SECURING DOCUMENTS WITH LIGHTWEIGHT WEB-BASED FILE ENCRYPTION APPLICATION

WAN NORAZLIN BINTI ABDULLAH

FACULTY OF COMPUTING AND INFORMATICS
UNIVERSITI MALAYSIA SABAH
2022

SECURING DOCUMENTS WITH LIGHTWEIGHT WEB-BASED FILE ENCRYPTION APPLICATION

WAN NORAZLIN BINTI ABDULLAH

THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE OF BACHELOR OF COMPUTER SCIENCE WITH HONOURS (SOFTWARE ENGINEERING)

FACULTY OF COMPUTING AND INFORMATICS
UNIVERSITI MALAYSIA SABAH
2022

NAME	: WAN NORAZLIN BINTI ABDULLAH
MATRIC NUMBER	: BI18110278
TITLE	: SECURING DOCUMENTS WITH LIGHTWEIGHT WEB -BASED ENCRYPTION APPLICATION
DEGREE	: BACHELOR OF COMPUTER SCIENCE WITH HONOURS (SOFTWARE ENGINEERING)
VIVA'S DATE	: 25 JANUARY 2022

CERTIFIED BY;

Signature

1. SUPERVISOR

DR TAN SOO FUN

DR TAN SOO FUN
SENIOR LECTURER
Faculty of Computing and Informatics
Universiti Malaysia Sabah

DECLARATION

I hereby declare that the material in this thesis is my own except for quotations, equations, summaries and references, which have been duly acknowledged.

25 JANUARY 2022

WAN NORAZLIN BINTI ABDULLAH

BI182110278

ACKNOWLEDGEMENT

First and foremost, I would like to express my gratitude to my supervisor, Dr Tan Soo Fun, for the continuous guidance and support during the Final Year Project. Her guidance helped me through the complicated parts of the project, and I am truly thankful for her utmost patience, motivation, and enthusiasm every time I approached her for any advice. I could not have imagined having a better supervisor for my degree study.

I would also like to thank my examiners, Dr Leau Yu Beng and Dr Kashif Nisar, for their insightful feedbacks on every presentation done, as it had helped me a lot in understanding and completing my project.

I would like to say my sincere thanks to all the lecturers that had participated in my questionnaires and software testing. Without them, I would not be able to move forward in the project and develop the system.

Not to forget, my fellow colleagues that has always helped me, no matter what issue it is. Thank you for always giving me the emotional support during this journey and accompanying me during my lowest. I appreciate and absolutely cherish all the humorous and wholesome conversations we had to support each other, especially during the extremely nerve-wrecking moments.

Finally, I would like to say my utmost gratitude to my parents, Norshahizan Bin Baharudin and Halimah Binti Sura Talib, for the never-ending support, love, and motivation they gave me, ever since I was born. Thank you for understanding me and allowing me to achieve what I wanted or dreamed of, including having a lot of cats. My cats had also given me tons of support surprisingly. Their presence had made this journey a whole lot bearable, by giving me warmth and comfort during my lowest days.

Wan Norazlin Binti Abdullah

25 JANUARY 2022

ABSTRACT

The Movement Control Order (MCO) and work from home practices due to the COVID-19 pandemic resulted in a significant increase use of online file sharing and storage services. Data security becomes a concern as the actions currently taken to secure data are more towards data-in-transmission, such as the "SSL/TLS Handshake" that ensures a certified and secured peer transmission. The lack of security for data-at-rest in storage services however has resulted in data breaches caused by unauthorized access. Hence, this project proposes a lightweight web-based encryption tool as a solution to protect the confidentiality and integrity of cloud storage. The methodology applied in this project is the waterfall approach, will comprises seven phases. Lightweight encryption algorithms are investigated by benchmarking their performance in terms of computation speed and ciphertext size. After identifying the best algorithm, a lightweight web-based encryption application with PHP programming and JavaScript that protects the confidentiality and integrity of cloud storage is designed and developed. The web-based encryption application is then evaluated for its performance by using the System Usability Scale (SUS) approach.

ABSTRAK

(MELINDUNGI DOKUMEN DENGAN APLIKASI WEB PENYULIPAN FAIL RINGAN)

Perintah Kawalan Pergerakan (PKP) dan amalan bekerja dari rumah (atau mudah dikenali sebagai 'Work from Home (WFH)') yang disebabkan oleh COVID-19 pandemik telah mengakibatkan peningkatan signifikan dalam penggunaan perkhidmatan perkongsian dan penyimpanan fail dalam talian. Sekuriti data menjadi satu kebimbangan kerana Langkah-langkah terkini dalam memastikan keselamatan data adalah lebih menjurus kepada data dalam peghantaran, seperti "SSL/TLS Handshake" yang memastikan bahawa data dihantar dalam keadaan yang disahkan selamat. Kekurangan sekuriti untuk data di dalam perkhidmatan file storan telah mengakibatkan jumlah kebocoran data disebabkan akses yang tidak sah. Maka, projek ini mencadangkan satu web aplikasi ringan enkripsi sebagai penyelesaian bagi melindungi kerahsiaan dan integriti storan atas talian. Metodologi yang digunakan dalam projek ini ialah 'waterfall approach', yang merangkumi tujuh fasa. Penyelidikan dijalankan terhadap algoritma kriptografi yang ringan dengan menjalankan eksperimen untuk mengukur kelajuan komputasi dan saiz teks sifer. Selepas mengenalpasti algoritma yang terbaik, web aplikasi enkripsi yang ringan akan direka dan dibangunkan menggunakan PHP dan JavaScript untuk melindungi kerahsiaan dan integriti storan atas talian. Web aplikasi enkripsi tersebut akan dinilai prestasinya menggunakan pendekatan System Usability Scale (SUS).

LIST OF CONTENTS

	Page
TITLE	i
DECLARATION	iv
ACKNOWLEDGEMENT	v
ABSTRACT	vi
ABSTRAK	vii
LIST OF CONTENTS	viii
LIST OF TABLES	xii
LIST OF FIGURES	xiv
LIST OF APPENDICES	xx
CHAPTER 1: INTRODUCTION	
1.1 Introduction	1
1.2 Problem Background	2
1.3 Problem Statement	3
1.4 Problem Objectives	4
1.5 Project Scope	4
1.6 Organization of the Report	8
1.7 Conclusion	10
CHAPTER 2: LITERATURE REVIEW	
2.1 Lightweight Symmetric Encryption Algorithms	11
2.1.1 Twofish	12
2.1.2 XTEA	13
2.1.3 Twine	15
2.1.4 LEA	16
2.1.5 LED	18
2.1.6 Comparison of Lightweight Symmetric Encryption Algorithm	19
2.2 Existing Web-based Encryption Tools	24

2.2.1 Free Crypt 24
2.2.2 TxtWizard 34
2.2.3 ENCRYPT.One 37
2.2.4 EncDec 41
2.2.5 Devglan 43
2.2.6 Comparison of Existing Web-based Encryption Tools 46

CHAPTER 3: METHODOLOGY
3.1 Introduction 52
3.2 Methodology 56
3.3 Conclusion 57

CHAPTER 4: SYSTEM ANALYSIS AND DESIGN
4.1 Introduction 58
4.2 Data Collections 58
4.3 Questionnaire Findings 58
4.3.1 Respondents Experience with Web-based Encryption Tools 59
4.4 System Design 65
4.4.1 Entity Relationship Diagram (ERD) 66
4.4.2 Data Flow Diagram (DFD) 66
4.4.2.1 Context Diagram 67
4.4.2.2 Data Flow Diagram (Level 0) 68
4.4.2.2 Data Flow Diagram (Level 1 1.0) 69
4.4.2.2 Data Flow Diagram (Level 1 2.0) 70
4.4.2.2 Data Flow Diagram (Level 1 3.0) 71
4.4.2.2 Data Flow Diagram (Level 1 4.0) 72
4.4.2.2 Data Flow Diagram (Level 1 5.0) 73
4.4.2.2 Data Flow Diagram (Level 1 6.0) 74
4.4.2.2 Data Flow Diagram (Level 1 7.0) 75
4.4.2.2 Data Flow Diagram (Level 1 8.0) 76
4.4.2.2 Data Flow Diagram (Level 1 9.0) 77

4.4.2.2 Data Flow Diagram (Level 1 10.0) 78
4.4.2.2 Data Flow Diagram (Level 1 11.0) 79
4.4.2.2 Data Flow Diagram (Level 1 12.0) 80
4.4.2.2 Data Flow Diagram (Level 1 13.0) 81
4.4.3 Data Dictionary 82
4.4.4 User Interface Design 86
4.5 Conclusion 106

CHAPTER 5: EXPERIMENT IMPLEMENTATION
5.1 Introduction 107
5.2 Dataset and Parameters Setting 107
5.3 Algorithm Implementation 109
5.3.1 Twofish Algorithm Implementation 110
5.3.2 LEA Algorithm Implementation 111
5.3.3 XTEA Algorithm Implementation 112

CHAPTER 6: DISCUSSION
6.1 Introduction 113
6.2 Discussion 113

CHAPTER 7: IMPLEMENTATION
7.1 Introduction 124
7.2 Implementation 124
7.2.1 Development environment 124
7.2.2 Components 125

CHAPTER 8: PRELIMINARY TESTING
8.1 Introduction 143
8.2 Unit Testing 143
8.2.1 User Registration and Authentication 144
8.2.2 User Account Management 154
8.2.3 Email Notification 170

8.2.4 Web-based Encryption Tool 174
8.2.5 Contact List and Access Policy 184
8.2.6 File Sharing Management 187
8.3 Integration Testing 190
8.3.1 User Account Registration and Verification 191
8.3.2 User Account Management 194
8.3.3 File Management 197

CHAPTER 9: SYSTEM USABILITY SCALE (SUS) TESTING
9.1 Introduction 201
9.2 Data Collections 201
9.3 Questionnaire Findings 202
9.3.1 SUS Score 202

CHAPTER 10: CONCLUSION
10.1 Summary of Project 205
10.2 Constraints 206
10.3 Future Works 207

REFERENCES 208
APPENDICES 214

LIST OF TABLES

			Page
Table 1.1	:	Project Modules	6
Table 2.1	:	Comparison of lightweight symmetric encryption algorithm	19
Table 2.2	:	Comparison of five web-based encryption tools	46
Table 3.1	:	Minimal desktop specifications for experiment implementation	53
Table 4.1	:	Data Dictionary for useraccount table	81
Table 4.2	:	Data Dictionary for filestorage table	82
Table 4.3	:	Data Dictionary for fileshare table	83
Table 4.4	:	Data Dictionary for reset_pwd table	83
Table 4.5	:	Data Dictionary for temp_otp table	84
Table 4.6	:	Data Dictionary for userfriends table	84
Table 4.7	:	Data Dictionary for verify_email table	85
Table 4.8	:	List of features/tasks of each user type	85
Table 5.1	:	Dataset involved in the experiment	107
Table 5.2	:	Specifications of machine being used for Experimentation	107
Table 5.3	:	Encryption algorithm being tested	108
Table 8.1	:	Positive Test Case for Registration Component	143
Table 8.2	:	Negative Test Case for Registration Component	145
Table 8.3	:	Positive Test Case for Login Component	149
Table 8.4	:	Negative Test Case for Login Component	151
Table 8.5	:	Positive Test Case for Updating User Details	153
Table 8.6	:	Negative Test Case for Updating User Details	157
Table 8.7	:	Positive Test Case for Reset Password	164
Table 8.8	:	Negative Test Case for Reset Password	167
Table 8.9	:	Positive Test Case for Email Verification	169
Table 8.10	:	Negative Test Case for Email Verification	171
Table 8.11	:	Positive Test Case for Web Encryption	173

