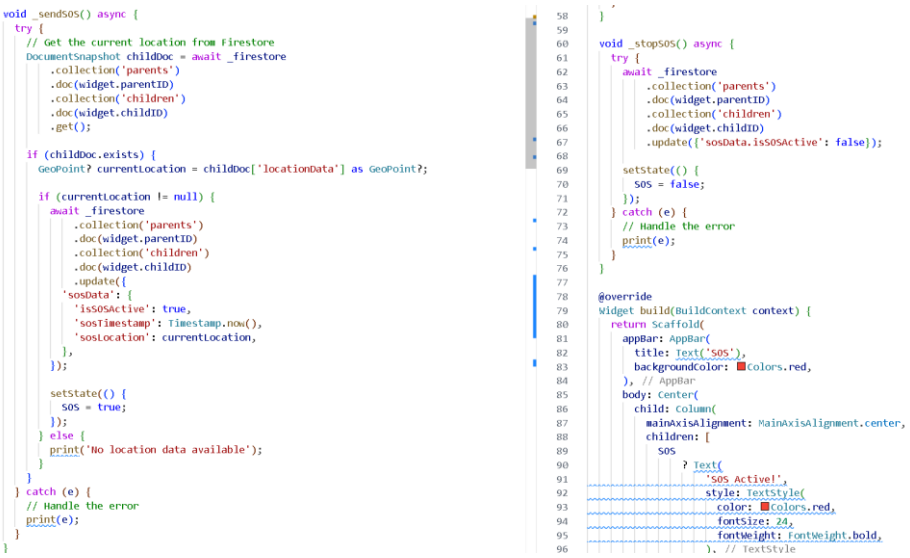


Figure 5.51: Child SOS Screen (send)

Figure 5.52: Child SOS Screen (Stop)



```
void _sendSOS() async {
  try {
    // Get the current location from Firestore
    DocumentSnapshot childDoc = await _firestore
        .collection('parents')
        .doc(widget.parentID)
        .collection('children')
        .doc(widget.childID)
        .get();

    if (childDoc.exists) {
      GeoPoint? currentLocation = childDoc['locationData'] as GeoPoint?;

      if (currentLocation != null) {
        await _firestore
            .collection('parents')
            .doc(widget.parentID)
            .collection('children')
            .doc(widget.childID)
            .update({
          'sosData': {
            'isSOSActive': true,
            'sosTimestamp': Timestamp.now(),
            'sosLocation': currentLocation,
          },
        });

        setState(() {
          SOS = true;
        });
      } else {
        print('No location data available');
      }
    }
  } catch (e) {
    // Handle the error
    print(e);
  }
}
```

```
}

void _stopSOS() async {
  try {
    await _firestore
        .collection('parents')
        .doc(widget.parentID)
        .collection('children')
        .doc(widget.childID)
        .update({'sosData.isSOSActive': false});

    setState(() {
      SOS = false;
    });
  } catch (e) {
    // Handle the error
    print(e);
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('SOS'),
      backgroundColor: Colors.red,
    ), // AppBar
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          SOS
              ? Text(
                  'SOS Active!',
                  style: TextStyle(
                    color: Colors.red,
                    fontSize: 24,
                    fontWeight: FontWeight.bold,
                  ), // TextStyle
```

Figure 5.53: Code of Child SOS Screen

When children are faced with urgent problems outdoors, they can navigate to the SOS feature from the navigation bar. After they press on the SOS, their location with SOS message will then send to parent device. Of course, when they stop the SOS, parents are also alerted because they can know that their children are safety now. From the code snippet, the *_sendSOS* function retrieves the child's location data from Firestore database. When the SOS is pressed, *isSOSActive* in SOSData will turn 'true' and store the updated location and timestamp. Same as well for the stop SOS, only the isSOSActice turn 'false' and update in the database. Therefore, the SOSData can retrieve from the parents.

5.3.2.6 Child Reward Screen

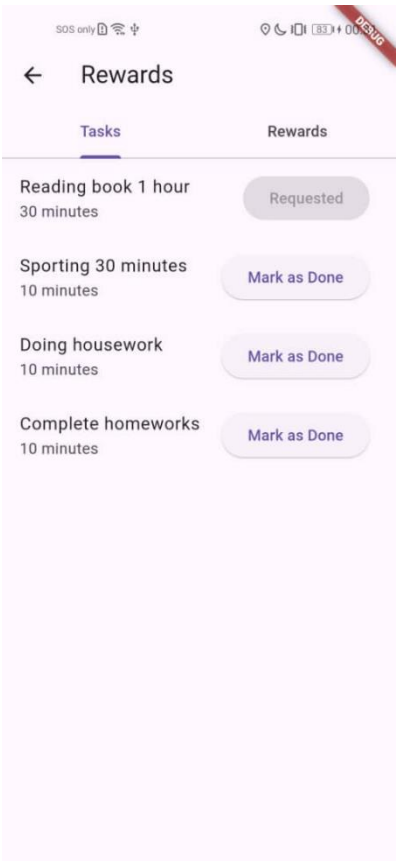


Figure 5.54: Child Reward Screen of Tasks Tab

When children navigate to Reward Screen, the tasks tab shows the tasks with the extra time given. After children done the task, they can mark the task as done, then the request will then send to the parents for action. Once the tasks is done today, it will only refresh on the next day.

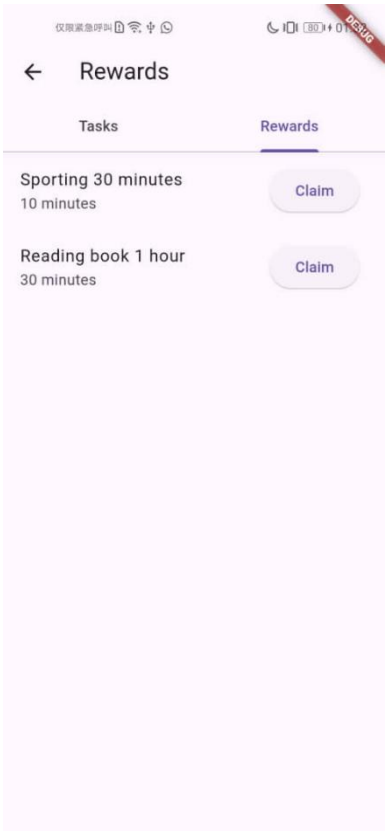


Figure 5.55: Child Reward Screen with Rewards Tab

After the parent approve with the task, the rewards will then send back to the children. They can claim the reward for the extra time and add the extra time to the daily time limit.

```
Future<void> _requestReward(Map<String, dynamic> task) async {
  if (_taskCompletionStatus[task['taskName']] ?? false) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('This task has already been requested today.')),
    );
    return;
  }

  try {
    await createRewardRequest(
      parentID: widget.parentID,
      childID: widget.childID,
      taskName: task['taskName'],
      timeReward: task['timeReward'],
      taskDescription: task['taskDescription'],
    );

    setState(() {
      _taskCompletionStatus[task['taskName']] = true; // Mark task as completed today
    });

    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Reward request sent to parent for approval.')),
    );
  } catch (e) {
    print(e);
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Failed to send reward request.')),
    );
  }
}
```

Figure 5.56: Code of Child Reward Screen

Based on the code snippet, the *_requestReward* function manages the process of requesting a reward for completing a task. After children mark the task, the function will create a reward request using *createRewardRequest* method which involve sending the task details such as the parent ID, child ID, task name, time reward, and task description, to the appropriate backend service. Upon successful sending of the reward request, the function updates the *_taskCompletionStatus* map to mark the task as completed for the day and displays a SnackBar message indicating that the reward request has been sent for parental approval.

CHAPTER 6
TESTING

6.1 Introduction

In this chapter, the selected software testing is Unit Test and User Acceptance Test (UAT). Software testing is important in every application to ensure that the system quality before delivery to the end user. User Acceptance Test (UAT) is the test which collect the feedback from the user when letting them to try using the application.

6.2 Unit Test

Manual Unit Test is selected to test the functionality and characteristic of the application. Black Box Testing is one of the methods used to check the functionality of the whole application by providing the different test cases manually. Every test case consists of selected inputs, and the actual output is compared to the expected results to verify accuracy. The test results of error messages will display in Appendix C.

Module Name	Parental Control Application
Created by	Tan Rocher
Date of Creation	28/8/2024
Date of Review	10/9/2024

6.2.1 Test Case 001: Check the user authentication for parent login feature

Table 6.1: Verify parent login with valid credentials

Test Case No 1	
Scenario	Verify parent login with valid credentials
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Enter a valid email and password. 4. Tap "Login".
Test Data	Email= zaza@gmail.com Password = Ab@13579

Expected Result	Successfully logged in and directed to Parent Dashboard
Actual Result	Successfully logged in and directed to Parent Dashboard
Status	Pass

Table 6.2: Verify parent login with invalid email

Test Case No 2	
Scenario	Verify parent login with invalid email
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Enter an invalid email. 4. Tap "Login".
Test Data	Email= zazaza@gmail.com Password = Ab@13579
Expected Result	Display error message: “Please enter a correct email or password.”
Actual Result	Display error message: “Please enter a correct email or password.”
Status	Pass

Table 6.3: Verify parent login with incorrect password

Test Case No 3	
Scenario	Verify parent login with incorrect password
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Enter a valid email and incorrect password. 4. Tap "Login".
Test Data	Email= zaza@gmail.com Password = Aa%123321
Expected Result	Display error message: “Please enter a correct email or password.”
Actual Result	Display error message: “Please enter a correct email or password.”
Status	Pass

Table 6.4: Verify login with empty email field

Test Case No 4	
Scenario	Verify login with empty email field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Leave the email field empty. 4. Tap "Login".
Test Data	Email= “” Password = Ab@13579
Expected Result	Display error message: “Please enter your email”
Actual Result	Display error message: “Please enter your email”
Status	Pass

Table 6.5: Verify login with empty password field

Test Case No 5	
Scenario	Verify login with empty password field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Leave the password field empty. 4. Tap "Login".
Test Data	Email= zaza@gmail.com Password = “”
Expected Result	Display error message: “Please enter your email”
Actual Result	Display error message: “Please enter your password”
Status	Pass

Table 6.6: Verify login with incorrect format email

Test Case No 6	
Scenario	Verify login with incorrect format email
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Enter an incorrect email and password.

	4. Tap "Login".
Test Data	Email= zaza@com Password = Ab@13579
Expected Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Actual Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Status	Pass

Table 6.7: Verify login with incorrect format password

Test Case No 7	
Scenario	Verify login with incorrect format password
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Enter a valid email and incorrect format password. 4. Tap "Login".
Test Data	Email= zaza@com Password = 123
Expected Result	Display error message: “Password must be at least 8 characters long”
Actual Result	Display error message: “Password must be at least 8 characters long”
Status	Pass

6.2.2 Test Case 002: Check the Sign Up feature for parents’ device

Table 6.8: Verify Sign Up with valid details

Test Case No 1	
Scenario	Verify Sign Up with valid details
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up”

	4. Enter a valid username, email, password, and confirm password. 5. Tap "Sign Up".
Test Data	Username: John Email: john@gmail.com Password: Aa!123321 Confirm password: Aa!123321
Expected Result	Successfully registered and directed to Parent Dashboard
Actual Result	Successfully registered and directed to Parent Dashboard
Status	Pass

Table 6.9: Verify Sign Up with invalid email format

Test Case No 2	
Scenario	Verify Sign Up with invalid email format
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter a valid username, invalid email, and valid password and confirm password. 5. Tap "Sign Up".
Test Data	Username: John Email: john.gmail.com Password: Aa!123321 Confirm password: Aa!123321
Expected Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Actual Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Status	Pass

Table 6.10: Verify signup with incorrect format password

Test Case No 3	
Scenario	Verify signup with incorrect format password

Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter a valid username, email, and password 5. Tap "Sign Up".
Test Data	Username: John Email: john.gmail.com Password: 12312
Expected Result	Display error message: “ Password must contain at least: • One uppercase letter • One lowercase letter • One symbol • At least 8 characters ”
Actual Result	Display error message: “ Password must contain at least: • One uppercase letter • One lowercase letter • One symbol • At least 8 characters ”
Status	Pass

Table 6.11: Verify signup with no match password

Test Case No 4	
Scenario	Verify signup with no match password
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter a valid username, email, password and confirm password. 5. Tap "Sign Up".