Table 8.12	:	Positive Test Case for Web Encryption	175
Table 8.13	:	Negative Test Case for Web Encryption	177
Table 8.14	:	Negative Test Case for Web Encryption	179
Table 8.15	:	Positive Test Case for Web Decryption	181
Table 8.16	:	Negative Test Case for Web Decryption	182
Table 8.17	:	Positive Test Case for Add Friend	183
Table 8.18	:	Negative Test Case for Add Friend	184
Table 8.19	:	Positive Test Case for Accept Friend	185
Table 8.20	:	Positive Test Case for Share File	186
Table 8.21	:	Negative Test Case for Share File	188
Table 8.22	:	Integration testing for user account creation	190
Table 8.23	:	Integration testing for user update process	193
Table 8.24	:	Integration testing for file management process	196
Table 9.1	:	Scoring SUS	201
Table 9.2	:	SUS Score Calculations	202

LIST OF FIGURES

			Page
Figure 2.1	:	Twofish algorithm	12
Figure 2.2	:	XTEA algorithm	14
Figure 2.3	:	Source code for encryption and decryption in XTEA	15
Figure 2.4	:	TWINE algorithm	16
Figure 2.5	:	LEA algorithm	17
Figure 2.6	:	Use of subkeys in LED algorithm during steps	18
Figure2.7	:	Free Crypt homepage	25
Figure 2.8	:	The two options under 'Encrypt'	25
Figure 2.9	:	The page to download or share encrypted file	26
Figure 2.10	:	File sharing web-tool provided by Free Crypt	27
Figure 2.11	:	Page for sharing the link	27
Figure 2.12	:	Password protection feature and file sharing through email	28
Figure 2.13	:	Display of 'Encryption for Pro's'	29
Figure 2.14	:	Decryption page	29
Figure 2.15	:	Decrypted file ready to be downloaded	30
Figure 2.16	:	Error message display	30
Figure 2.17	:	Homepage of Secure Free Crypt	31
Figure 2.18	:	Scripts in the web-tool for full client-side encryption	32
Figure 2.19	:	Page for uploading file to be encrypted	32
Figure 2.20	:	Page for setting up passphrase as key	33
Figure 2.21	:	Page for downloading the encrypted file	33
Figure 2.22	:	Homepage of TxtWizard	34
Figure 2.23	:	Cryptography page of the web-tool	35
Figure 2.24	:	Encryption feature	36
Figure 2.25	:	Decryption feature	36

Figure 2.26	:	Homepage of ENCRYPT.One	38
Figure 2.27	:	Encryption feature	38
Figure 2.28	:	Link that can be shared and generated for access to decrypt file	39
Figure 2.29	:	Decryption feature	39
Figure 2.30	:	The same link accessed but in different tab browsers	40
Figure 2.31	:	Encryption without secret passphrase	40
Figure 2.32	:	Encryption with secret passphrase	41
Figure 2.33	:	Homepage of EncDec web-tool	42
Figure 2.34	:	The limit size of data for file	42
Figure 2.35	:	The limit size of data for text	43
Figure 2.36	:	Homepage Devglan encryption tool	44
Figure 2.37	:	Encryption and decryption feature for text	45
Figure 2.38	:	Encryption and decryption feature for image file	45
Figure 3.1	:	Methodology diagram	56
Figure 4.1	:	Respondents that have used web-encryption tools	59
Figure 4.2	:	Web Browser Based File Encryption/Decryption by MTI	60
Figure 4.3	:	File type that is often encrypted by respondents	61
Figure 4.4	:	Average time taken by respondents to encrypt files	61
Figure 4.5	:	Cloud storage services often used by respondents	62
Figure 4.6	:	Limitations in web-based encryption tools	63
Figure 4.7	:	Expected features in web-based encryption tool	64
Figure 4.8	:	Expected performance of encryption and decryption service	64
Figure 4.9	:	The ERD of the system	66
Figure 4.10	:	The Content Diagram	67
Figure 4.11	:	DFD Level 0	68
Figure 4.12	:	DFD Level 1 1.0 Register account	69
Figure 4.13	:	DFD Level 1 2.0 Login into account	70

Figure 4.14	:	DFD Level 1 3.0 Manage user details	71
Figure 4.15	:	DFD Level 1 4.0 Encryption using Lightweight Encryption Algorithm	72
Figure 4.16	:	DFD Level 1 5.0 Decryption using Lightweight Encryption Algorithm	73
Figure 4.17	:	DFD Level 1 6.0 Download encrypted file	74
Figure 4.18	:	DFD Level 1 7.0 Search users	75
Figure 4.19	:	DFD Level 1 8.0 Add users as contacts	76
Figure 4.20	:	DFD Level 1 9.0 Share File	77
Figure 4.21	:	DFD Level 1 10.0 Delete File	78
Figure 4.22	:	DFD Level 1 11.0 View File	79
Figure 4.23	:	DFD Level 1 12.0 Manage Sharing Session	80
Figure 4.24	:	DFD Level 1 13.0 Email Notification	81
Figure 4.25	:	Logo of the Web Application	87
Figure 4.26	:	Landing page of Horusdoc	88
Figure 4.27	:	Quick Introduction of Web Application	88
Figure 4.28	:	Website footer	89
Figure 4.29	:	'About Page' page	89
Figure 4.30	:	Registration page	90
Figure 4.31	:	Input validation if input elements are empty	90
Figure 4.32	:	Input validation when user submits empty form	91
Figure 4.33(a)	:	Input validation message if email is registered	91
Figure 4.33(b)	:	Input validation message if email is registered	91
Figure 4.34	:	Error message from server-side	91
Figure 4.35	:	Confirmation message of user registration	92
Figure 4.36	:	Login page	92
Figure 4.37	:	Homepage of user account	93
Figure 4.38	:	Search results of a user	93
Figure 4.39	:	Button change after user send friend request	94
Figure 4.40	:	Friend request page	94
Figure 4.41	:	Accepted friend request page	94
Figure 4.42	:	Profile page of user account	94

Figure 4.43	:	File encryption page	96
Figure 4.44	:	File decryption page	96
Figure 4.45	:	Modal prompt user to enter key for decryption	97
Figure 4.46	:	Error alert after invalid key submitted	97
Figure 4.47	:	Download prompt after successful decryption	97
Figure 4.48	:	Display of user's friends	98
Figure 4.49	:	Page to manage stored encrypted files	99
Figure 4.50	:	Prompt user to choose contacts to share file	99
Figure 4.51(a)	:	Tick box appearance once users tick it	100
Figure 4.51(b)	:	Tick box appearance once users tick it	100
Figure 4.52	:	Prompt to submit access password/phrase	100
Figure 4.53	:	Confirmation message for email notification	101
Figure 4.54	:	Delete file modal	101
Figure 4.55	:	Shared Files Paged	102
Figure 4.56	:	Modal that prompts access phrase/password	102
Figure 4.57	:	Error message for invalid access phrase	103
Figure 4.58	:	Modal for users to view/download the file	103
Figure 4.59	:	Settings page	104
Figure 4.60	:	Page to change username	104
Figure 4.61	:	Update email address page	105
Figure 4.62	:	Update secret phrase page	105
Figure 5.1	:	Initialization of key length and IV for Twofish	110
Figure 5.2	:	Twofish class through CBC Mode function for encryption	110
Figure 5.3	:	Twofish class through CBC Mode function for decryption	110
Figure 5.4	:	Initialization of key length and IV for LEA	111
Figure 5.5	:	LEA class through CBC Mode function for encryption	111
Figure 5.6	:	LEA class through CBC Mode function	111

		for decryption	
Figure 5.7	:	Initialization of key length and IV for XTEA	112
Figure 5.8	:	XTEA class through CBC Mode function for encryption	112
Figure 5.9	:	XTEA class through CBC Mode function for decryption	112
Figure 6.1	:	Wall time of encryption using 128-key length	115
Figure 6.2	:	Wall time of encryption using 192-key length	116
Figure 6.3	:	Wall time of encryption using 256-key length	116
Figure 6.4	:	CPU time of encryption using 128-key length	117
Figure 6.5	:	CPU time of encryption using 192-key length	117
Figure 6.6	:	CPU time of encryption using 256-key length	118
Figure 6.7	:	Wall time of decryption using 128-key length	119
Figure 6.8	:	Wall time of decryption using 192-key length	120
Figure 6.9	:	Wall time of decryption using 256-key length	120
Figure 6.10	:	CPU time of decryption using 128-key length	121
Figure 6.11	:	CPU time of decryption using 192-key length	121
Figure 6.12	:	CPU time of decryption using 256-key length	122
Figure 6.13	:	Encrypted files after encryption	122
Figure 7.1	:	Landing page of the project	125
Figure 7.2	:	The registration form on registration page	126
Figure 7.3	:	Demonstration of the input fields in the registration form when empty	127
Figure 7.4	:	Demonstration of the empty input fields in the registration form when submitted	127
Figure 7.5	:	Username validity check	128
Figure 7.6	:	Email validity check	128
Figure 7.7	:	The process of reCAPTCHA puzzle	128
Figure 7.8	:	The error messages produced by reCAPTCHA	129
Figure 7.9	:	The confirmation messages produced by reCAPTCHA	129

Figure 7.10	:	Email verification sent to user's inbox	130
Figure 7.11	:	Alert message upon successful email verification	130
Figure 7.12	:	Confirmation message after successful alert message	130
Figure 7.13	:	Status of user's email verification being updated	131
Figure 7.14	:	The login page	131
Figure 7.15	:	The page for password resetting	132
Figure 7.16	:	Password reset link sent to user's email inbox	132
Figure 7.17	:	Alert message if the verify tokens match	133
Figure 7.18	:	Password reset page	133
Figure 7.19	:	Alert message for successful password reset	134
Figure 7.20	:	The landing page for user's account	134
Figure 7.21	:	The 'Settings' page	135
Figure 7.22	:	Page for updating username	136
Figure 7.23	:	Page for updating email	136
Figure 7.24	:	Page for changing password	136
Figure 7.25	:	The OTP (One-Time-Password) process	137
Figure 7.26	:	Deactivate account page	137
Figure 7.27	:	Flowchart of encryption	139
Figure 7.28	:	Flowchart of decryption	140
Figure 9.1	:	The feedback left by respondents	204

LIST OF APPENDICES

			Page
Appendix A	:	Feedback to Proposal Review	214
Appendix B	:	Questionnaire about Web-Based Encryption Tool	217
Appendix C	:	Feedback to FYP Progress	221
Appendix D	:	Results from Experiment Implementation	223
Appendix E	:	Feedback to FYP I	227
Appendix F	:	SUS Testing Questions	228
Appendix G	:	Instruction Sheet for UAT	229

CHAPTER 1

INTRODUCTION

1.1 Introduction

The world has been struck by the news of COVID-19 (Coronavirus Disease-19) in late December 2019, which was later declared as a pandemic outbreak by the World Health Associations (WHO) in March 2020 (WHO, 2020). In Malaysia, one of the initiatives taken to reduce the cases is to enforce partial lockdown or better known as Movement Control Order (MCO), where the public is required to oblige the standard operating procedures (SOP) stated by Majlis Keselamatan Negara (MKN) and practice preventive measures (MKN, 2020).

Due to the circumstances, a lot of activities such as education and work are being done online. Working from home (WFH) and online learning has been the adaptation for a lot of employees and students worldwide. WFH is where employees move their workplace from an office to their home. Online learning is education being done fully online. Both adaptations require people to practice online file sharing, which is a service that stores and access information in the cloud. Storage services are often the medium used for file sharing, as it is affordable and readily accessible from any device online (Abdalla and Varol, 2019).

The increased demands for storage services raise concerns about security risks and data breaches. Companies are unable to ensure the security level of their employees' devices and internet connection hence sensitive data are prone to be leaked and accessed by unauthorized persons (Hayes & LLP, 2020).

Since companies rely on storage service providers in terms of data security, the data is susceptible to unwanted alteration. For example, a school in Nevada experienced a cloud document storage issue that is suspected to be caused by

misconduct, which affected the staff and teachers (News 4 and FOX 11 Digital Team, 2021). Another recent incident happened with Singapore's Singtel where 129,000 customers' personal information is stolen after a recent data breach of a third-party file-sharing system (The Economic Times, 2021).

Hence, securing data in storage services is important to avoid any loss of user's personal information. Especially towards companies where maintaining trust with customers is an utmost priority.

1.2 Problem Background/ Motivation

Cloud storage service providers are companies that offer cloud computing services which are Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) that are accessible through the Internet (Goundar, 2017). Recently, cloud storage has become more alternative for online file storing and sharing by both online users and organizations. The public cloud computing market worldwide reached around 257.5 billion U.S dollars in 2020, partially caused by the sudden pandemic that caused many to work from home (Statista, 2021).

Due to this increased usage of cloud storage, data security has become a debate among people. Some recent solutions being done by Cloud Service Providers (CSP) to ensure data security is by implementing SSL/TLS protocol to ensure a secure network transmission between user and CSP servers. This protocol ensures CSP being in a secure peer transmission with the user (Satapathy and Livingston, 2016). Besides that, CSP like Google has their eco-system and require users to make a valid user account (Raphael, 2017). Hence, user verification and authentication will always be done before giving access to any user's cloud storage. CSP also encrypts user data before storing it into the database, hence if any attack occurs, the user's data will appear jumbled, thus making it usually useless (McAfee, 2019).

It is no doubt that the security measures done by cloud providers are robust and considered safer than keeping sensitive data on personal devices. However, CSP has full control over the data, making it vulnerable to any alteration. Besides that, most cloud storage users tend to leave their online accounts always logged in, causing the data to easily susceptible to unauthorized access. Lack of protecting data-

at-rest in cloud file storage has become a concern, hence encrypting data before uploading is usually recommended.

There are various file encryption software tools that exist in the market such as AxCrypt, CryptoExpert, VeraCrypt, and CertainSafe (Fearn and Turner, 2020), however, these tools are limited to provide encryption on cloud file storage and requires users to download and install the software.

1.3 Problem Statements

i. Conventional web-based encryption tools are limited to protect cloud storage files

Conventional web-based encryption tools often offer various modes and algorithms for users to choose without proper explanation. Users will mismatch the options, and this could affect the confidentiality of their file. (Guez, 2018)

ii. Lack of lightweight data encryption algorithms

Conventional cryptographic algorithms impose some problems regarding physical size, processing requirements, memory limitation, and energy drain (Buchanan *et al.*, 2017). Although still considered being secure and fast, lightweight cryptography has been the proposed method to overcome these limitations. Currently used encryption algorithms in web-based encryption tools often struggle to support large files and may take a longer time to encrypt certain file formats due to the computing speed being slow especially on heavy-sized files (Aljawarneh *et al.*, 2017).

iii. Lack of support file sharing and data integrity features

Currently, existing encryption web tools mostly do not include file-sharing options online. Besides that, some web-based encryption tools compress the files after encrypting which results in lower quality. Thus, affecting the integrity of the data (Guez, 2018).

1.4 Project Objectives

i. To investigate a lightweight encryption algorithm in terms of computation speed and ciphertext size by using a systematic review approach and experimental

ii. To design and develop a lightweight web-based encryption tool that capable to protect the document's confidentiality and integrity by using the waterfall approach

iii. To evaluate the system usability performance of the developed lightweight web-based encryption tool by using the System Usability Scale (SUS) approach.

1.5 Project Scope

The platform that will be used for this project is a web-based application, and the supported page language will be in English. A web-based application is used because it is easily accessible online and does not require users to install additional applications such as when using a mobile platform. The working environment will be mainly on desktop and laptop.

The encryption algorithm is scoped in block ciphers using a symmetric algorithm. This project involves mainly the encryption and decryption of fixed-sized data where the size of the data is known. Such examples of data are a Portable Document Format (PDF) file or a document file (Word document file). Although stream cipher is faster and more efficient compared to block ciphers; it is best applied in a continuous stream of data such as online audio and video streaming (Gibson, 2017). Hence, block ciphers are best suited for this project. Besides, there are several block ciphers available, and open-sourced facilitates the exploration of research investigation and coding amendments to support the system requirements. Symmetric algorithm is chosen because it only provides one private key to users per file and that makes it easier for users to use an encryption tool. This project aims to help lecturers secure their documents quickly hence using a private key algorithm is better to keep it easy and simple for lecturers to use.

The research embedded element for this project is a lightweight symmetric encryption algorithm. The term "lightweight cryptography" has been a hot topic in recent years as the industry is moving towards more secure and faster but lower-cost ciphers in terms of software and hardware use in highly constrained devices

(NIST, 2021). Hence, this topic is chosen as a research embedded element, to implement a lightweight encryption algorithm into a web-based encryption tool as websites are easily accessible on various devices with a network.

This project will be tested and used by lecturers, specifically from the Faculty of Computing and Informatics (FKI). Industry collaborations are not possible due to transportation issues (in case of a physical meeting is needed) and the accessibility to nearby businesses or industries is limited. There are several modules involved in this project. Below are the modules and its explanation:

Table 1.1: Project Modules

Module	Description	Targeted Users/Roles
User Registration and Authentication	Allows users to make an account	Users will have an account that allows them to view and alter the encrypted files, including viewing encrypted passphrases within 14 days. This module also includes authentication procedures to verify users.
User account management	An online account that is present once a user completes the registration	Users can manage their encrypted files, and change basic information such as email, password, and username. Users can also choose to deactivate their accounts. If a user chooses to deactivate, any existing files and sessions will be deleted from the database alongside their personal information.
File Sharing Management	Handles online file sharing	Allows users to share files into storage services and among other users. Records, updates, and deletes the link for encrypted file access after 48 hours. This module also allows users to view the file with selected added contacts.
Web-Based encryption tool	Encrypts and decrypt uploaded files	Allows users to encrypt or decrypt files that are uploaded into the web application.
E-mail notification	Informs users on invitation for accessing encrypted files	Send emails towards users if they are granted access for viewing an encrypted file. It also sends a notification to users if a granted access is expired.

Contact List and Access Policy Management	Allows users to add friends	Users can friend, unfriend and block other user accounts. This module mainly allows the user to do file sharing with selected users.

Other than providing encryption and decryption for documents, the proposed system will provide a way for lecturers to share encrypted files between each other, and only be decrypted towards selected user accounts. Lecturers will need to make an account, where all the encrypted files and encrypted keys are kept for 14 days. After that, a lecturer can pick other lecturers (that must also have an account) to send an encrypted file. If the file is not encrypted, a prompt will appear to ask user to encrypt it first. Then user will need to select the accounts that they want to share it with. Only accounts that are ticked or included for sharing will be granted access to view it. An email will be sent as a notification towards the shared accounts to view it. The access only lasts for 48 hours to avoid unwanted access.

An example can be given in a scenario with a group of research lecturers. Lecturer A wants to send a file to other lecturers within the group but wants it to be discreet. Hence Lecturer A encrypts the file using the proposed application and send invitation towards the selected lecturers in the group. The selected lecturers will receive an email and link for access. They do not need to decrypt it because it is automatically decrypted for them to view. After 48 hours of granted access, the file will not be available for viewing. Lecturer A wants to send another encrypted file to Lecturer D but discovered the file has already been in stored for 13 days. Lecturer A still can send and grant access to Lecturer D, however a prompt will appear to remind them that the file will immediately be deleted after the 48-hour access.

1.6 Organization of The Report

This report will have six chapters including references and appendices. Below are the ten chapters within the report:

i. Chapter 1 Introduction

This chapter explains the current issue revolving around the security of online file sharing using cloud storage services and the alternative users use to secure their online data. The problem background of the subject is also discussed to identify the problems and limitations of the alternative way. Problem statements and project objectives are to identify the problems to be further investigated. Project scope explains details regarding the project, which compromises the type of platform, goals, target users, research embedded element involved, and modules in the project.

ii. Chapter 2 Literature Review

This chapter reviews data encryption algorithms, lightweight encryption algorithms including existing web-based encryption tools in the market. The literature review is done to study more in-depth the important elements of this project that later be used to evaluate, design, and develop the project.

iii. Chapter 3 Methodology

This chapter explains the methodology used in this project, which are the approaches and phases involved within this project. It also explains the techniques and decisions made to execute each phase. The phases are also illustrated using a diagram.

iv. Chapter 4 System Analysis and Design

This chapter discusses the findings gathered from the user requirements gathering done using the methods mentioned in Section 3.1. Artefacts such as DFD, ERD and User Interface (UI) designs is also included to visualise and explain the design of the system.

v. Chapter 5 Experiment Implementation

This chapter discusses the experiment implementation, which involves the dataset and parameter settings, and algorithm implementations. Each of the algorithm implementations will be described.

vi. Chapter 6 Discussion

This chapter explains the results from the experimentation, including the theoretical results that is expected to obtain. The experiment measured the wall time and CPU time after encryption and decryption, including the ciphertext size. The results are being presented using a graph bar.

vi. Chapter 7 Implementation

This chapter the implementation of components according to the modules being used in this system. The development environment is clarified, and the implementation of components are explained with figures and flowcharts.

vi. Chapter 8 Preliminary Testing

This chapter elaborates on the unit testing done during the development of the project. Unit testing and integration is done to identify if any of the use cases having problems or errors. There are mainly two types of test cases, which are positive and negative test cases.

vi. Chapter 9 System Usability Scale (SUS) Testing

This chapter discusses the data collections and analysis of the findings obtained from the SUS testing. The calculations for SUS will be shown, and any feedbacks from the testers will also be discussed as testers are required to follow an instruction sheet throughout the testing process, which simultaneously acts as a User Acceptance Testing (UAT).

vii. Chapter 10 Conclusion

This chapter discusses the constraints of the project and the future works that could be done to improve the project. The system is also discussed to see if it meets with the project's objectives stated on Section 1.4 and whether the problem statements stated on Section 1.3 is solved.