Test Data	Username: John Email: john.gmail.com Password: Aa!123321 Confirm password: Ab#123121
Expected Result	Display error message: “Password do not match”
Actual Result	Display error message: “Password do not match”
Status	Pass

Table 6.12: Verify Sign Up with empty username field

Test Case No 5	
Scenario	Verify Sign Up with empty username field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter an empty username, valid email, password, and confirm password. 5. Tap "Sign Up".
Test Data	Username: “” Email: john@gmail.com Password: Aa!123321 Confirm password: Aa!123321
Expected Result	Display error message: “Please enter your username”
Actual Result	Display error message: “Please enter your username”
Status	Pass

Table 6.13: Verify Sign Up with empty email field

Test Case No 6	
Scenario	Verify Sign Up with empty email field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter an empty email, valid username, password, and confirm password.

	5. Tap "Sign Up".
Test Data	Username: John Email: “” Password: Aa!123321 Confirm password: Aa!123321
Expected Result	Display error message: “Please enter your email (abc@gmail.com)”
Actual Result	Display error message: “Please enter your email (abc@gmail.com)”
Status	Pass

Table 6.14: Verify Sign Up with empty password field

Test Case No 7	
Scenario	Verify Sign Up with empty password field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Sign Up” 4. Enter an empty password, valid username, email, and confirm password. 5. Tap "Sign Up".
Test Data	Username: John Email: john@gmail.com Password: “” Confirm password: “”
Expected Result	Display error message: “Please enter your password”
Actual Result	Display error message: “Please enter your password”
Status	Pass

Table 6.15: Verify Sign Up with already registered email

Test Case No 8	
Scenario	Verify Sign Up with already registered email
Test Step	1. Open the app. 2. Select “Get Started as Parent”.

	3. Click on the “Sign Up” 4. Enter a valid username, email that’s already registered, and password. 5. Tap "Sign Up".
Test Data	Username: Zaza Email: zaza@gmail.com Password: Aa!123321 Confirm password: Aa!123321
Expected Result	Display error message: “This email address us already in use by other account”
Actual Result	Display error message: “This email address us already in use by other account”
Status	Pass

6.2.3 Test Case 003: Check the forget password feature for parents’ device

Table 6.16: Verify Sign Up with valid email

Test Case No 1	
Scenario	Verify Sign Up with valid email
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Forgot Password” 4. Enter a valid email. 5. Tap "Send Reset Link".
Test Data	Email: zaza@gmail.com
Expected Result	Successfully send the password reset link and can check in inbox.
Actual Result	Successfully send the password reset link and can check in inbox.
Status	Pass

Table 6.17: Verify Sign Up with invalid email

Test Case No 2	
Scenario	Verify Sign Up with invalid email
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Forgot Password” 4. Enter an invalid email. 5. Tap "Send Reset Link".
Test Data	Email: zaza.gmail.com
Expected Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Actual Result	Display error message: “Please enter a valid email (abc@gmail.com)”
Status	Pass

Table 6.18: Verify Sign Up with empty email field

Test Case No 3	
Scenario	Verify Sign Up with empty email field
Test Step	1. Open the app. 2. Select “Get Started as Parent”. 3. Click on the “Forgot Password” 4. Enter an empty email. 5. Tap "Send Reset Link".
Test Data	Email: “”
Expected Result	Display error message: “Please enter your email”
Actual Result	Display error message: “Please enter your email”
Status	Pass

6.2.4 Test Case 004: Check parent user profile feature

Table 6.19: Verify parent profile information is displayed correctly

Test Case No 1	
Scenario	Verify parent profile information is displayed correctly

Test Step	1. Open the app. 2. Login as a parent. 3. Navigate to the Parent Profile screen.
Test Data	Logged in parent account with details like username and email saved.
Expected Result	Display username and connected device.
Actual Result	Display username and connected device.
Status	Pass

Table 6.20: Verify parent can change the password

Test Case No 2	
Scenario	Verify parent can change the password
Test Step	1. Open the app. 2. Login as a parent. 3. Navigate to the Parent Profile screen. 4. Click on “Edit Password”. 5. Enter the current password and new password with confirm password. 6. Tap “Change Password”.
Test Data	Current Password: Ab@13579 New Password: Zaza!112 Confirm password: Zaza!112
Expected Result	Password successfully changed and return back to profile screen.
Actual Result	Password successfully changed and return back to profile screen.
Status	Pass

Table 6.21: Verify parent can add a child device

Test Case No 3	
Scenario	Verify parent can add a child device
Test Step	1. Open the app. 2. Login as a parent.

	3. Navigate to the Parent Profile screen. 4. Click on “Add Device”. 5. Generate a unique code and connect a child’s device using that code.
Test Data	Use the code in child’s device
Expected Result	The child device is successfully linked and displayed under the parent’s profile with the child's name and id.
Actual Result	The child device is successfully linked and displayed under the parent’s profile with the child's name and id.
Status	Pass

Table 6.22: Verify parent can remove a child device

Test Case No 4	
Scenario	Verify parent can remove a child device
Test Step	1. Open the app. 2. Login as a parent. 3. Navigate to the Parent Profile screen. 4. Locate the connected child device. 5. Tap the "Remove" icon next to the child device.
Test Data	Tap the "Remove" icon next to the child device.
Expected Result	The child device is successfully removed from the parent’s profile.
Actual Result	The child device is successfully removed from the parent’s profile.
Status	Pass

Table 6.23: Verify parent can log out

Test Case No 5	
Scenario	Verify parent can log out
Test Step	1. Open the app. 2. Login as a parent. 3. Click on the “setting” icon on right top conner. 4. Tap the "Logout" button.

Test Data	Tap the "Logout" button.
Expected Result	The parent is logged out and returned to the onboarding screen.
Actual Result	The parent is logged out and returned to the onboarding screen.
Status	Pass

6.2.5 Test Case 005: Check the Dashboard for parent's device

Table 6.24: Verify children icon is display and can be selected in dashboard

Test Case No 1	
Scenario	Verify children icon is display and can be selected in dashboard
Test Step	1. Open the app. 2. Login as a parent. 3. Click left or right to change child.
Test Data	Child's icon
Expected Result	Child is successfully to display and change.
Actual Result	Child is successfully to display and change.
Status	Pass

Table 6.25: Verify the selected child's time limit and app usage is display in the dashboard

Test Case No 2	
Scenario	Verify the selected child's time limit and app usage is display in the dashboard
Test Step	1. Open the app. 2. Login as a parent. 3. Wait for the time limit and app usage to update.
Test Data	Time limit and app usage
Expected Result	Time limit and app usage are successfully displayed.
Actual Result	Time limit and app usage are successfully displayed.
Status	Pass

Table 6.26: Verify the navigator can successfully to navigate

Test Case No 3	
Scenario	Verify the navigator can successfully to navigate
Test Step	1. Open the app. 2. Login as a parent. 3. Click on each navigator.
Test Data	Home: Dashboard Screen Location: Location Tracking Screen Notification: Notification Screen Profile: Profile Screen Set Time: Set Time Limit Screen Filter: Content Filtering Screen Setting: Setting Screen
Expected Result	All the screens are successfully to navigate.
Actual Result	All the screens are successfully to navigate.
Status	Pass

6.2.6 Test Case 006: Check the Time Limit Feature for parents' device

Table 6.27: Verify that a time limit can be set for a child device

Test Case No 1	
Scenario	Verify that a time limit can be set for a child device
Test Step	1. Open the app. 2. Log in as a parent. 3. Select the children. 4. Click on the "Set Time" button. 5. Set a daily screen time limit (e.g., 2 hours) for a week. 6. Save the changes.
Test Data	Monday: 30 min Tuesday: 30 min Wednesday: 30 min Thursday: 30 min

	Friday: 1 hr Saturday: 2 hr Sunday: 1 hr 30 min
Expected Result	The time limit is successfully set, and it will show in the dashboard for daily.
Actual Result	The time limit is successfully set, and it will show in the dashboard for daily.
Status	Pass

Table 6.28: Verify that a time limit for today can be display in the dashboard

Test Case No 2	
Scenario	Verify that a time limit for today can be display in the dashboard
Test Step	1. Open the app. 2. Log in as a parent. 3. Select the children. 4. Wait for the time limit today for update.
Test Data	Thursday: 30 min
Expected Result	The time limit for today is shown.
Actual Result	The time limit for today is shown.
Status	Pass

6.2.7 Test Case 007: Check the Content Filtering Feature for parents' device

Table 6.29: Verify that content filtering can be enabled for a child device

Test Case No 1	
Scenario	Verify that content filtering can be enabled for a child device
Test Step	1. Open the app. 2. Log in as a parent. 3. Select the children. 4. Click on the "Filter" button.

	5. Toggle the content filtering option to enable it. 6. Save the changes.
Test Data	Enable apply filtering
Expected Result	Content filtering is successfully enabled, and the child device restricts access to blocked for the selected content.
Actual Result	Content filtering is successfully enabled, and the child device restricts access to blocked for the selected content.
Status	Pass

Table 6.30: Verify that content filtering categories can be selected and applied

Test Case No 2	
Scenario	Verify that content filtering categories can be selected and applied
Test Step	1. Open the app. 2. Log in as a parent. 3. Select the children. 4. Click on the "Filter" button. 5. Select categories to block. 6. Save the changes.
Test Data	Categories: Gambling, Video Streaming
Expected Result	The selected categories are successfully blocked, and the child device is blocked to access those websites related to the categories.
Actual Result	The selected categories are successfully blocked, and the child device is blocked to access those websites related to the categories.
Status	Pass

Table 6.31: Verify that the parent can update the blocked categories

Test Case No 3	
Scenario	Verify that the parent can update the blocked categories
Test Step	1. Open the app. 2. Log in as a parent.

	3. Select the children. 4. Click on the "Filter" button. 5. Change the blocked categories. 6. Save the changes.
Test Data	Original Categories: Gambling, Video Streaming Updated Categories: Gambling, Youtube
Expected Result	The updated categories are successfully applied.
Actual Result	The updated categories are successfully applied.
Status	Pass

6.2.8 Test Case 008: Check the Location Tracking Feature for parents' device

Table 6.32: Verify that the parent can view the real-time location of the child device

Test Case No 1	
Scenario	Verify that the parent can view the real-time location of the child device
Test Step	1. Open the app. 2. Log in as a parent. 3. Navigate to location.
Test Data	Real-time location tracking
Expected Result	The real-time location of the child device is displayed on the map.
Actual Result	The real-time location of the child device is displayed on the map.
Status	Pass

Table 6.33: Verify that parent can mark a place with a range of meter

Test Case No 2	
Scenario	Verify that parent can mark a place with a range of meter
Test Step	1. Open the app. 2. Log in as a parent.

	3. Navigate to location. 4. Long press on the map. 5. Enter the name, select the radius, and colour. 6. Save the changes.
Test Data	Place Name: school Radius: 600 m Colour: Blue
Expected Result	A geofence is successfully set around the specified location, and the area is marked with the colour on the map.
Actual Result	A geofence is successfully set around the specified location, and the area is marked with the colour on the map.
Status	Pass

Table 6.34: Verify that parent can view the detail of marked place

Test Case No 3	
Scenario	Verify that parent can view the detail of marked place
Test Step	1. Open the app. 2. Log in as a parent. 3. Navigate to location. 4. Tap the marked place on the map.
Test Data	Place Name: school Radius: 600 m Colour: Blue
Expected Result	The Place Name, Radius and Colour is displayed.
Actual Result	The Place Name, Radius and Colour is displayed.
Status	Pass

Table 6.35: Verify that parent can edit the marked place

Test Case No 4	
Scenario	Verify that parent can edit the marked place
Test Step	1. Open the app.