1.7 Conclusion

Due to WFH culture and online learning becoming a more prominent lifestyle among people during the pandemic, encryption has become an essential step in protecting files, especially when shared online. Including the more frequent usage of cloud storage, users need to be more alert on protecting and securing their data from unauthorized access. Web-based encryption tools are the alternative for securing data online however it lacks several matters in protecting cloud storage files. It also lacks in using a lightweight algorithm and often does not provide file sharing features. Data integrity is also not guaranteed by existing web-based encryption tools. Hence, a web-based encryption tool using a lightweight algorithm has been proposed to overcome these limitations. Lightweight algorithms will be benchmarked based on computation speed and storage size to choose the best-performing one to be implemented.

CHAPTER 2

LITERATURE REVIEW

2.1 Lightweight Symmetric Encryption Algorithms

Symmetric encryption algorithms, or also known as private-key cryptography, uses one key for encryption and decryption. There are two types of symmetric algorithms: block ciphers and stream ciphers. Block ciphers, as suggested in the name, hold data in the form of blocks with a fixed length of bits, and it is based on Pseudo-Random Permutations (PRP). The stream cipher encrypts data as it is streaming or flowing in the system's memory, and it is based on One Time Pad (OTP) (Valea *et al.*, 2019). As of now, the commonly used symmetric algorithm is Advanced Encryption Standard (AES) due to it being declared as the standard encryption algorithm by NIST, including for being a secure cipher for almost 23 years since it was published.

Recently, there has been an emerging demand for lightweight cryptography as usage of small network devices to transport data has been increasing such as in IoT (Internet of Things) applications. A lot of the current cryptographic algorithms are not suitable for hardware implementation as it consumes CPU or memory size beyond the capabilities of a small confined electronic device. Due to that issue, some of the encryption algorithms are even too slow to be implemented on the constrained small devices. Thus, lightweight cryptography aims to implement cryptography on resource-constrained devices using less memory, less computing resources, and less power supply (Buchanan *et al.*, 2018).

In this section, five known lightweight encryption algorithms will be studied, which are Twofish, XTEA, TWINE, LEA, and LED. A brief history of the algorithm will be included, along with its design, implementation, strengths, and limitations.

2.1.1 Twofish

Twofish is a symmetric block cipher designed by Bruce Schneier, John Kelsey, Chris Hall, Niels Ferguson, David Wagner, and Doug Whiting in 1998 as a submission for the 1997 DES encryption algorithm replacement challenge. It has a block size of 128 bits and supports a key size of maximum 256-bits. According to Bruce Schneier (1988), this algorithm is a general-purpose algorithm, where is designed to be used for both hardware and software including network applications. Bruce Schneier recommended Twofish for modern applications. The reason was not disclosed but it was assumed because Twofish was designed to compete against AES and it abides by the recommended block size to be used nowadays.

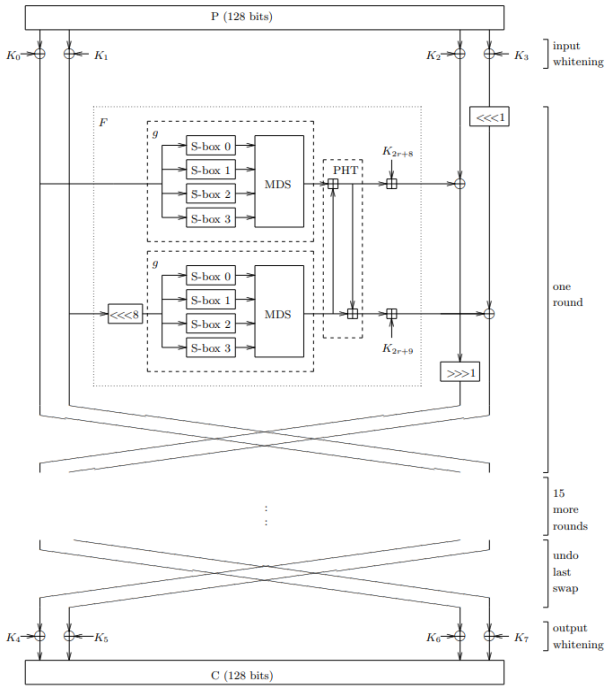


Figure 2.1: Twofish Algorithm

Source: Schneier *et al.*(1998)

Twofish is a Feistel network, thus the plaintext is divided into two parts in each round. The first part of the plaintext is sent to the left that must go through the Feistel function (F function) and the result of it will be XOR-ed with the remaining plaintext on the right.

Referring to Figure 1, in every round of Twofish, 128-bit of the plaintext are divided into two 64-bit, and each 32-bit will undergo a process called "Input Whitening" or "Pre-Whitening" before entering the F function. As the two 32-bits enter the F function, the first 32-bit plaintext will go straight into the G function, whilst the other 32-bits will go through an 8-bit rotation before entering the G function. When it enters the G function, it will be broken up into four bytes and each will be sent to four different key-dependent substitution boxes (S-boxes). Each S-box takes 8-bits of input and outputs 8-bits. This is called bijective. Then, the result of the four bytes will be combined by using the Maximum Distance Separable (MDS) matrix and will become a 32-bit text. Using Pseudo-Hadamard transform (PHT), the two 32-bit texts are combined, added to two round subkeys, and XOR-ed with the right part of the text. A 1-bit rotation was placed before and after the XOR step. After that, the algorithm continues its rounds until the last round, the last swap is undone, and "Output Whitening" or "Post Whitening" is applied to the two 64 bits. The whitening process ("pre-whitening" and "post-whitening") is where the additional subkeys are XORed into the text. To summarize, Twofish has the following algebraic groups: S-boxes, MDS matrix in GF (2^8), addition in GF (2^{32}) in PHT, addition in GF (2) used in XOR and 1-bit rotations. The mathematical models are mixed up by the round function.

2.1.2 XTEA

XTEA algorithm is an extension of TEA algorithm, which were both designed by Roger Needham and David Wheeler. XTEA was published to combat the two weaknesses of TEA algorithm, that is impractical related keys attack and the key length being 126-bits. Since XTEA is a successor of TEA, its design is the same except for some changes involving the key schedules and key material (Needham and Wheeler, 1997).

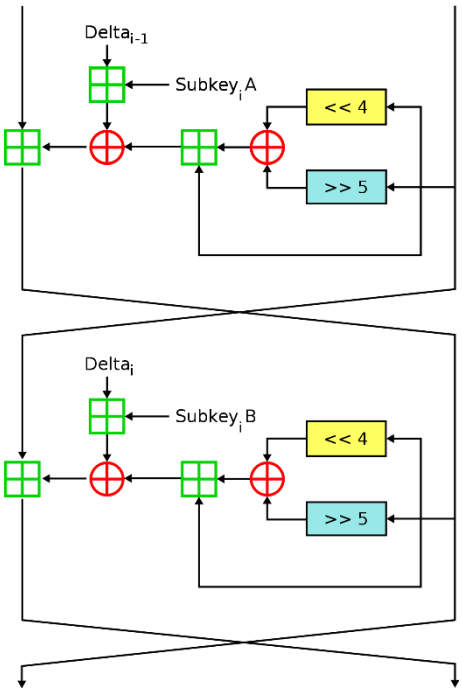


Figure 2.2: XTEA algorithm

Source: Wheeler and Needham (1997)

XTEA is based on the Feistel network, and it has 64-bit block sizes with 128-bit key sizes. XTEA does 64 rounds and has a more complex key schedule than TEA, including a rearrangement of shifts, XORs, and additions. Both TEA and XTEA has a 'magic' value being assigned which is $DELTA = 0x9e3779b9$, where it is derived from the golden number ($(\sqrt{5} - 1)2^{31}$).

Based on Figure 2, the plaintext will be divided into two parts, where for convenience, the left side will be named LS and the right side will be RS. The RS will be further divided into two parts and shifting will be done. Next, the two shifted RS parts are combined using XOR. Once combined, the new RS will go through addition and be XORed with the result of key scheduling. Key scheduling is the addition operation of DELTA and sub-key. After that, the new RS will go through another addition operation with the LS. This whole process continues for 32 cycles.

```
tean( long * v, long * k, long N) {
unsigned long y=v[0], z=v[1], DELTA=0x9e3779b9 ;
if (N>0) {
       /* coding */
   unsigned long limit=DELTA*N, sum=0 ;
   while (sum!=limit)
     y+= (z<<4 ^ z>>5) + z ^ sum + k[sum&3],
     sum+=DELTA,
     z+= (y<<4 ^ y>>5) + y ^ sum + k[sum>>11 &3] ;
         }
else  {


           /* decoding */
    unsigned long sum=DELTA*(-N) ;
    while (sum)
      z-= (y<<4 ^ y>>5) + y ^ sum + k[sum>>11 &3],
      sum-=DELTA,
      y-= (z<<4 ^ z>>5) + z ^ sum + k[sum&3] ;
       }
v[0]=y, v[1]=z ;
return ;                                  }
```

Figure 2.3: The source code for encryption and decryption process in XTEA

Source: Wheeler and Needham (1997)

2.1.3 Twine

Twine is a symmetric lightweight 64-bit block cipher that has key sizes of 80-bit (TWINE-80) or 128-bit (TWINE-128), designed by Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. All the designers are from a Japanese multinational information technology and electronics company called NEC Corp.

According to Suzaki *et al.* (2013), it implements enhanced Generalized Feistel Structure (GFS) to improve diffusion, with the absence of bit permutation and Galois-Field matrix. The diffusion layer in GFS applied in this algorithm performs block shuffle instead of left cyclic shift, following a proposal of an improved GFS from Suzaki and Minematsu (2010). The goal of this algorithm is to minimize the footprint on hardware implementation, small memory consumption on software implementation, and unified encryption and decryption method.

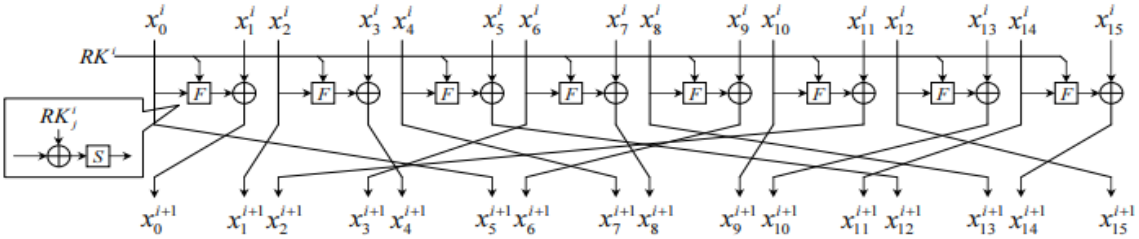


Figure 2.4: TWINE algorithm

Source: Suzaki and Minamatsu (2010)

TWINE algorithm only has one 4-bit S-box and 4-bit XOR. The structure of TWINE is Type-II GFS where there are eight pairs of 8-bit data paths (4-bit per data path). Each pair has an F function which is denoted as

$$y = S(x \otimes k)$$

where S is the 4-bit S-box, x is the input and k is the round key. The round key, k is produced by a key schedule using the secret key provided, where the length is either 80-bit or 128-bit. The key schedule uses 35 6-bit constants to output a 36 32-bit round keys (Ertaul and Rajegowda, 2017). Basically, in each round, the F function will XORed the input (or left data according to Figure 4) with a sub-key, then applies the 4-bit S-box. The result will be XORed with the data on the right path. After that, a diffusion layer will be done. It does 36 rounds, and this goes for both key sizes. For the last round, the diffusion layer is not applied.

2.1.4 LEA

Lightweight Encryption Algorithm or LEA was developed in South Korea, and designed by Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Dong-Geon Lee. It is a symmetric 128-bit block cipher with three available key sizes which are 128-bit, 192-bit, and 256-bit. The number of rounds depends on the key sizes used hence, LEA-128 does 24 rounds, LEA-192 does 28 rounds and LEA-256 does 32 rounds. The structure of the LEA algorithm is based on ARX operations which are modular addition, bitwise rotation, and bitwise XOR for 32-bit words. According to Hong *et al.* (2014), LEA was designed to be a lightweight

algorithm to be used in software implementations, including on limited RAM/ROM or CPU devices such as smart meters.

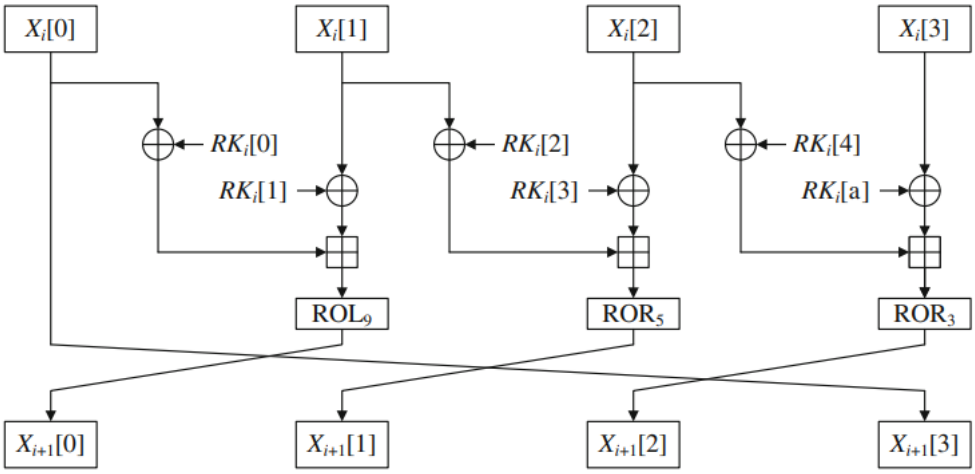


Figure 2.5: LEA algorithm

Source: Hong *et al.* (2014)

Referring to Figure 5, the LEA algorithm starts with initializing the X_i values with plaintext. Then, in each round, each X_i value on the left side will be XORed with a round key and later will undergo modular addition with the next X_i value that also has been XORed with a round key. Starting from $X_i[1]$ until $X_i[3]$, the result from the modular addition will be rotated to produce the output. To make it easier, the algorithm is generally in three phases:

- Initialization: The plaintext is used to initialize X_i values. Where for each *r* rounds, plaintext $P = (P[0], P[1], P[2], P[3])$ is being set to ciphertext $C = (C[0], C[1], C[2], C[3])$.
- Round iteration: The outputs which are $X_{i+1}[n]$ (where $0 \leq n \leq 3$) of each round are calculated as

$$X_{i+1}[0] = ROL_9\big((X_i[0] \otimes RK_i[0]) \boxplus (X_i[1] \otimes RK_i[1])\big) X_{i+1}[1] = ROL_5\big((X_i[1] \otimes RK_i[2]) \boxplus (X_i[1] \otimes RK_i[3])\big)$$

$$X_{i+1}[2] = ROL_9\big((X_i[2] \otimes RK_i[4]) \boxplus (X_i[1] \otimes RK_i[5])\big)$$

$$X_{i+1}[3] = X_i[0]$$

- Finalization: Ciphertext is obtained once the rounds have finished. It is produced from the value X_r, where r is the last round.

The decryption process is the inversion of all the phases above. The algorithm does not apply a specific last round operation as other algorithms due to the designers prioritizing the algorithm for encryption. However, it is stated that the algorithm still decrypts as fast as other block ciphers.

2.1.5 LED

Light Encryption Device (LED) algorithm is a 64-bit block cipher that accepts two key sizes (which are 64-bit and 128-bit) lightweight block cipher designed by Jian Guo, Thomas Peyrin, Axel Poschmann, and Matt Robshaw. It is based on AES design principles, however, altered to perform faster on hardware. The cipher is arranged in a 4x4 matrix and is loaded in rows instead of columns because it is more hardware-friendly.

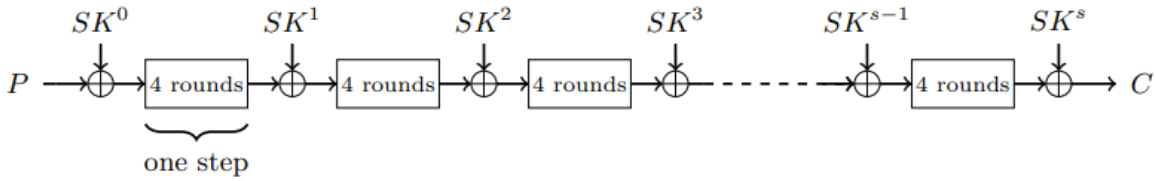


Figure 2.6: Use of subkeys in LED algorithm during steps

Source: Guo *et al.* (2011)

The encryption process in a LED algorithm consists of two main operations: addRoundKey(STATE, SK^i) and step(STATE) operations. The number of steps (s) depends on the key size chosen, where 64-bit key size results in 8 steps and 128-bits results in 12 steps. The second operation called step (STATE) consists of four rounds of encryption and it involves four operations that are executed sequentially: adding constants (AddConstants), the substitution of cells using PRESENT S-box (SubCells), shifting rows (ShiftRow), and mixing columns (MixColumnSerial). The

final value of the STATE will be XORed with the last sub-key (SK^8) to achieve the ciphertext.

2.1.6 Comparison of Lightweight Symmetric Encryption Algorithm

Five lightweight symmetric encryption algorithms have been observed by understanding its brief history, design, and implementation. The algorithms are further observed by listing the differences, alongside comparing, and discussing the strengths and limitations.

Table 2.1: Comparison of lightweight symmetric encryption algorithm

Criteria	**Lightweight Symmetric Encryption Algorithms**				
	Twofish	**XTEA**	**TWINE**	**LEA**	**LED**
Structure	Feistel network	Feistel network	Type-II Generalized Feistel Structure	Feistel network	Feistel network
Main functions	Input whitening, substitutions, MDS matrix, additions, rotations, output whitening	Additions, subtractions, shifts	Substitution, XOR, box-shuffle	ARX operations (Modular addition, bitwise rotation, bit wise XOR)	Substitution, Add-constants, Row-shifts, Column-mix, Add-round key
Key Size (in bits)	128/192/256	128	80/128	128/192/256	64/128
Block Size (in bits)	128	64	64	128	64

Processing rounds	16	64	16	24/28/32	32/48
Platform implementation	Hardware, software, and network applications	Software	Hardware, and software	Hardware and software	Hardware
Computation speed (in cycles)	5200	83735	15673	1171	35891
RAM footprint (in bytes)	200	120	474	664	654
Cryptanalysis attacks	1. Differential cryptanalysis 2. Side-channel attacks	1. Related-key differential attack 2. Related-key rectangle attack	1. Bi-clique attack 2. Impossible Differential 3. Saturation attack 4. Differential attack 5. Key scheduled-based attack	1. Differential cryptanalysis 2. Truncated differential 3. Linear 4. Zero correlation 5. Boomerang 6. Impossible differential 7. Integral 8. Differential 9. Linear	1. Differential attacks 2. Bi-clique attacks 3. Algebraic attacks

				10. Related-key differential	
Physical attacks	None	1. Man-in-the-Middle (MiTM) attack	1. Man-in-the-Middle (MiTM) attack	None	1. Man-in-the-middle attacks (MiTM)
Strengths	1. No recorded successful physical attacks 2. The performance can be altered accordingly with trade-offs such as longer key setup for faster encryption or fast key setup but slower encryption	1. Executes fast due to small usage of memory	1. TWINE-80 is suitable for microprocessor implementations, and wireless sensor networks	1. Faster than AES by 1.5-2 times speed 2. Has a security margin greater than 37% 3. Known for being resistant towards all known cipher attacks 4. Safe structure as it uses different key schedule algorithms	1. Provides security against related-key attacks and most cipher attacks

				5. Energy-efficient	
Limitations	1. low performance in terms of speed, memory consumption, and security	1. Not suitable for hardware implementation 2. Susceptible to related-key attacks 3. Slow computation speed compared to other algorithms	1. There is not a lot of research on the uses of this algorithm in software implementation	1. Decryption is slower compared to AES	1. Software implementations are not optimal

Twofish is a general-purpose encryption algorithm, it has been used in a variety of software and hardware applications. As of now, no known successful physical attacks are being done on the algorithm, and only the two best theoretical attacks have been recorded to be susceptible such as impossible differential attack (Ferguson, 1999) and truncated differential cryptanalysis (Moriai and Yin, 2000). Schneier (2005) has personally responded towards the truncated differential attack to debunk rumors on Twofish being broken, by stating that the researchers did not perform any cryptanalysis attacks but rather theorized the possible differential characteristics that could break the algorithm. Besides that, according to Ortiz and Compton (2016), a simple power analysis attack can fully break Twofish using Hamming weights of the Key Schedule Computation. However, Schneier (2017) responded that it is possible to happen if the device undergoing encryption is physically controlled. A side-channel attack was done by Ma *et al.* (2017) however only 96-bits out of 128-bits were successfully solved, with the knowledge of ciphertext or plaintext under 2 hours. It can be concluded that Twofish is secure as

of now, and the only susceptible attacks are physical attacks that are non-mathematical. The weaknesses that Twofish owns are that its performance in terms of speed, memory consumption, and security is lower compared to its predecessor (Blowfish) and AES (Jeevalatha and SenthilMurugan, 2018).

XTEA is known as a very simple and short algorithm, that is designed for software implementation since it does not require a lot of memory space and executes rather fast. However, XTEA is susceptible to related-keys attacks and it is not ideal to be implemented on hardware due to it exceeding 3000 Gate Equivalent (GE) (Hatzivasilis *et al.*, 2017).

TWINE is a lightweight encryption algorithm; it has been tested on several resource-limited devices such as microprocessors and circuits. According to Hosseinzadeh and Bafghi (2017), TWINE-80 displayed satisfying performances for Atmega128 microprocessor implementations in terms of memory usage and energy. It is also concluded that TWINE-80 is most suitable for wireless sensor networks. Several cryptanalysis attacks had been done towards TWINE. An electromagnetic analysis attack has been proposed to measure the vulnerability of TWINE cipher on hardware (Yoshikawa *et al.*, 2016). Other theoretical attacks are bi-clique attacks, and MitM (Man-in-the-Middle) attacks (Toiba and Youssef, 2015).