	2. Log in as a parent. 3. Navigate to location. 4. Tap the marked place on the map. 5. Enter the name, select the radius, and colour. 6. Save the changes.
Test Data	Original: Place Name: school Radius: 600 m Colour: Blue Updated: Place Name: school Radius: 1000 m Colour: Yellow
Expected Result	The marked is successfully updated.
Actual Result	The marked is successfully updated.
Status	Pass

Table 6.36: Verify that parent can delete the marked place

Test Case No 5	
Scenario	Verify that parent can delete the marked place
Test Step	1. Open the app. 2. Log in as a parent. 3. Navigate to location. 4. Tap the marked place on the map. 5. Click on delete.
Test Data	Place Name: school Radius: 600 m Colour: Blue
Expected Result	The marked is successfully deleted.
Actual Result	The marked is successfully deleted.
Status	Pass

Table 6.37: Verify that parent can receive notification when child enter or leave the marked place

Test Case No 6	
Scenario	Verify that parent can receive notification when child enter or leave the marked place
Test Step	1. Open the app. 2. Log in as a parent. 3. Navigate to location. 4. Wait for the child device to enter or leave the marked place.
Test Data	Place Name: school Radius: 600 m Colour: Blue
Expected Result	The parent receives a notification when the child device enters or exits the marked place.
Actual Result	The parent receives a notification when the child device enters or exits the marked place.
Status	Pass

6.2.9 Test Case 009: Check the Child Login

Table 6.38: Verify child login with valid credentials

Test Case No 1	
Scenario	Verify child login with valid credentials
Test Step	1. Open the app. 2. Select “Get Started as Child”. 3. Enter the code generated by the parent and their name. 4. Tap "Login".
Test Data	Name: Herry Child ID: 1353
Expected Result	Successfully logged in and directed to Child Dashboard
Actual Result	Successfully logged in and directed to Child Dashboard
Status	Pass

Table 6.39: Verify child login with invalid child ID

Test Case No 2	
Scenario	Verify child login with invalid child ID
Test Step	1. Open the app. 2. Select “Get Started as Child”. 3. Enter the invalid code generated by the parent and their name. 4. Tap "Login".
Test Data	Name: Herry Child ID: 2222
Expected Result	Display error message “Invalid name or child ID. Please try again.”
Actual Result	Display error message “Invalid name or child ID. Please try again.”
Status	Pass

Table 6.40: Verify child login with empty name or child ID field

Test Case No 3	
Scenario	Verify child login with empty name or child ID field
Test Step	1. Open the app. 2. Select “Get Started as Child”. 3. Enter the invalid code generated by the parent and their name. 4. Tap "Login".
Test Data	Name: “”, child ID: 1352 Name: Herry, Child ID: “” Name: “”, child ID: “”
Expected Result	Display error message “Please enter your name and child ID.”
Actual Result	Display error message “Please enter your name and child ID.”
Status	Pass

Table 6.41: Verify child connect with parent device

Test Case No 4	
Scenario	Verify child connect with parent device
Test Step	1. Open the app. 2. Select “Get Started as Child”. 3. Click on “Does not link to parent?” 4. Enter the valid child ID, name and age. 5. Tap “Connect Device”.
Test Data	Name: Herry Age: 10 Child ID: 1353
Expected Result	Successfully logged in and directed to Child Dashboard
Actual Result	Successfully logged in and directed to Child Dashboard
Status	Pass

Table 6.42: Validate the child connect field

Test Case No 5	
Scenario	Validate the child connect field
Test Step	1. Open the app. 2. Select “Get Started as Child”. 3. Click on “Does not link to parent?” 4. Enter the empty child ID or name or age field. 5. Tap “Connect Device”.
Test Data	Name: “”, Age: 10, Child ID: 1353 Name: Herry, Age: “”, Child ID: 1353 Name: Herry, Age: 10, Child ID: “” Name: “”, Age: “”, Child ID: 1353 Name: Herry, Age: “”, Child ID: “” Name: Herry, Age: “”, Child ID: 1353 Name: “”, Age: “”, Child ID: “”
Expected Result	Display error message “Please enter a valid code, name, and select age.”

Actual Result	Display error message “Please enter a valid code, name, and select age.”
Status	Pass

6.2.10 Test Case 0010: Check the Reward Feature for children’s device

Table 6.43: Verify that child can request the reward after done the task

Test Case No 1	
Scenario	Verify that child can request the reward after done the task
Test Step	1. Open the app. 2. Login as child. 3. Navigate to Reward screen. 4. In the tasks tab, click “mark as done” for the task to send request to parent.
Test Data	Reading book 1 hour: mark as done
Expected Result	Successfully send the request to parent.
Actual Result	Successfully send the request to parent.
Status	Pass

Table 6.44: Verify that child can claim the reward after parent approve

Test Case No 2	
Scenario	Verify that child can claim the reward after parent approve
Test Step	1. Open the app. 2. Login as child. 3. Navigate to Reward screen. 4. In the rewards tab, click “claim” for the task to send add extra time.
Test Data	Reading book 1 hour: claimed
Expected Result	Successfully added the extra time to the time today.
Actual Result	Successfully added the extra time to the time today.
Status	Pass

Table 6.45: Verify that parent can receive the request from the child

Test Case No 3	
Scenario	Verify that parent can receive the request from the child
Test Step	1. Open the app. 2. Login as parent. 3. Navigate to Notification screen. 4. Wait for child send the request.
Test Data	Reading book 1 hour: request
Expected Result	The parent receives a notification when the child send request.
Actual Result	The parent receives a notification when the child send request.
Status	Pass

Table 6.46: Verify that parent can action on the request from the child

Test Case No 4	
Scenario	Verify that parent can action on the request from the child
Test Step	1. Open the app. 2. Login as parent. 3. Navigate to Notification screen. 4. Click on "✓" as approve and "×" as reject.
Test Data	Reading book 1 hour: ✓
Expected Result	The reward successfully returns to child when approved.
Actual Result	The reward successfully returns to child when approved.
Status	Pass

6.2.11 Test Case 011: Check the SOS Feature for children's device

Table 6.47: Verify that child can send SOS to parent

Test Case No 1	
Scenario	Verify that child can send SOS to parent
Test Step	1. Open the app.

	2. Login as children. 3. Navigate to SOS screen. 4. Click on the “SOS” button to send.
Test Data	Send: SOS
Expected Result	The SOS is active and send an alert to parent.
Actual Result	The SOS is active and send an alert to parent.
Status	Pass

Table 6.48: Verify that child can stop the SOS

Test Case No 2	
Scenario	Verify that child can stop the SOS
Test Step	1. Open the app. 2. Login as children. 3. Navigate to SOS screen. 4. Click on the “stop” button to stop.
Test Data	Stop: SOS
Expected Result	The SOS is stopped and send an alert to parent.
Actual Result	The SOS is stopped and send an alert to parent.
Status	Pass

Table 6.49: Verify that parent can receive notification of SOS

Test Case No 3	
Scenario	Verify that parent can receive notification of SOS
Test Step	1. Open the app. 2. Login as parent. 3. Navigate to Notification screen. 4. Wait for children send or stop SOS.
Test Data	Send: SOS Location: Latitude:3.0411, Longitude: 101.7882 Timestamp: 2024-08-28 23:35:14.883182
Expected Result	The parent receives a notification when the children send or stop SOS.

Actual Result	The parent receives a notification when the children send or stop SOS.
Status	Pass

6.3 User Acceptance Test (UAT)

User Acceptance Test (UAT) is conducted through a Google Form questionnaire. The Google Form can view at the Appendix B. It is aimed to collect the feedback from user after they have used the application. Users' feedback is important for ensuring that the application meets the needs of parents and helps guide future enhancements. By the survey form, the usability, performance and functionality of the application was evaluated.

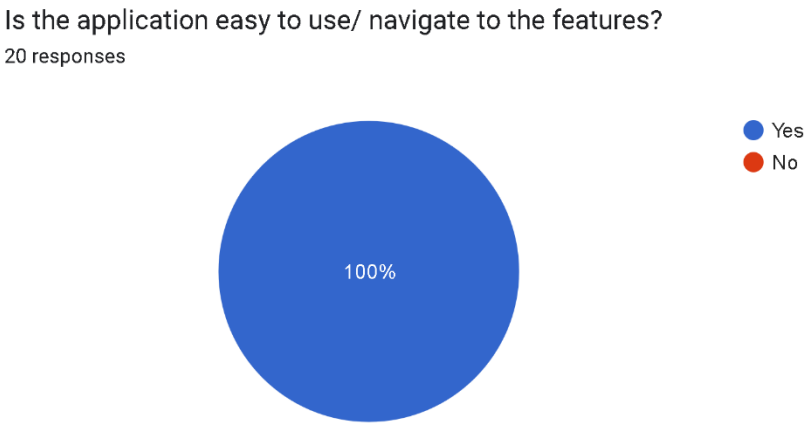


Figure 6.1: Result of Usability Question 1

Figure 6.1 shows the user-friendliness of the respondents to the application. All the respondents (100%) are feel that the application is easy to use. Therefore, it indicates that the application is meet the requirements of user-friendly interface which simple the user to interact with.

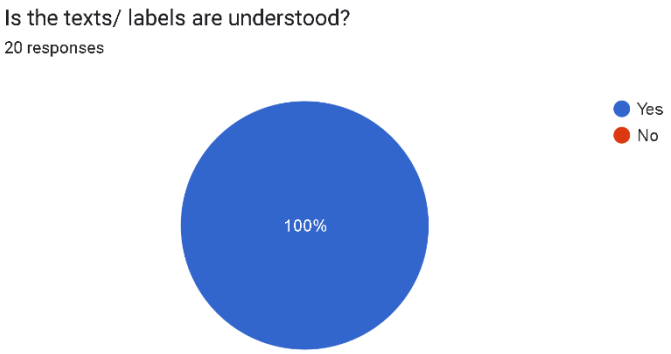


Figure 6.2: Result of Usability Question 2

The result of question 2 as shown in Figure 6.2 also shows that all the respondents (100%) are agree that the application is clear and understandable. They found the text and labels used within the app to be easy to comprehend. This feedback highlighted the easy understanding of the application's text and labels.

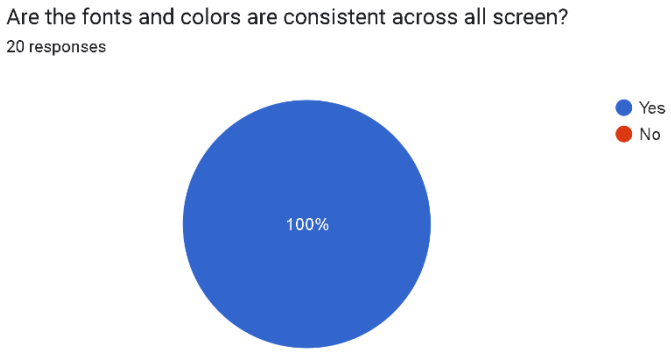


Figure 6.3: Result of Usability Question 3

Figure 6.3 indicates that all the respondents (100%) think that the fonts and colours are consistent. This feedback suggests that the application has a consistent and uniform design with the same font styles and colour schemes across all screens. It creates a visually appealing and well user experience.

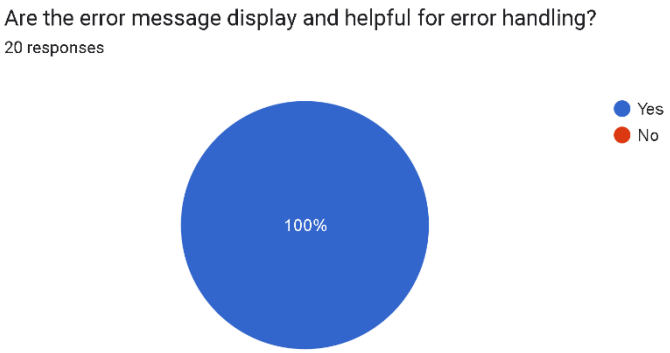


Figure 6.4: Result of Usability Question 4

From the result in Figure 6.4, all the respondents (100%) found that the application will show the error messages when the input data is wrong. Therefore, it means that the error messages are clear, informative and useful in guiding them to resolve the issues effectively. The feedback indicates that the application provides an enough assistance in managing error, enhancing the overall usability and user experience.

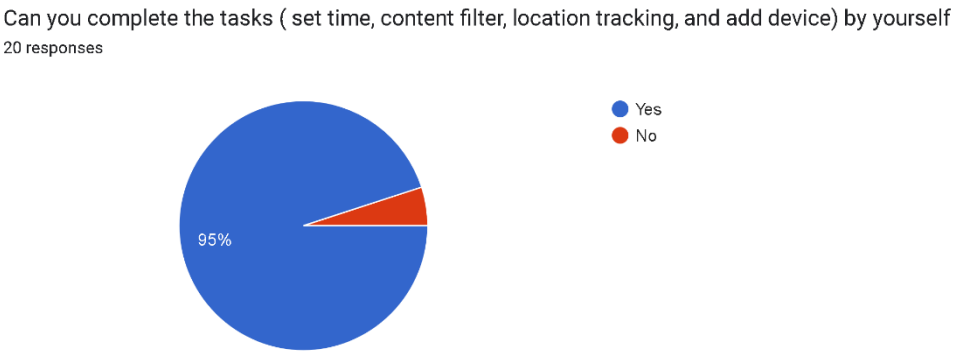


Figure 6.5: Result of Usability Question 5

95% of respondents can complete the tasks without helping and 5% of responses cannot do it, which the result is get from the Figure 6.5. It indicates that majority found that the application is easy to navigate and use independently for those features while minority of users faced some problem. This feedback emphasizes the importance of possibly enhancing user guidance or feature accessibility to ensure that all users can accomplish tasks independently.