LEA is designed for implementation in software and hardware, and it is known to be resistant against all known cipher attacks. This algorithm has been used in research involving resource-bounded IoT devices, where certain techniques have been introduced to further improve the performance of the algorithm such as Bit-Serial Technique (Choi *et al.*, 2018) and pipelining implementation (Mishra et al., 2019). According to Hong *et al.* (2013), LEA-128 is faster than AES-128 and consumes less ROM size in software implementation. However, decryption is much slower compared to AES due to the serial process of decryption.

LED is designed with hardware in mind, software implementation may be worse compared to other algorithms. According to Appel *et al.* (2016), when compared with several other algorithms (TEA, XTEA, AES, KATAN, and KTANTAN), LED displayed amongst the worst in terms of software implementations. It is also susceptible to MiTM attacks and differential attacks. However, there are advantages of LED such that the flexibility of its key length allows it to be adjusted to the security level that is wanted.

From the comparison findings table above, the three algorithms that will be selected for further experimental investigation are Twofish, LEA, and XTEA. Two from these three algorithms have considerably fast computation speed among the other algorithms. XTEA has the lowest RAM footprint compared to other algorithms. Since this project will be a web-based application, memory consumption is important for it to function efficiently and fast, especially on smartphones. It would be interesting to compare LEA with the other algorithms as it has the highest RAM footprints despite having the fastest computation speed. Besides that, the limitations of these three algorithms are less concerning and do not affect the project too much, compared to TWINE and LED due to it being more catered towards hardware and 8-bit platforms.

2.2 Existing Web-based Encryption Tools

There are several free web-based encryption tools available in the industry. Five of them will be identified by using purposive sampling and observed to discuss its overall features, interface design, process flow, strengths, and limitations.

2.2.1 Free Crypt

Free Crypt is online file encryption and decryption tool. On the website, there are three visible features which are file encryption, file decryption, and encrypted file analysis. The website also has icons that are linked to other of its five websites: Sticky Notes, EncryptionBox, File Encryption, Runaway, and Weather. Two of the separate websites ("EncryptionBox" and "File Encryption") are also encryption tools and provide similar but more features than Free Crypt.

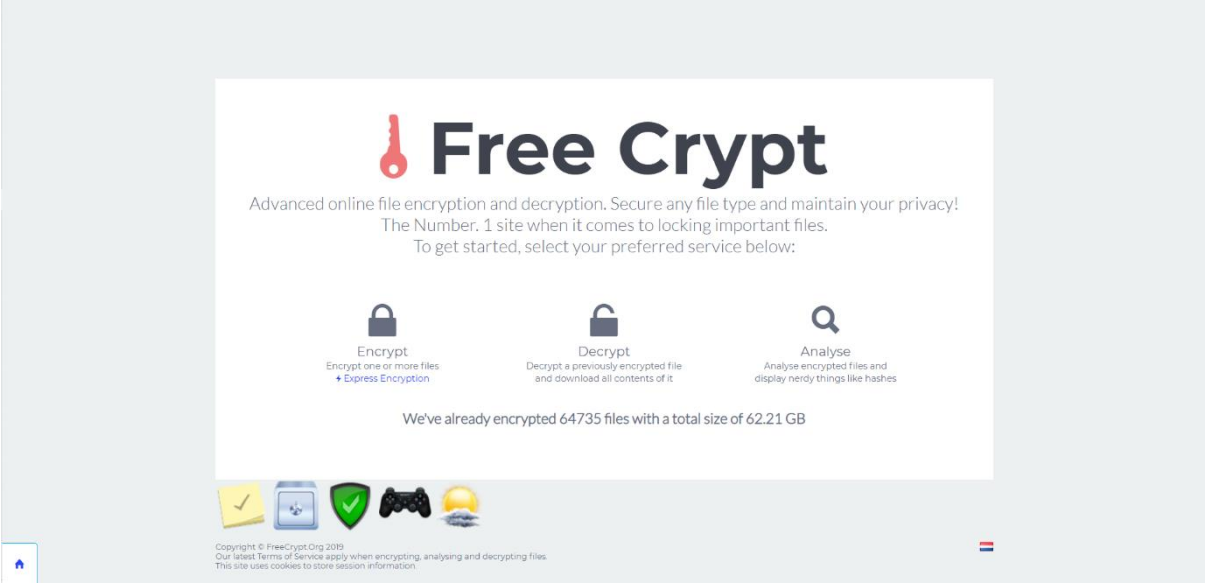


Figure 2.7: Free Crypt homepage

Source: FreeCrypt.Org (2019)

The first feature as shown in Figure 8, is "Encrypt" and it further provides two choices for users: "Express Encryption" and "Encryption for Pro's".

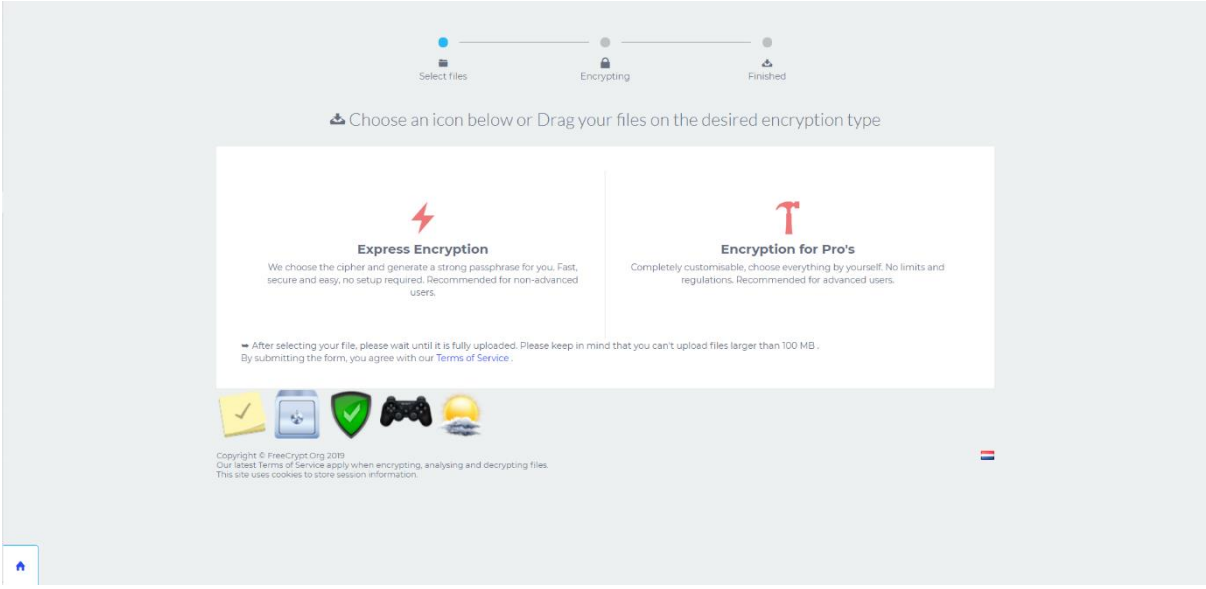


Figure 2.8: The two options under 'Encrypt'

Source: FreeCrypt.Org (2019)

After clicking the first option ("Express Encryption"), a graphical user interface (GUI) will appear and allows users to choose the wanted files to be encrypted. Users can choose multiple files by pressing the control (ctrl) key while selecting files.

Selected files are compiled into one cry file (.CRY). After selecting the wanted files, users will be directed to a page containing the encrypted file, and an auto-generated key is stated in blue alongside a reminder for users to copy and paste the key from the page or by scanning a QR code so that it can be used for decrypting the file.

Users are required to download the file into their devices to obtain the encrypted file. There is also a "Host and Share" option where users can share the encrypted file via email. However, users must download the file first as it is a separate web tool and directs to a new page on the same browser tab upon clicking the option.

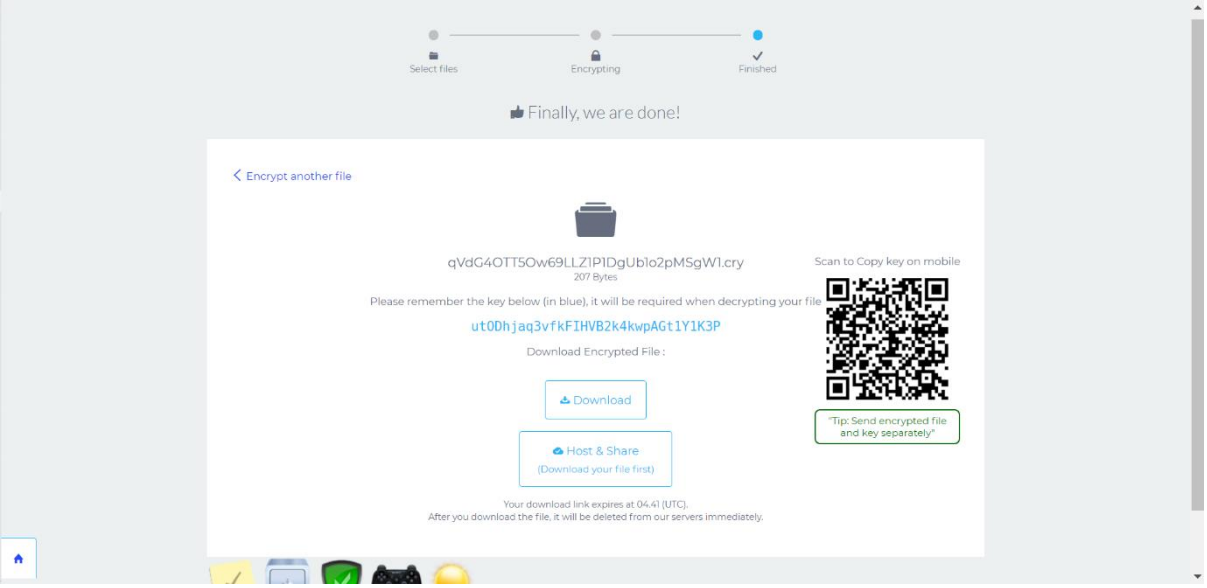


Figure 2.9: The page to download or share encrypted file

Source: FreeCrypt.Org (2019)