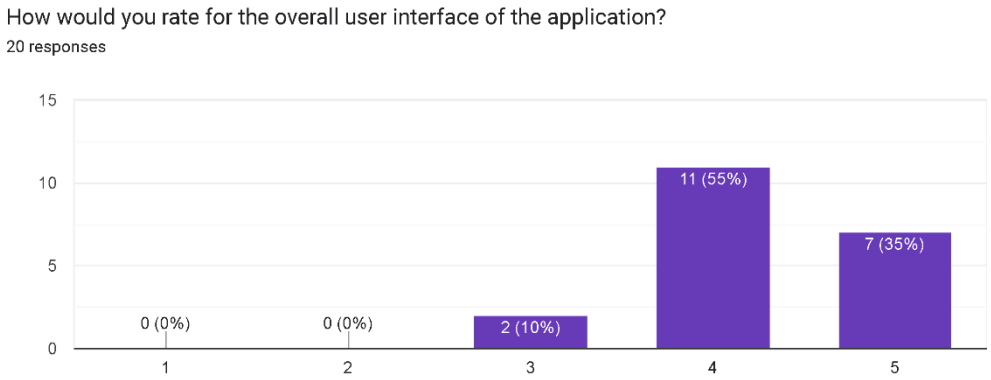


Figure 6.6: Result of Usability Question 6

Figure 6.6 shows the rating of parents to the user interface of the application. 35% of respondents rated it as excellent with 5, 55% gave it a rating of 4, and 10% rated it a 3. The majority of users (90%) are highly satisfied with the user interface as they perceive it as visually pleasing and user-friendly. However, 10% of users feel that the user interface still needs to enhance.

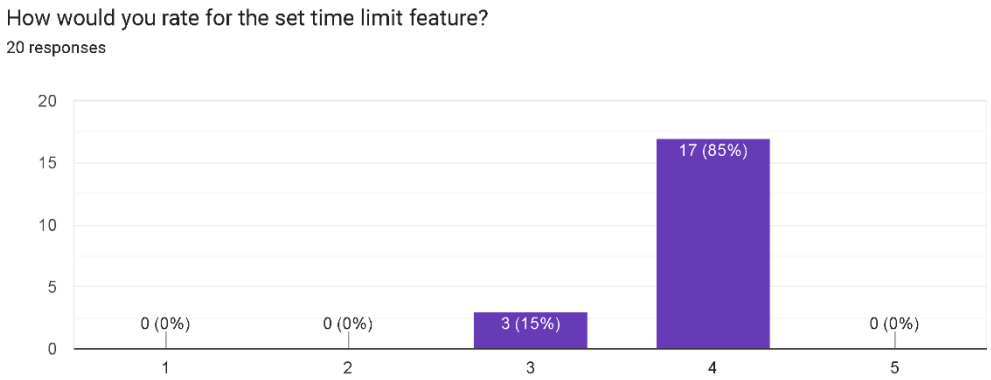


Figure 6.7: Result of Functionality Question 1

In the response to the question of rating the set time limit feature as shown in Figure 6.7, 85% of respondents rated it as 4, while 15% rated it as 3. It shows that most users find its function and the usefulness of managing screen time, and it meets their expectations. While the 15% implies that some users may have

faced minor obstacles or found ways in which the feature could be enhanced for improved usability and performance.

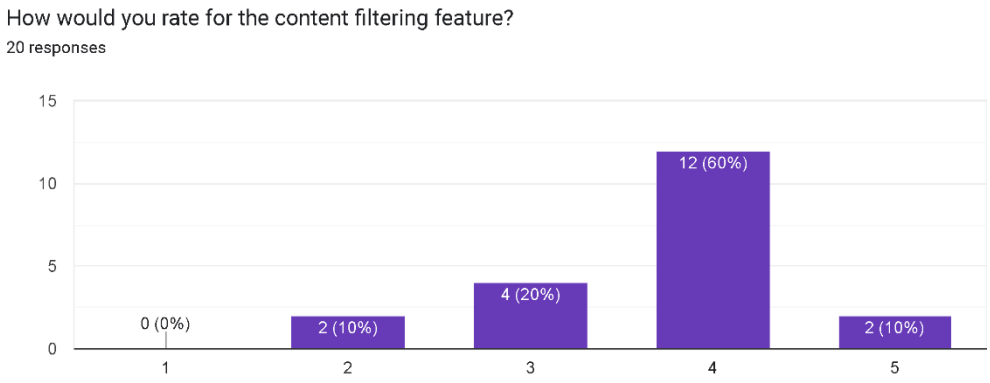


Figure 6.8: Result of Functionality Question 2

In Figure 6.8, question has ask about "How would you rate the content filtering feature?", which 10% of respondents rated it as 2, 20% rated it as 3, 60% gave it as 4, and 10% rated it as 5. It shows that most (70%) users have a positive evaluation of this feature (with rates of 4 and 5), which indicates the overall satisfaction with the content filtering function, while 30% of respondents rate it lower (with rates 2 and 3). This indicates that although the feature is effective for the majority of users, there are certain aspects that require enhancement, such as add more for the block categories to better meet their expectations.

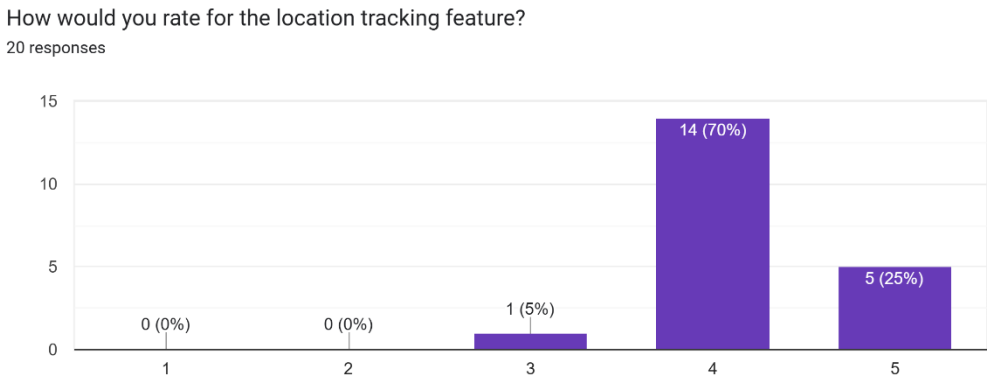


Figure 6.9: Result of Functionality Question 3

The result in Figure 6.9 indicates that the majority of users (95%) are satisfied with the location tracking feature, with 70% giving it a rating of 4 and 25% rating it as excellent, which means that it is already meet the needs of most users. he 5% rating it as 3 suggests that even the feature was useful, but still have some minor improvements needed to optimize its accuracy.

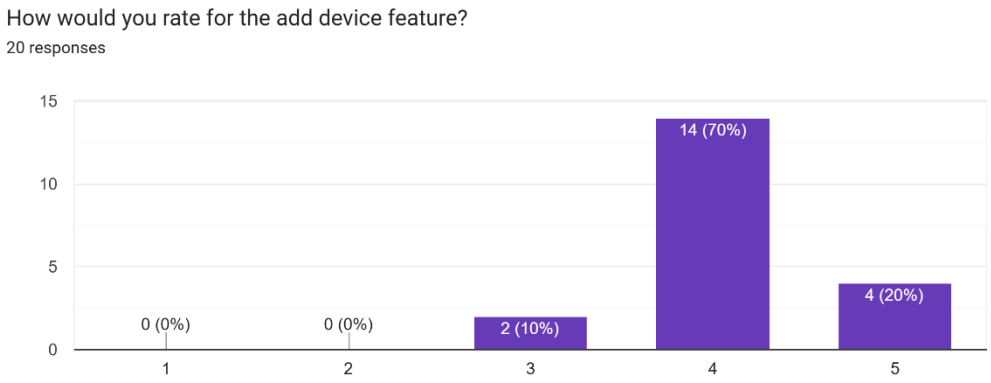


Figure 6.10: Result of Functionality Question 4

The ratings for the "Add Device" feature shown in Figure 6.10 indicates that 10% of respondents rated it a 3, 70% rated it a 4, and 20% rated it a 5. The majority of users giving it a 4 or 5 reflects a strong overall approval, suggesting that the feature is usually effective and well-liked. The 10% giving it a 3 hint at possible areas to work on. Improving these aspects could boost the feature to a greater level of user satisfaction.

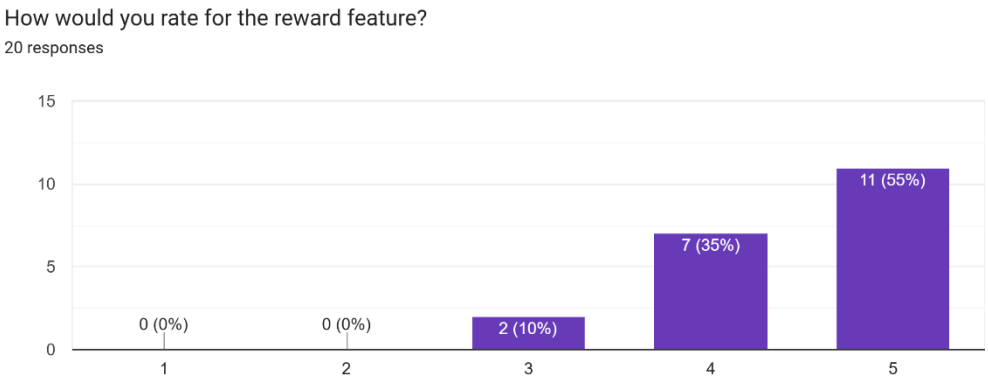


Figure 6.11: Result of Functionality Question 5

The "Reward" feature received positive feedback, with 55% of respondents rating it as 5, 35% rating it as 4, and 10% rating it as 3, which shown in Figure 6.11. It shows the high effective and valuable of the feature for most users. However, still have to make enhancement like parent can assign task as 10% who rated it as 3 suggest that maintain and enhance the application can make the application more prefect.

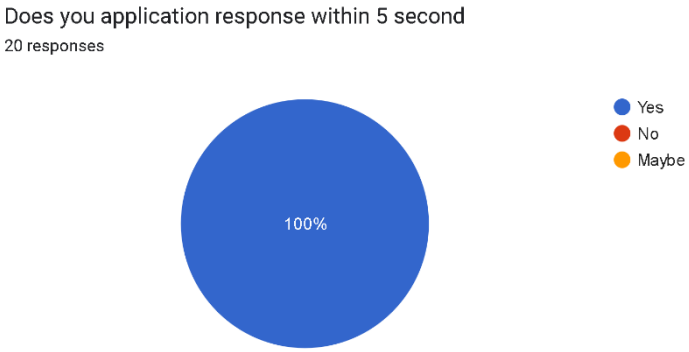


Figure 6.12: Result of Performance Question 1

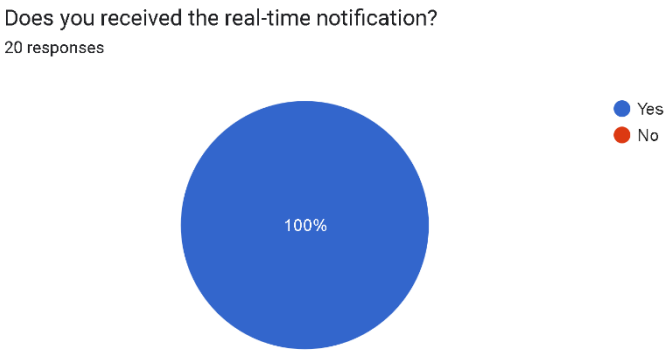


Figure 6.13: Result of Performance Question 2

Regarding the Performance, Figure 6.12 and Figure 6.13 show that the application has successfully met the requirements. Specifically, all the respondents (100%) confirmed that the application responds within 5 seconds and delivers real-time notifications as expected. This consistent positive response highlights the efficiency and effectiveness of the application in meeting performance standards.

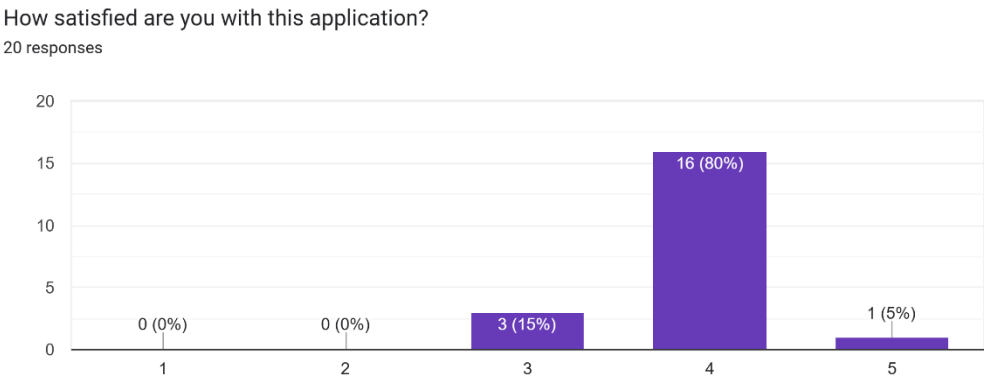


Figure 6.14: Result of Overall Satisfaction Question 1

According to the Figure 6.14, 5% of respondents are very satisfied to the application with the features, which they rated 5. The majority of respondents (80%) rated as 4, which indicates that the application is met the needs of them. Another 15% of the respondents rated as 3 to suggest that the application need to keep improvement. Overall, the feedback shows a high level of satisfaction

with the application, but there are still got opportunities to address the issues and further improve the user experience.

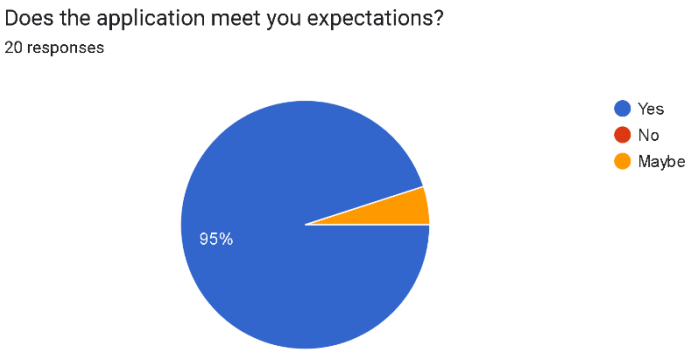


Figure 6.15: Result of Overall Satisfaction Question 2

In the response to question of Figure 6.17, most respondents (95%) stated that the application has met their expectations and needs, where 5% or respondents said that maybe some aspects or features need to improve or added to fully align with all user's expectations. This feedback is mostly positive and shows that the application is mostly effective in achieving its intended goal, with some potential for improvement to resolve any remaining issues.

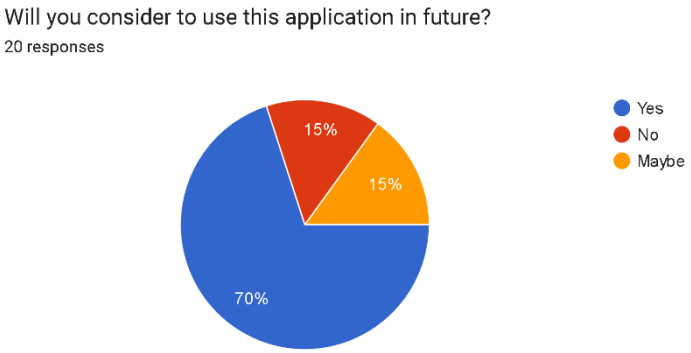


Figure 6.16: Result of Overall Satisfaction Question 3

Figure 6.16 had asking whether respondents will use this application in future, 70% of them responded "yes", which indicates a strong interest in continued use

of the application. However, 15% of respondents answered “no” maybe because they are using other parental control system, or they child does not need it anymore. Another 15% of respondents who answered “maybe” reflecting their uncertainty about future use. This feedback shows that most users are optimistic about the future potential of the application, but there are areas that need improvement to change the "maybe" and "no" feedback into positive responses.

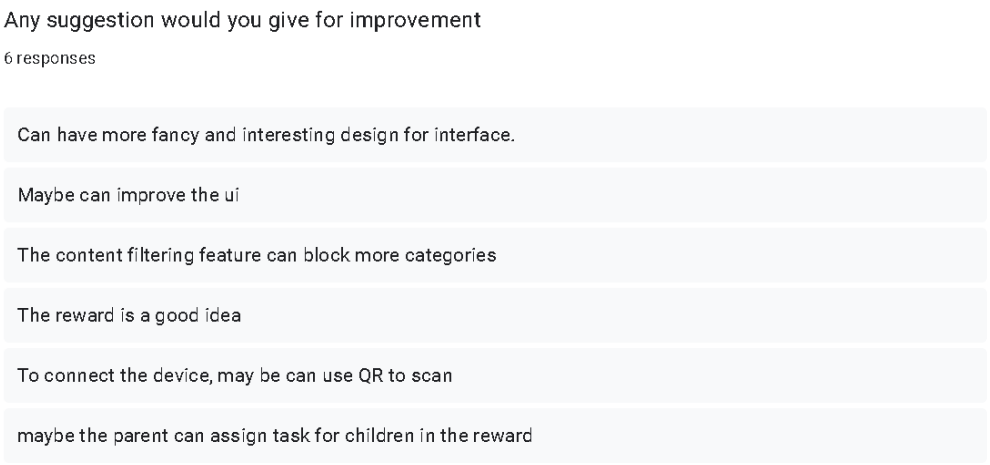


Figure 6.17: Result of Suggestion from Respondents

Figure 6.17 shows the result of suggestion from the respondents. Out of 20 respondents, 6 respondents (30%) were providing the suggestion for improvement the application. Key recommendations involve improving the interface design to be more attractive and interactive, as well as enhancing the user interface to improve usability. Users also suggested that expanding the content filtering feature by adding extra categories to enhance the level of protection. Though the reward system got a good response, there is a desire to enhance this feature further. Moreover, suggestions were made for valuable enhancements such as incorporating QR code scanning for device connection and allowing parents to assign tasks within the reward system. These recommendations point out where enhancements can be made and offer guidance for upcoming updates to enhance user satisfaction.

6.4 Summary

In summary, this section had conduct two software testing methods which are Black Box Testing as a type of Manual Unit Testing and User Acceptance Test

(UAT). Black Box Testing was used to confirm the application's functionality by testing different features without examining the internal code. This approach made sure that every feature of application, including login, signup, forget password and parental control features, worked correctly and aiding in discovering and fixing any problems linked to the app's main operations.

Besides, the User Acceptance Test (UAT) collected useful feedback from users via a questionnaire in a Google Form, focusing on usability, performance, and overall satisfaction. The response s gave important information on areas that could be enhanced, including interface design, content filtering, and the reward system. By using these software testing tools, testing procedure validated both the technical functions and ensure the application has met the user requirements, which lays the foundation for further enhancement based on the feedback.

CHAPTER 7
CONCLUSION AND RECOMMENDATIONS

7.1 Conclusion

The expected outcome of this project is to develop a remote parental control application which create a robust platform that allows parents to remote monitor and manage their children's device usage, online activities and location in real-time. The application would provide parents with tools to control screen time, apply content filtering, and track their movements to create a safer and more secure online environment for their children. During the development process, key features like set limit time, content filtering, real time location tracking, geofencing, SOS and reward are successfully implemented, providing a comprehensive solution to parental control challenges.

The project has achieved the aim and objectives by defining the requirements specification through the existing similar parental control application. It also successfully implements the functional features such as content filtering and location tracking for parents to monitor their children. Besides, the User Acceptance Test (UAT) is conducted to gather the feedback from users, ensuring that the application met their needs and identify areas for further improvement.

In conclusion, this project had met all the objectives and the project requirements, delivering a fully functional parental control application that enables parents to monitor and manage their children's online activities and device usage in real-time.

7.2 Project Challenges and Solution

There are some challenges faced during this project. First, the challenge is working with programming languages and frameworks that were not covered in syllabus. Even though the theoretical concepts were similar, applying them practically required additional learning and adaptation. In order to overcome this, I engaged in self-directed learning through online courses, documentation, and developer communities such as Google, YouTube and GitHub. This method

helps me to gain the skill and knowledge to handle these new technologies efficiently.

Besides, the project faced difficulties with hardware requirements. The large codebase and data requirements quickly filled up my disk space, causing issues with the program's performance. I transferred the project code to a desktop computer with large storage by reinstall the tools needed and dependencies to solve this problem. The change enabled me to effectively handle the code and data and ensured that I could continue development without interruptions.

7.3 Future Improvement and Recommendations

In the future, there are several areas for future improvement and enhancement in this project gained from the feedback. The expansion of content filtering categories is one of the major improvements. By adding more categories, parents will have more control over what their children can access online. This means that there are more options to block or allow certain types of content, making the filtering system more suitable for individual needs.

Next, another idea is to enhance the rewards feature. With the enhance system, parents are allowed to assign specific tasks to their children. For example, parents can add a new task named “Drawing” and set the extra time of 30 minutes. Providing this feature not only motivate children to manage their time more effectively, but also make the application more attractive and interactive.

In addition, simplify the device connect feature with QR codes would improve the usability. This feature can be like parents’ device generate a QR code for children to scan it, then it is successfully connected. This update would enable parents and children to easily and quickly link their devices, improving the user experience as a whole.

These enhancements will improve the application's functionality and ease of use, giving parents better tools to manage and monitor their children’s online activities.

REFERENCES

Anderson, M. (2019). How parents feel about – and manage – their teens' online behavior and screen time. [online] Pew Research Center. Available at: https://www.pewresearch.org/short-reads/2019/03/22/how-parents-feel-about-and-manage-their-teens-online-behavior-and-screen-time/ [Accessed 16 Feb. 2024].

Emerson, L. M., Ogielda, C., & Rowse, G. (2019). The role of experiential avoidance and parental control in the association between parent and child anxiety. Frontiers in Psychology, 10(FEB). https://doi.org/10.3389/fpsyg.2019.00262 [Accessed 18 Feb. 2024].

families.google.com. (n.d.). Family Link from Google - Family Safety & Parental Control Tools. [online] Available at: https://families.google/familylink/ [Accessed 29 Mar. 2024].

Global Newswire. "Parental Control Software Market Booming amid Growing Digital Dangers and Increased Screen Time, Report by Persistence Market Research." GlobeNewswire News Room, 16 Jan. 2024, www.globenewswire.com/en/news-release/2024/01/16/2809971/0/en/Parental-Control-Software-Market-Booming-Amid-Growing-Digital-Dangers-and-Increased-Screen-Time-Report-by-Persistence-Market-Research.html [Accessed 24 Mar. 2024].

Grossinger, Paul. "Parental Control Is Now All about Smartphones and Mobile - MMGuardian." Www.mmguardian.com, 11 Mar. 2013, www.mmguardian.com/blog/mmguardian-parental-control [Accessed 24 Mar. 2024].

HRISTOVSKA, P. (2024). Bark Parental Control Review 2023 — Is It Worth the Cost? [online] SafetyDetectives. Available at: https://www.safetydetectives.com/best-parental-control/bark/ [Accessed 30 Mar. 2024].

Inc, C.T. (2023). Emergency Phone For Your Child: Why It's Important. [online] COSMO Technologies, Inc. Available at: https://cosmotogether.com/blogs/news/emergency-phone-for-your-child-why-its-important [Accessed 18 Feb. 2024].

Ivor.Pro (2023). Protecting your children from harmful online content in 2 simple steps. [online] Medium. Available at: https://ivorpro.medium.com/the-history-of-parental-control-f48fe5dc514 [Accessed 25 Feb. 2024].

metafisica. "Parental Control Market Furnishes Information on Market Share, Market Trends, and Market Growth." Www.linkedin.com, 26 Mar. 2024, www.linkedin.com/pulse/parental-control-market-furnishes-information-share-trends-qecrf/ [Accessed 26 Mar. 2024].

Mobicip. (n.d.). Parental Control Software & Internet Filter. [online] Available at: https://www.mobicip.com/ [Accessed 28 Mar. 2024].

Qustodio. (n.d.). Best Parental Control Software. [online] Available at: https://www.qustodio.com/en/ [Accessed 28 Mar. 2024].

Shakir, U. (2022). Google's updated Family Link app can tell parents when their kids leave school. [online] The Verge. Available at: https://www.theverge.com/2022/10/18/23409786/google-family-link-update-parental-controls-location-alerts [Accessed 27 Mar. 2024].

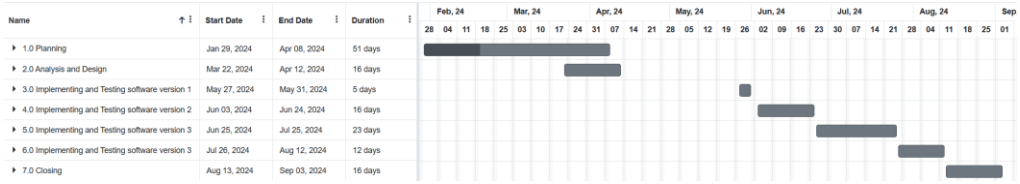
Up, E.D. (2023). Children are increasingly getting online, and at an earlier age. [online] Free Malaysia Today. Available at: https://www.freemalaysiatoday.com/category/leisure/2023/01/28/children-are-increasingly-getting-online-and-at-an-earlier-age/ [Accessed 18 Feb. 2024].

WU, R. (2024). Google Family Link Review 2024: It's Free, But Is It Good? [online] SafetyDetectives. Available at: https://www.safetydetectives.com/best-parental-control/google-family-link/#review-3 [Accessed 30 Mar. 2024].

WU, R. (2024). Mobicip Parental Control Review 2024: Is It Any Good? [online] SafetyDetectives. Available at: https://www.safetydetectives.com/best-parental-control/mobicip/ [Accessed 30 Mar. 2024].