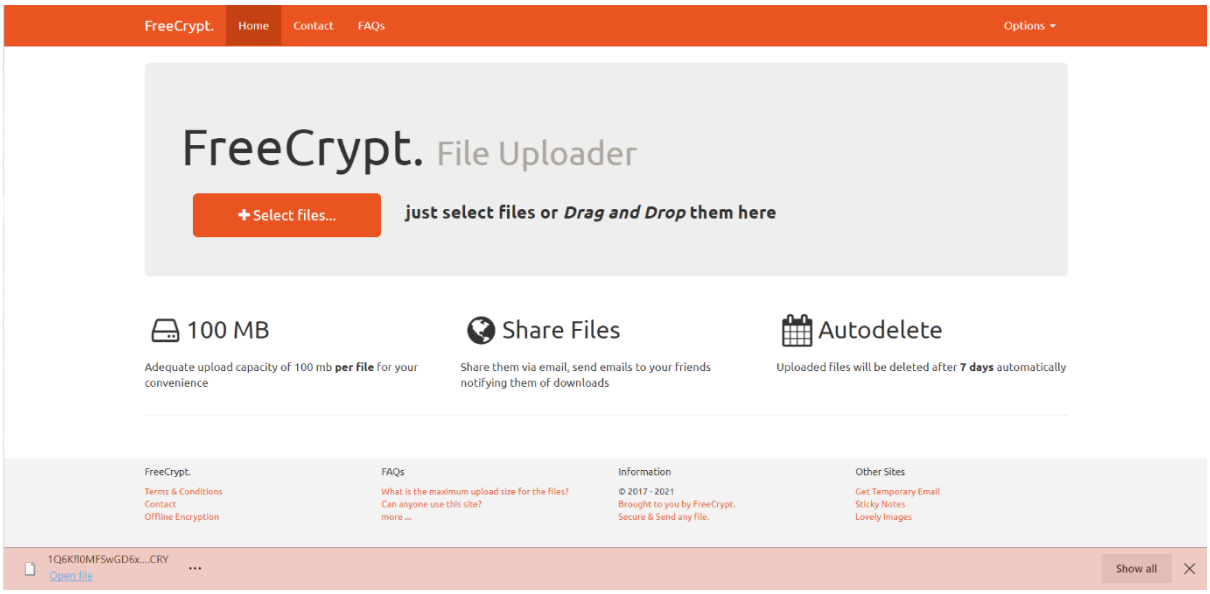


Figure 2.10: File sharing web-tool provided by Free Crypt

Source: FreeCrypt.Org (2019)

On the file-sharing website, users can upload the encrypted files or any selected files by clicking "Select files…" or drag and drop the files into the designated area. After successfully uploading the files, users are given two links for downloading and deleting a file, that can be shared with other people using email or by following the link. Users only can include a maximum of three recipients.

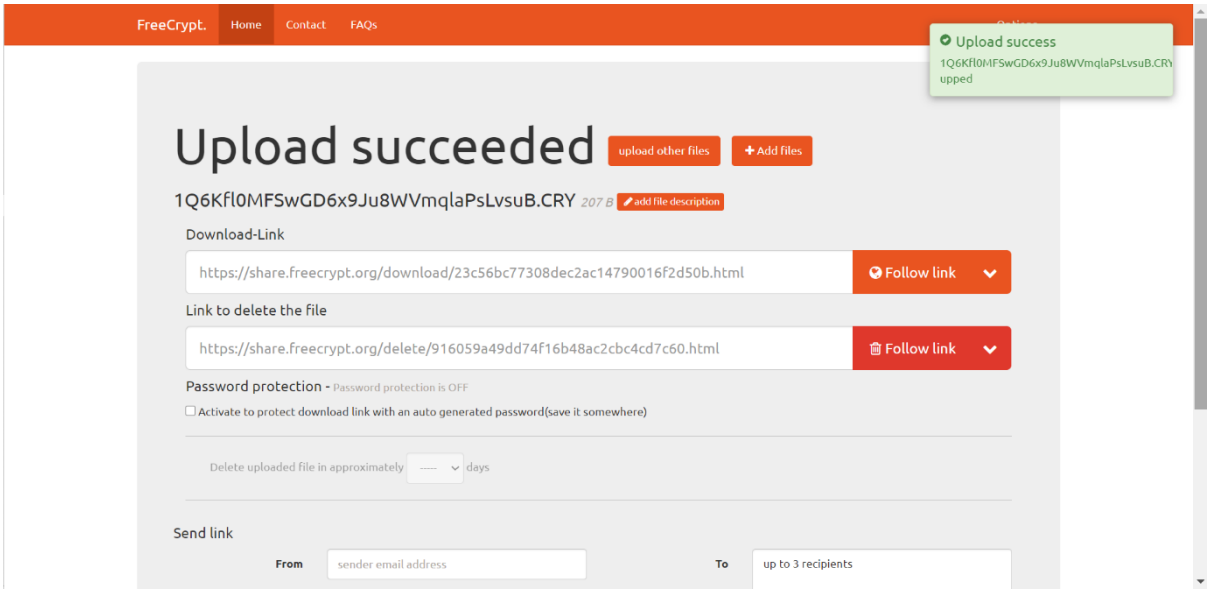


Figure 2.11: Page for sharing the link

Source: FreeCrypt.Org (2019)

In addition, users can set a password on the file, but it is auto-generated by the web tool. Users can also choose the time duration for the files to be kept in the servers. The maximum time is seven days.

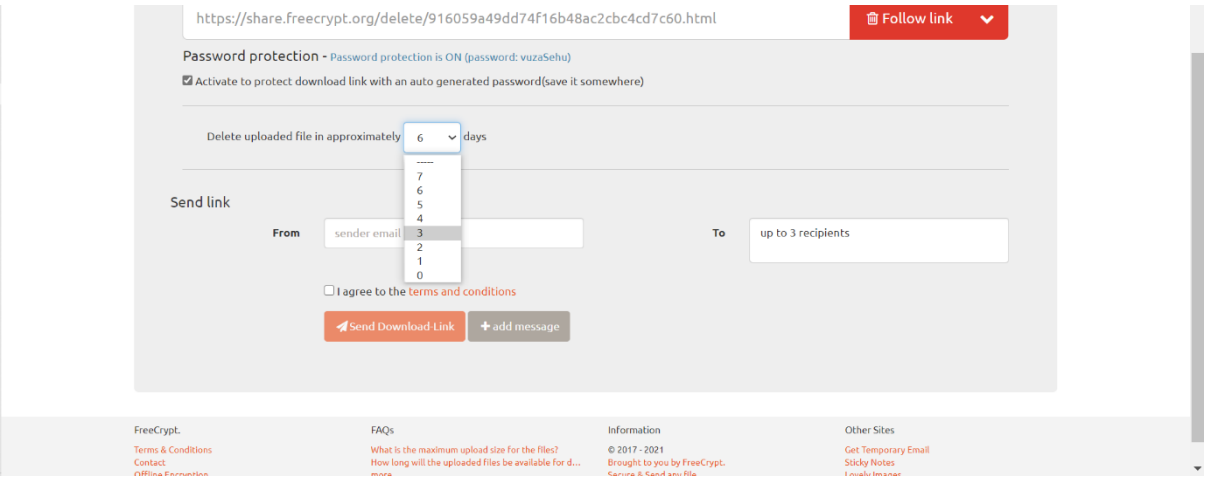


Figure 2.12: Password protection feature and file sharing through email

Source: FreeCrypt.Org (2019)

The second feature under "Encrypt" is "Encryption for Pro's". This feature is recommended for advanced users, where it is stated that these users can customize their encryption methods. Similar to the first feature, users are greeted with a GUI and are required to select files to be encrypted. However, there are no further actions after selecting the files. The website refreshes and the selected files are gone, as seen in Figure 15.

Figure 14 (i)

Figure 14 (ii)

Figure 2.13: Display of 'Encryption for Pro's' that is unable to be processed

Source: FreeCrypt.Org (2019)

The second feature available on the web-tool homepage is "Decrypt", where it provides a file decryption service. After clicking the "Decrypt" icon, it is directed to a page where users can upload the encrypted file along with the key. After uploading the file and providing the key, users can download the decrypted file.

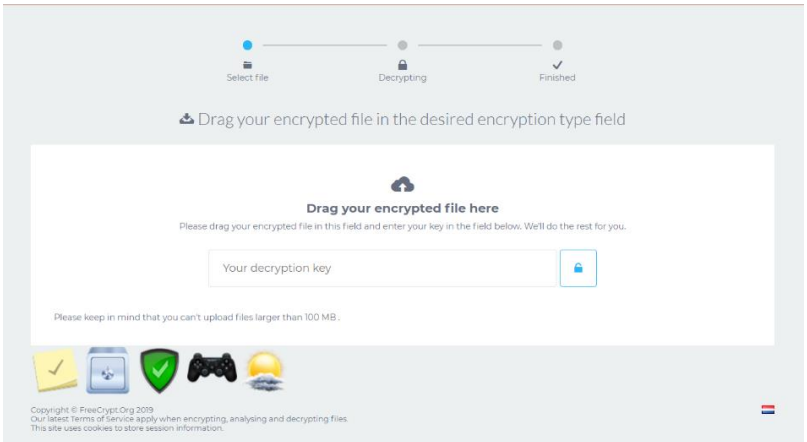


Figure 2.14: Page for uploading files and inserting key for decryption

Source: FreeCrypt.Org (2019)

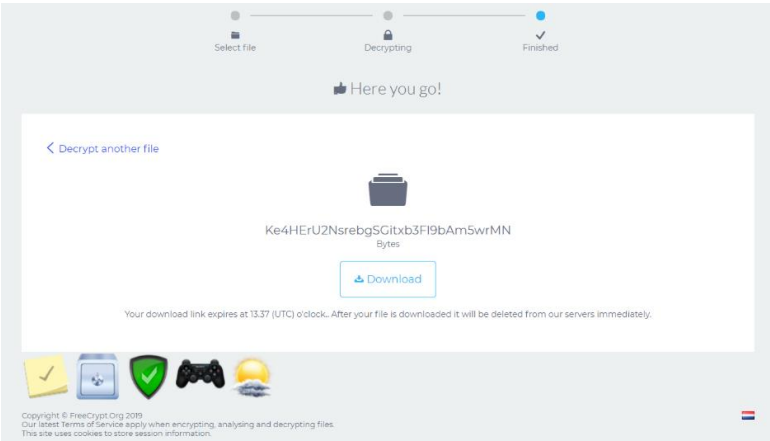


Figure 2.15: Decrypted file ready to be downloaded

Source: FreeCrypt.Org (2019)

Encrypted files from this web tool are set to 207 Bytes specifically, regardless of the size of the original data. Hence, the encrypted files can be decrypted always but at certain times, the web tool is unable to automatically download the file into the device. An error page will appear in these cases, as such in Figure 18.

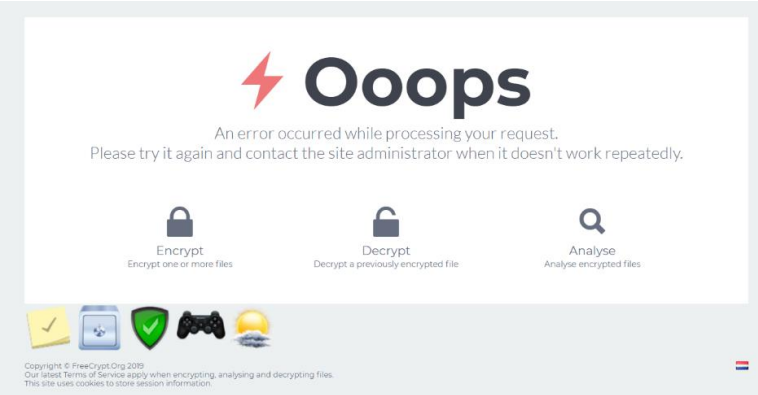


Figure 2.16: Error message display after clicking 'Download' button

Source: FreeCrypt.Org (2019)

The last available feature is the "Analyse" feature that supposedly analyses the uploaded encrypted file to display details such as hash. Unfortunately, this feature cannot be tested as it is currently not operable. It showed an error image that is similar to Figure 18.