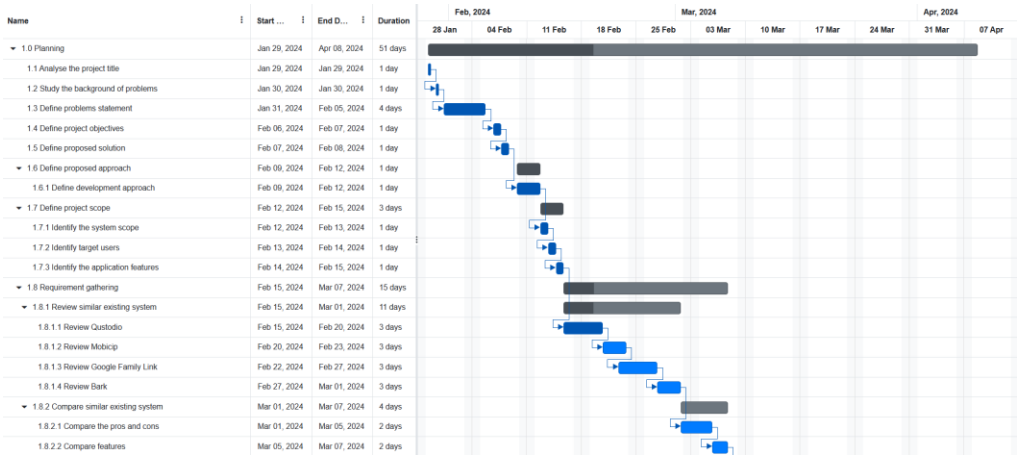
APPENDICES

Appendix A: Gantt Chart



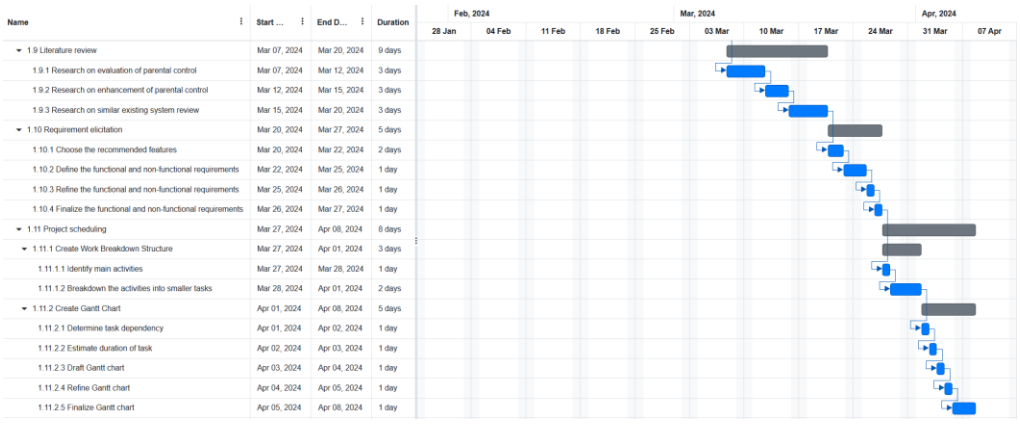
Name	Start Date	End Date	Duration
1.0 Planning	Jan 29, 2024	Apr 08, 2024	51 days
2.0 Analysis and Design	Mar 22, 2024	Apr 12, 2024	16 days
3.0 Implementing and Testing software version 1	May 27, 2024	May 31, 2024	5 days
4.0 Implementing and Testing software version 2	Jun 03, 2024	Jun 24, 2024	16 days
5.0 Implementing and Testing software version 3	Jun 25, 2024	Jul 25, 2024	23 days
6.0 Implementing and Testing software version 3	Jul 26, 2024	Aug 12, 2024	12 days
7.0 Closing	Aug 13, 2024	Sep 03, 2024	16 days

Figure A-1: Overview of the Project Schedule



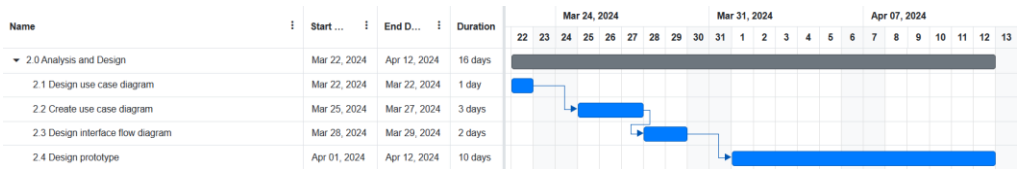
Name	Start ...	End D...	Duration
1.0 Planning	Jan 29, 2024	Apr 08, 2024	51 days
1.1 Analyse the project title	Jan 29, 2024	Jan 29, 2024	1 day
1.2 Study the background of problems	Jan 30, 2024	Jan 30, 2024	1 day
1.3 Define problems statement	Jan 31, 2024	Feb 05, 2024	4 days
1.4 Define project objectives	Feb 06, 2024	Feb 07, 2024	1 day
1.5 Define proposed solution	Feb 07, 2024	Feb 08, 2024	1 day
1.6 Define proposed approach	Feb 09, 2024	Feb 12, 2024	1 day
1.6.1 Define development approach	Feb 09, 2024	Feb 12, 2024	1 day
1.7 Define project scope	Feb 12, 2024	Feb 15, 2024	3 days
1.7.1 Identify the system scope	Feb 12, 2024	Feb 13, 2024	1 day
1.7.2 Identify target users	Feb 13, 2024	Feb 14, 2024	1 day
1.7.3 Identify the application features	Feb 14, 2024	Feb 15, 2024	1 day
1.8 Requirement gathering	Feb 15, 2024	Mar 07, 2024	15 days
1.8.1 Review similar existing system	Feb 15, 2024	Mar 01, 2024	11 days
1.8.1.1 Review Qustodio	Feb 15, 2024	Feb 20, 2024	3 days
1.8.1.2 Review Mobicip	Feb 20, 2024	Feb 23, 2024	3 days
1.8.1.3 Review Google Family Link	Feb 22, 2024	Feb 27, 2024	3 days
1.8.1.4 Review Bark	Feb 27, 2024	Mar 01, 2024	3 days
1.8.2 Compare similar existing system	Mar 01, 2024	Mar 07, 2024	4 days
1.8.2.1 Compare the pros and cons	Mar 01, 2024	Mar 05, 2024	2 days
1.8.2.2 Compare features	Mar 05, 2024	Mar 07, 2024	2 days

Figure A-2: Planning Phase Schedule



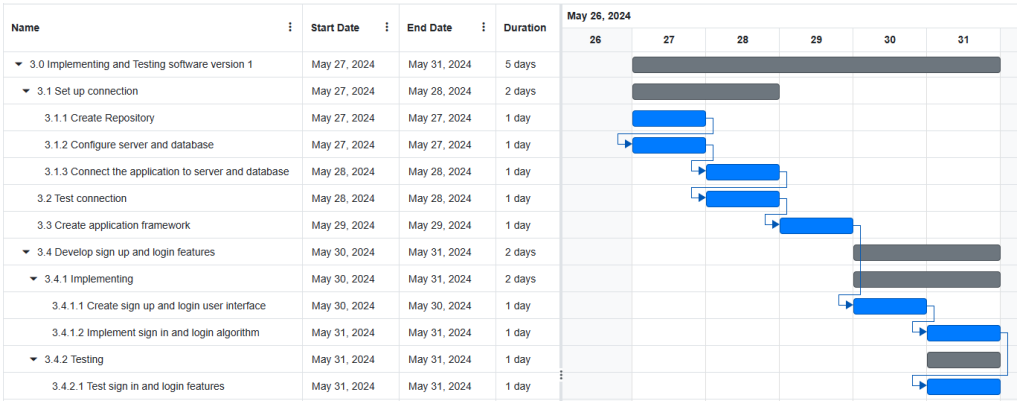
Name	Start ...	End D...	Duration
1.9 Literature review	Mar 07, 2024	Mar 20, 2024	9 days
1.9.1 Research on evaluation of parental control	Mar 07, 2024	Mar 12, 2024	3 days
1.9.2 Research on enhancement of parental control	Mar 12, 2024	Mar 15, 2024	3 days
1.9.3 Research on similar existing system review	Mar 15, 2024	Mar 20, 2024	3 days
1.10 Requirement elicitation	Mar 20, 2024	Mar 27, 2024	5 days
1.10.1 Choose the recommended features	Mar 20, 2024	Mar 22, 2024	2 days
1.10.2 Define the functional and non-functional requirements	Mar 22, 2024	Mar 25, 2024	1 day
1.10.3 Refine the functional and non-functional requirements	Mar 25, 2024	Mar 26, 2024	1 day
1.10.4 Finalize the functional and non-functional requirements	Mar 26, 2024	Mar 27, 2024	1 day
1.11 Project scheduling	Mar 27, 2024	Apr 08, 2024	8 days
1.11.1 Create Work Breakdown Structure	Mar 27, 2024	Apr 01, 2024	3 days
1.11.1.1 Identify main activities	Mar 27, 2024	Mar 28, 2024	1 day
1.11.1.2 Breakdown the activities into smaller tasks	Mar 28, 2024	Apr 01, 2024	2 days
1.11.2 Create Gantt Chart	Apr 01, 2024	Apr 08, 2024	5 days
1.11.2.1 Determine task dependency	Apr 01, 2024	Apr 02, 2024	1 day
1.11.2.2 Estimate duration of task	Apr 02, 2024	Apr 03, 2024	1 day
1.11.2.3 Draft Gantt chart	Apr 03, 2024	Apr 04, 2024	1 day
1.11.2.4 Refine Gantt chart	Apr 04, 2024	Apr 05, 2024	1 day
1.11.2.5 Finalize Gantt chart	Apr 05, 2024	Apr 08, 2024	1 day

Figure A-3: Planning Phase Schedule (cont.)



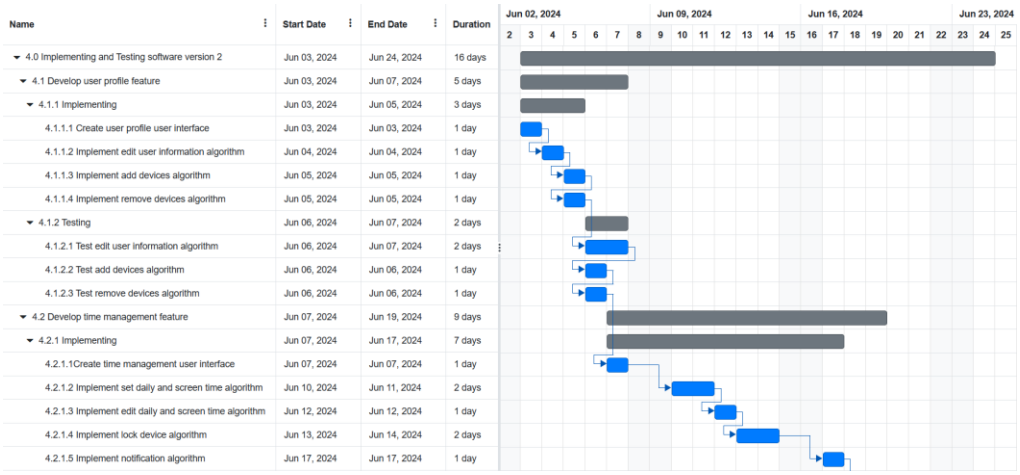
Name	Start ...	End D...	Duration
2.0 Analysis and Design	Mar 22, 2024	Apr 12, 2024	16 days
2.1 Design use case diagram	Mar 22, 2024	Mar 22, 2024	1 day
2.2 Create use case diagram	Mar 25, 2024	Mar 27, 2024	3 days
2.3 Design interface flow diagram	Mar 28, 2024	Mar 29, 2024	2 days
2.4 Design prototype	Apr 01, 2024	Apr 12, 2024	10 days

Figure A-4: Analysis and Design Phase Schedule



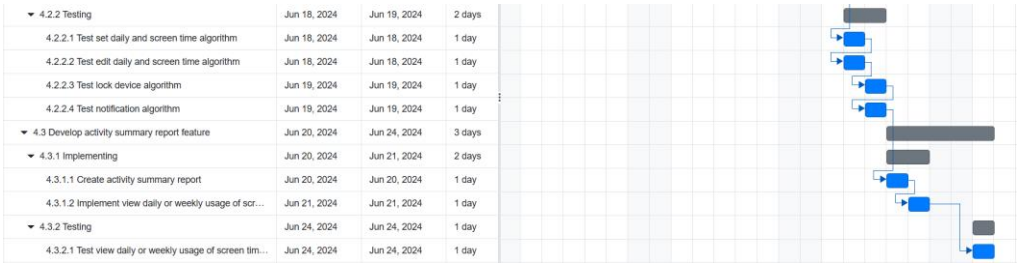
Name	Start Date	End Date	Duration
3.0 Implementing and Testing software version 1	May 27, 2024	May 31, 2024	5 days
3.1 Set up connection	May 27, 2024	May 28, 2024	2 days
3.1.1 Create Repository	May 27, 2024	May 27, 2024	1 day
3.1.2 Configure server and database	May 27, 2024	May 27, 2024	1 day
3.1.3 Connect the application to server and database	May 28, 2024	May 28, 2024	1 day
3.2 Test connection	May 28, 2024	May 28, 2024	1 day
3.3 Create application framework	May 29, 2024	May 29, 2024	1 day
3.4 Develop sign up and login features	May 30, 2024	May 31, 2024	2 days
3.4.1 Implementing	May 30, 2024	May 31, 2024	2 days
3.4.1.1 Create sign up and login user interface	May 30, 2024	May 30, 2024	1 day
3.4.1.2 Implement sign in and login algorithm	May 31, 2024	May 31, 2024	1 day
3.4.2 Testing	May 31, 2024	May 31, 2024	1 day
3.4.2.1 Test sign in and login features	May 31, 2024	May 31, 2024	1 day

Figure A-5: Implementing and Testing Software Version 1 Schedule



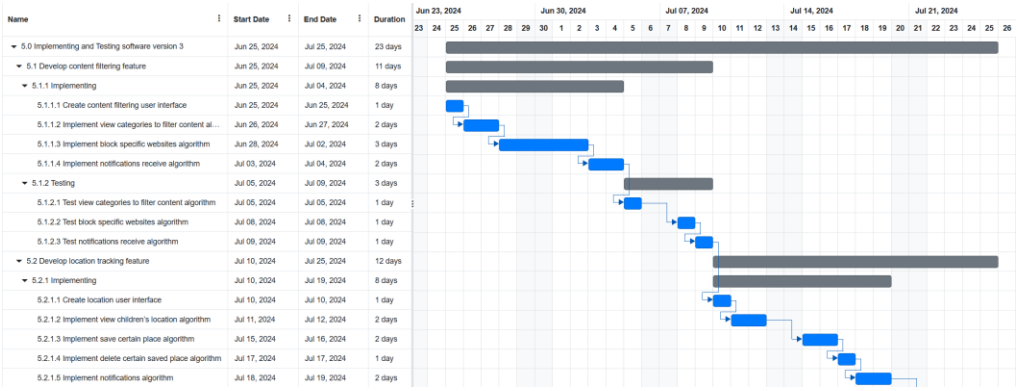
Name	Start Date	End Date	Duration
4.0 Implementing and Testing software version 2	Jun 03, 2024	Jun 24, 2024	16 days
4.1 Develop user profile feature	Jun 03, 2024	Jun 07, 2024	5 days
4.1.1 Implementing	Jun 03, 2024	Jun 05, 2024	3 days
4.1.1.1 Create user profile user interface	Jun 03, 2024	Jun 03, 2024	1 day
4.1.1.2 Implement edit user information algorithm	Jun 04, 2024	Jun 04, 2024	1 day
4.1.1.3 Implement add devices algorithm	Jun 05, 2024	Jun 05, 2024	1 day
4.1.1.4 Implement remove devices algorithm	Jun 05, 2024	Jun 05, 2024	1 day
4.1.2 Testing	Jun 06, 2024	Jun 07, 2024	2 days
4.1.2.1 Test edit user information algorithm	Jun 06, 2024	Jun 07, 2024	2 days
4.1.2.2 Test add devices algorithm	Jun 06, 2024	Jun 06, 2024	1 day
4.1.2.3 Test remove devices algorithm	Jun 06, 2024	Jun 06, 2024	1 day
4.2 Develop time management feature	Jun 07, 2024	Jun 19, 2024	9 days
4.2.1 Implementing	Jun 07, 2024	Jun 17, 2024	7 days
4.2.1.1Create time management user interface	Jun 07, 2024	Jun 07, 2024	1 day
4.2.1.2 Implement set daily and screen time algorithm	Jun 10, 2024	Jun 11, 2024	2 days
4.2.1.3 Implement edit daily and screen time algorithm	Jun 12, 2024	Jun 12, 2024	1 day
4.2.1.4 Implement lock device algorithm	Jun 13, 2024	Jun 14, 2024	2 days
4.2.1.5 Implement notification algorithm	Jun 17, 2024	Jun 17, 2024	1 day

Figure A-6: Implementing and Testing Software Version 2 Schedule



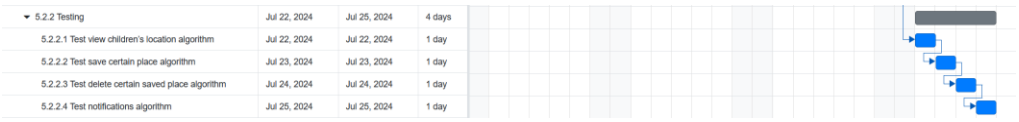
4.2.2 Testing	Jun 18, 2024	Jun 19, 2024	2 days
4.2.2.1 Test set daily and screen time algorithm	Jun 18, 2024	Jun 18, 2024	1 day
4.2.2.2 Test edit daily and screen time algorithm	Jun 18, 2024	Jun 18, 2024	1 day
4.2.2.3 Test lock device algorithm	Jun 19, 2024	Jun 19, 2024	1 day
4.2.2.4 Test notification algorithm	Jun 19, 2024	Jun 19, 2024	1 day
4.3 Develop activity summary report feature	Jun 20, 2024	Jun 24, 2024	3 days
4.3.1 Implementing	Jun 20, 2024	Jun 21, 2024	2 days
4.3.1.1 Create activity summary report	Jun 20, 2024	Jun 20, 2024	1 day
4.3.1.2 Implement view daily or weekly usage of scr...	Jun 21, 2024	Jun 21, 2024	1 day
4.3.2 Testing	Jun 24, 2024	Jun 24, 2024	1 day
4.3.2.1 Test view daily or weekly usage of screen tim...	Jun 24, 2024	Jun 24, 2024	1 day

Figure A-7: Implementing and Testing Software Version 2 Schedule (cont.)



Name	Start Date	End Date	Duration
5.0 Implementing and Testing software version 3	Jun 25, 2024	Jul 25, 2024	23 days
5.1 Develop content filtering feature	Jun 25, 2024	Jul 09, 2024	11 days
5.1.1 Implementing	Jun 25, 2024	Jul 04, 2024	8 days
5.1.1.1 Create content filtering user interface	Jun 25, 2024	Jun 25, 2024	1 day
5.1.1.2 Implement view categories to filter content al...	Jun 26, 2024	Jun 27, 2024	2 days
5.1.1.3 Implement block specific websites algorithm	Jun 28, 2024	Jul 02, 2024	3 days
5.1.1.4 Implement notifications receive algorithm	Jul 03, 2024	Jul 04, 2024	2 days
5.1.2 Testing	Jul 05, 2024	Jul 09, 2024	3 days
5.1.2.1 Test view categories to filter content algorithm	Jul 05, 2024	Jul 05, 2024	1 day
5.1.2.2 Test block specific websites algorithm	Jul 08, 2024	Jul 08, 2024	1 day
5.1.2.3 Test notifications receive algorithm	Jul 09, 2024	Jul 09, 2024	1 day
5.2 Develop location tracking feature	Jul 10, 2024	Jul 25, 2024	12 days
5.2.1 Implementing	Jul 10, 2024	Jul 19, 2024	8 days
5.2.1.1 Create location user interface	Jul 10, 2024	Jul 10, 2024	1 day
5.2.1.2 Implement view children's location algorithm	Jul 11, 2024	Jul 12, 2024	2 days
5.2.1.3 Implement save certain place algorithm	Jul 15, 2024	Jul 16, 2024	2 days
5.2.1.4 Implement delete certain saved place algorithm	Jul 17, 2024	Jul 17, 2024	1 day
5.2.1.5 Implement notifications algorithm	Jul 18, 2024	Jul 19, 2024	2 days

Figure A-8: Implementing and Testing Software Version 3 Schedule

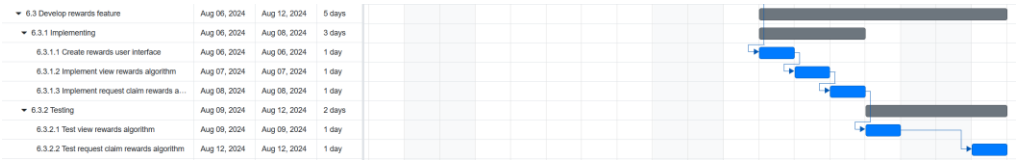


5.2.2 Testing	Jul 22, 2024	Jul 25, 2024	4 days
5.2.2.1 Test view children's location algorithm	Jul 22, 2024	Jul 22, 2024	1 day
5.2.2.2 Test save certain place algorithm	Jul 23, 2024	Jul 23, 2024	1 day
5.2.2.3 Test delete certain saved place algorithm	Jul 24, 2024	Jul 24, 2024	1 day
5.2.2.4 Test notifications algorithm	Jul 25, 2024	Jul 25, 2024	1 day

Figure A-9: Implementing and Testing Software Version 3 Schedule (cont.)

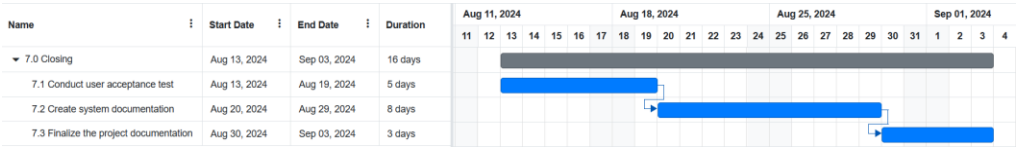
Name	Start D...	End Date	Duration
6.0 Implementing and Testing software version 3	Jul 26, 2024	Aug 12, 2024	12 days
6.1 Develop dashboard feature	Jul 26, 2024	Jul 30, 2024	3 days
6.1.1 Implementing	Jul 26, 2024	Jul 29, 2024	2 days
6.1.1.1 Create dashboard user interface	Jul 26, 2024	Jul 26, 2024	1 day
6.1.1.2 Implement view dashboard algorithm	Jul 29, 2024	Jul 29, 2024	1 day
6.1.1.3 Implement add device algorithm	Jul 29, 2024	Jul 29, 2024	1 day
6.1.2 Testing	Jul 30, 2024	Jul 30, 2024	1 day
6.1.2.1 Test view dashboard algorithm	Jul 30, 2024	Jul 30, 2024	1 day
6.1.2.2 Test add device algorithm	Jul 30, 2024	Jul 30, 2024	1 day
6.2 Develop SOS feature	Jul 31, 2024	Aug 05, 2024	4 days
6.2.1 Implementing	Jul 31, 2024	Aug 02, 2024	3 days
6.2.1.1 Create SOS user interface	Jul 31, 2024	Jul 31, 2024	1 day
6.2.1.2 Implement send SOS algorithm	Aug 01, 2024	Aug 01, 2024	1 day
6.2.1.3 Implement stop SOS algorithm	Aug 02, 2024	Aug 02, 2024	1 day
6.2.2 Testing	Aug 05, 2024	Aug 05, 2024	1 day
6.2.2.1 Test send SOS algorithm	Aug 05, 2024	Aug 05, 2024	1 day
6.2.2.2 Test stop SOS algorithm	Aug 05, 2024	Aug 05, 2024	1 day

Figure A-10: Implementing and Testing Software Version 4 Schedule



6.3 Develop rewards feature	Aug 06, 2024	Aug 12, 2024	5 days
6.3.1 Implementing	Aug 06, 2024	Aug 08, 2024	3 days
6.3.1.1 Create rewards user interface	Aug 06, 2024	Aug 06, 2024	1 day
6.3.1.2 Implement view rewards algorithm	Aug 07, 2024	Aug 07, 2024	1 day
6.3.1.3 Implement request claim rewards a...	Aug 08, 2024	Aug 08, 2024	1 day
6.3.2 Testing	Aug 09, 2024	Aug 12, 2024	2 days
6.3.2.1 Test view rewards algorithm	Aug 09, 2024	Aug 09, 2024	1 day
6.3.2.2 Test request claim rewards algorithm	Aug 12, 2024	Aug 12, 2024	1 day

Figure A-11: Implementing and Testing Software Version 4 Schedule (cont.)



Name	Start Date	End Date	Duration
7.0 Closing	Aug 13, 2024	Sep 03, 2024	16 days
7.1 Conduct user acceptance test	Aug 13, 2024	Aug 19, 2024	5 days
7.2 Create system documentation	Aug 20, 2024	Aug 29, 2024	8 days
7.3 Finalize the project documentation	Aug 30, 2024	Sep 03, 2024	3 days

Figure A-12: Closing

Appendix B: Google Survey Form

(https://forms.gle/uN1YWdk73UFLPgwL9)

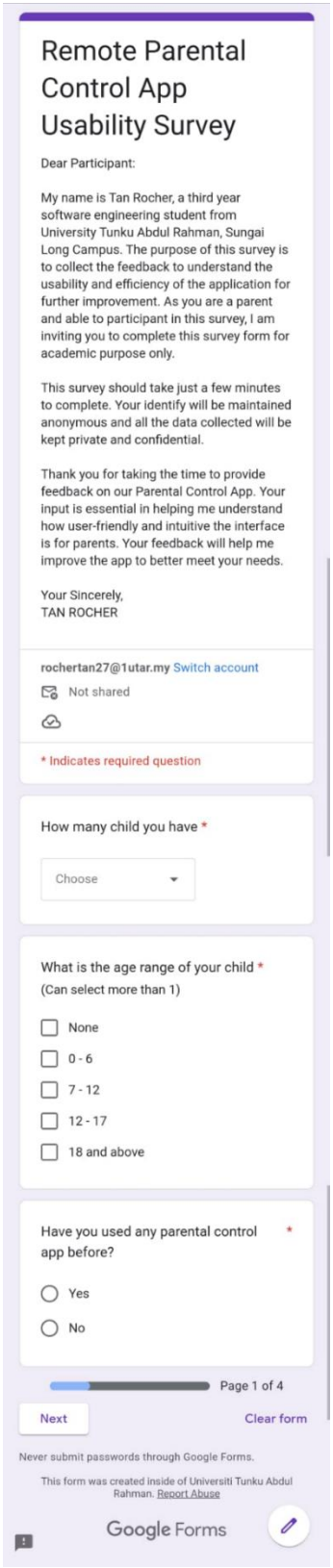


Figure B-1: Questionnaire Section 1

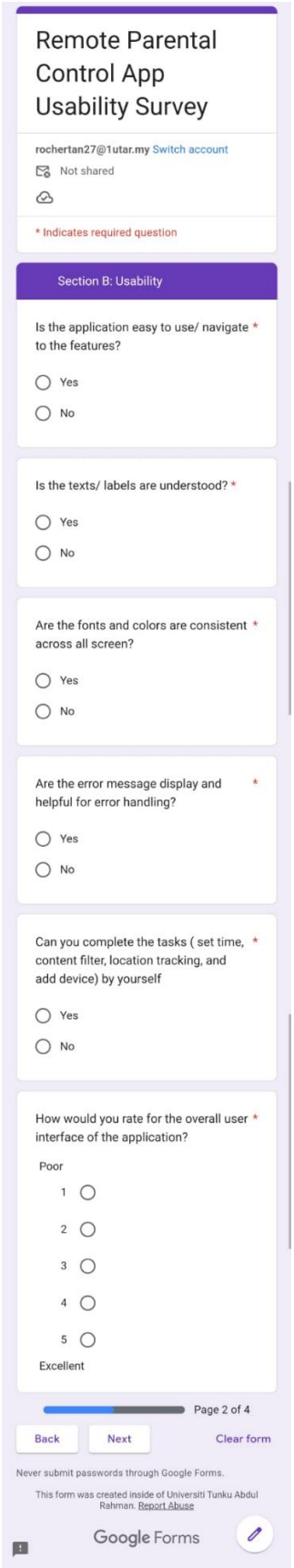


Figure B-2: Questionnaire Section 2

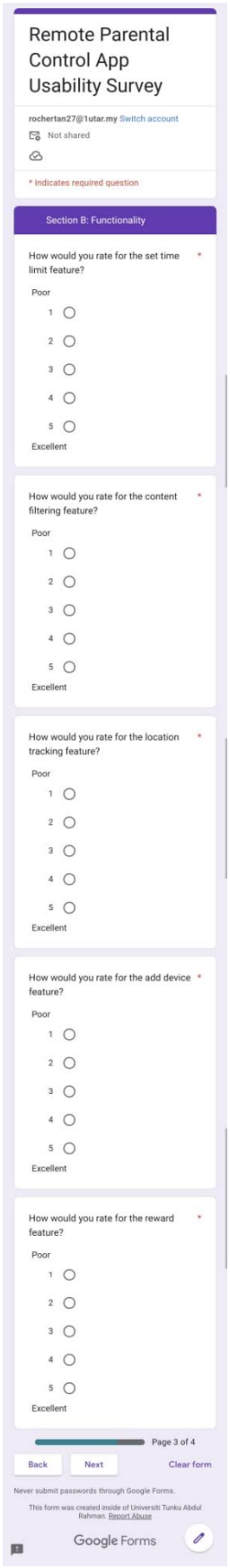


Figure B-3: Questionnaire Section 3

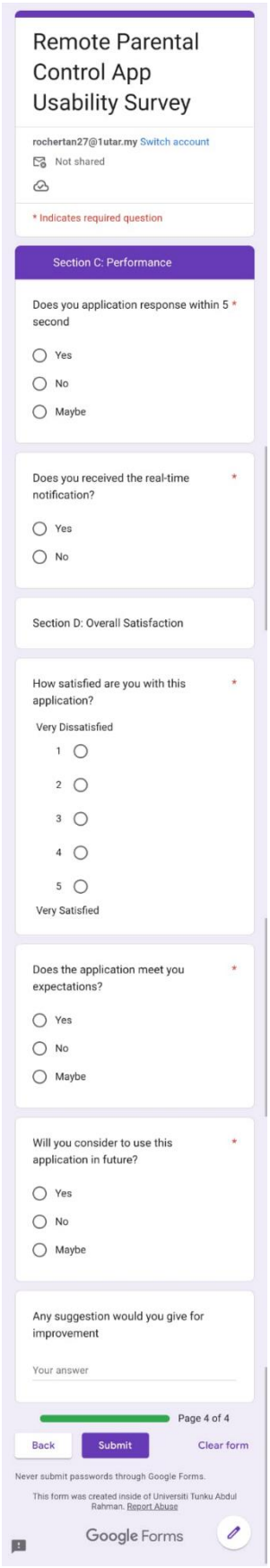


Figure B-4: Questionnaire Section 4

Appendix C: Test Results

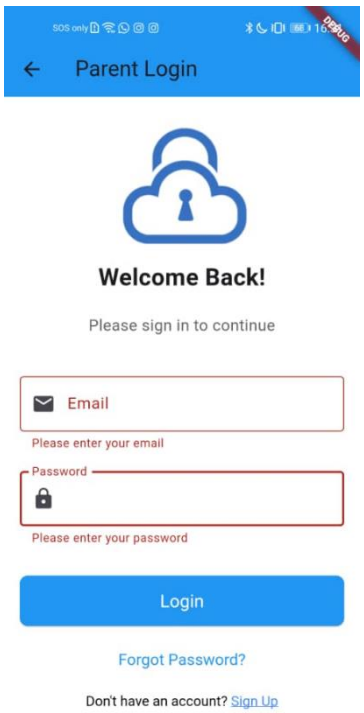


Figure C-1: Error Message of Empty Input for Parent Login

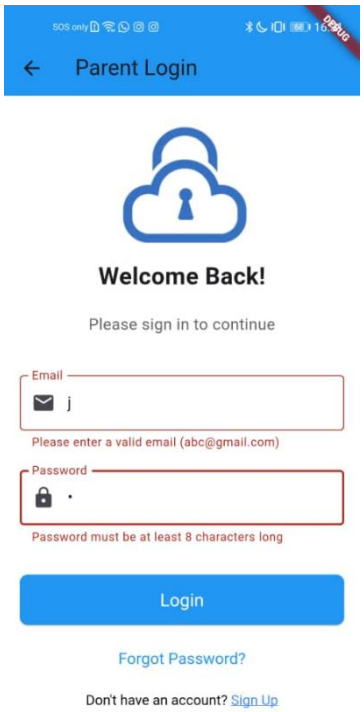


Figure C-2: Error Message of Invalid Input for Parent Login

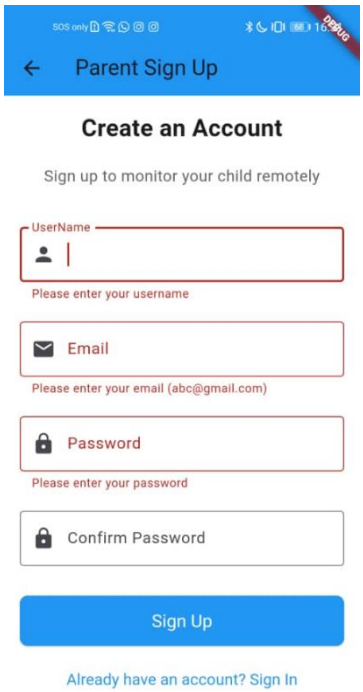


Figure C-3: Error Message of Empty Input for Parent Signup

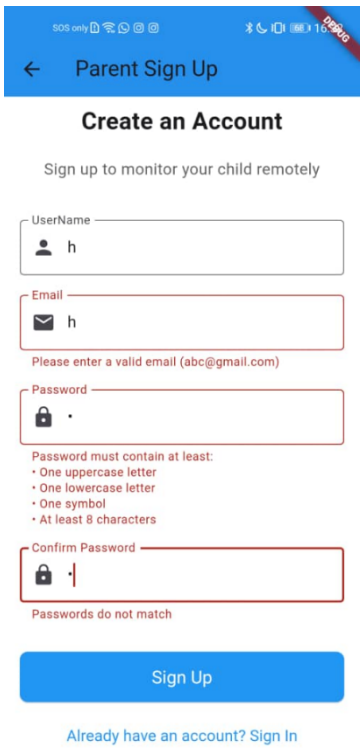


Figure C-4: Error Message of Invalid Input for Parent Signup

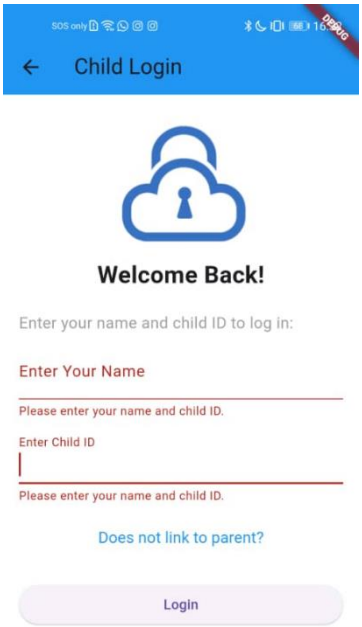


Figure C-5: Error Message of Empty Input for Child Login

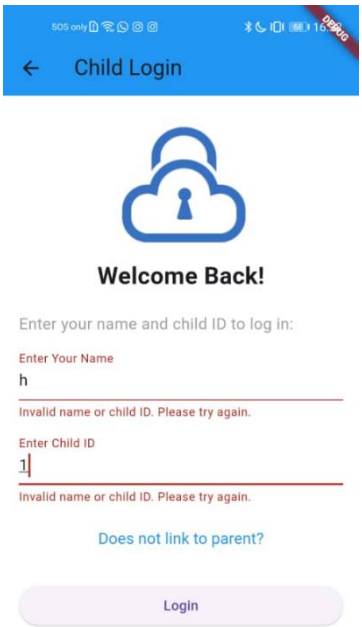


Figure C-6: Error Message of Invalid Input for Child Login

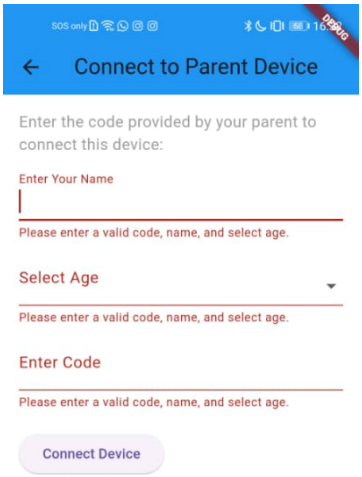


Figure C-7: Error Message of Empty Input for Child Connect Link

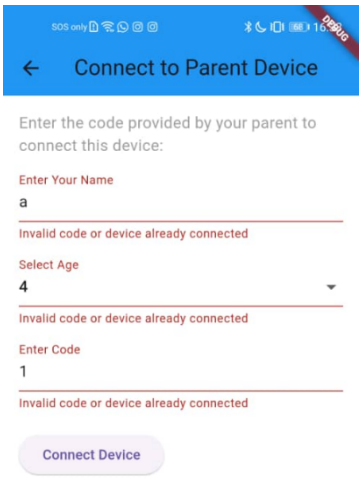


Figure C-6: Error Message of Invalid Input for Child Connect Link