One of the main problems with this web tool is the choice of the file extension. Three different file types use the cry file extension, which are Crypto Chat II Encrypted Notebook, Atrial Jaguar CRY-format graphic image, and CryLocker ransomware. The website does not specifically state the type of cry file it uses but it can be assumed to be the first option. It is still concerning as the CryLocker ransomware also encrypts the affected user's file and later asks for ransom using Bitcoin by forcing the victim to buy a fake encryption product (Meskauskas, 2020). This can induce panic and distrust in users that are aware of the issue.

Besides that, this web tool does not have error prevention. If a user still tries to upload a file of more than 100 MB, the web tool still processes it and soon crashes due to it being unable to support it. Sometimes the features on the web tool cannot be used and no explanation of the error is visible on the web tool as it immediately refreshes the page.

There is another available encryption tool being linked within this web tool, which is 'Secure Free Crypt'. This separate but similar web tool provides three algorithms for users to choose from, which are AES, SHA-1, and Serpent. On the homepage, it states that it is an offline encryption tool, which means it is a fully client-side encryption tool. This can be seen through the page source, as the developer included the algorithms in the page file. The algorithms are coded in JavaScript.

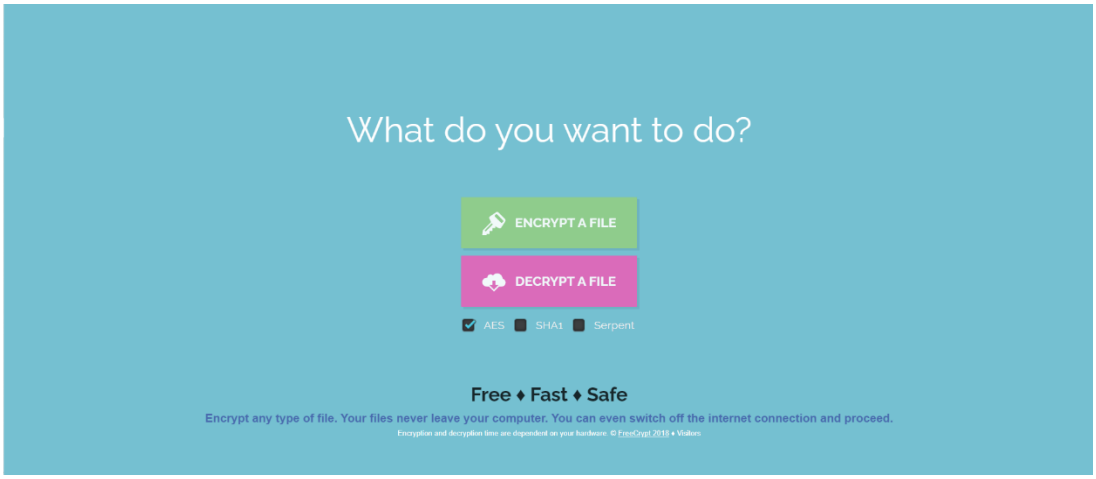


Figure 2.17: Homepage of Secure Free Crypt

Source: FreeCrypt.Org (2019)

```
155     <!--<footer>
156         <a class="tz" href="https://freecrypt.org">Free Online Encryption</a>
157         <div id="tzine-actions"></div>
158         <span class="close"></span>
159     </footer>-->
160     <!-- Include the AES algorithm of the crypto library -->
161     <!--Coded by FreeCrypt.Org-->
162     <script src="assets/js/aes.js"></script>
163     <script src="assets/js/crypto-js.js"></script>
164     <script src="assets/js/serpent_cipherEnc.js"></script>
165     <script src="assets/js/pbkdf2.js"></script>
166     <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script>
167     <script src="assets/js/script.js"></script>
168     <script src="assets/js/Blob.js"></script>
169     <script src="assets/js/canvas-toBlob.js"></script>
170     <script src="assets/js/FileSaver.min.js"></script>
171     <script src="assets/js/base64js.min.js"></script>
172     <script src="assets/js/download2.js"></script>
173     <script>
174         window.onload = function () {
175             (function (a) { "use strict"; var b = a.HTMLCanvasElement && a.HTMLCanvasElement.prot
176         };
177     </script>
178
```

Figure 2.18: Scripts included in the web-tool for full client-side encryption

Source: FreeCrypt.Org (2019)

To use this web tool, users need to choose which method and algorithm to execute on the homepage. Once chosen, the page will be transitioned to a different view that has a button for users to upload files. Users can only upload one file per encryption.

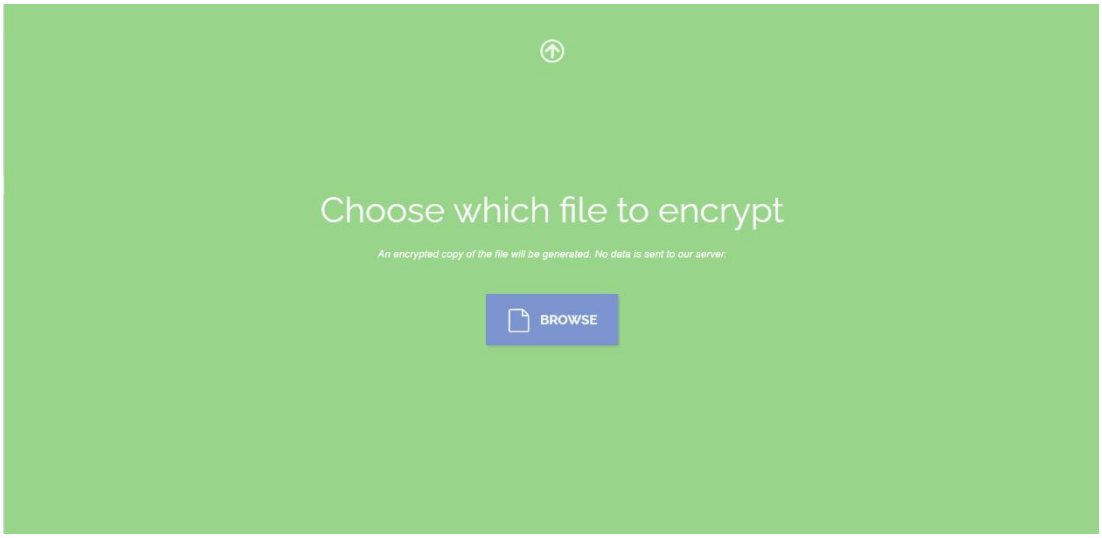


Figure 2.19: Page for uploading file to be encrypted

Source: FreeCrypt.Org (2019)

After that, the page transitions and shows a text field that for users to enter the passphrase. The page firmly states for users to remember the passphrase as there is no recovery feature available once forgotten. After clicking the 'Encrypt!' button, the page encrypts the file and transitions to a page that displays a download button for users to download the encrypted file. The web tool encrypts files into FRY extension file (.fry). FRY file or known as Fryrender Scene Format file, is one of the 3D image files used in operating systems.

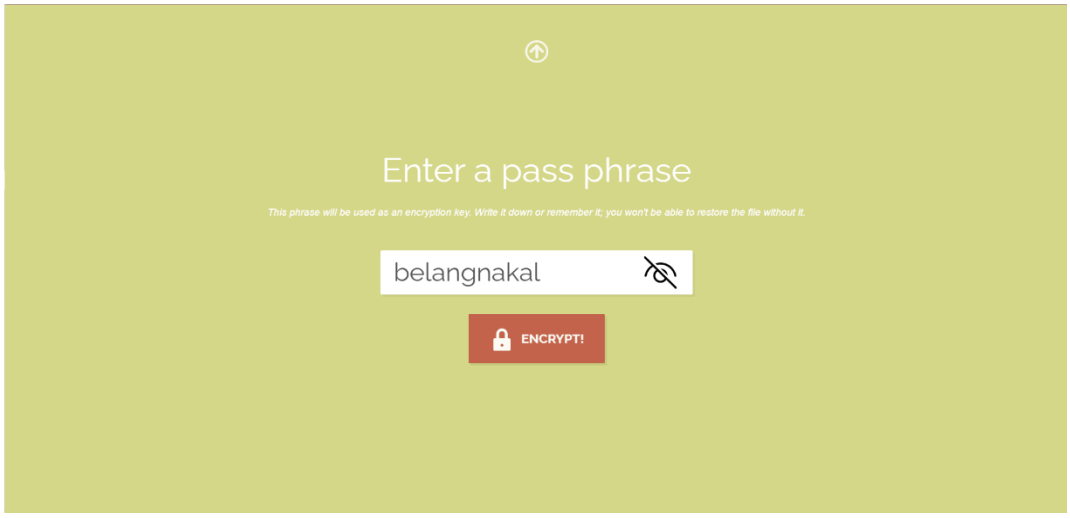


Figure 2.20: Page for setting up passphrase as the key

Source: FreeCrypt.Org (2019)

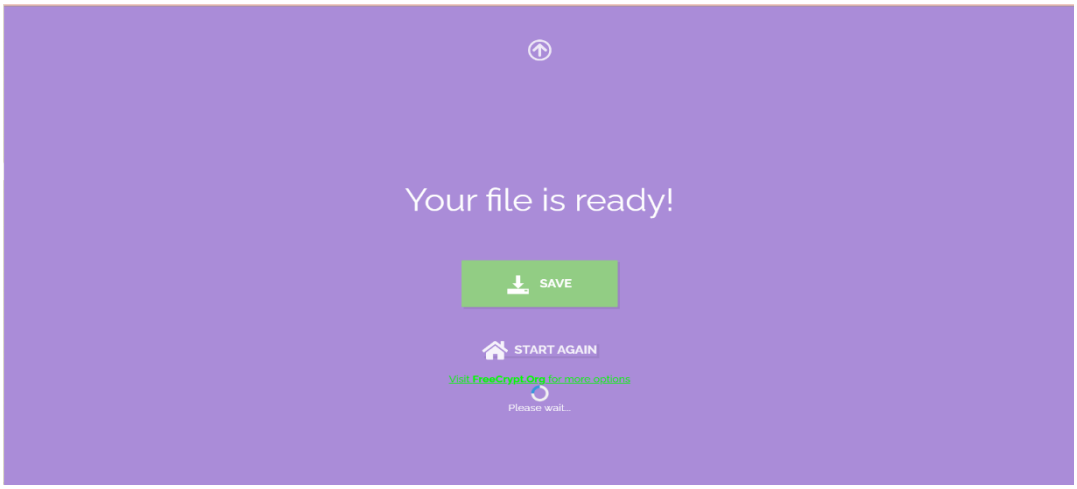


Figure 2.21: Page for downloading the encrypted file

Source: FreeCrypt.Org (2019)

This web tool runs smoothly, and its features are executed nicely. However, it does not state the maximum file size it can support, thus users may need to test with different sizes of files to know. It performs better than its online encryption tool in terms of speed, and user interface. The choice of file extension for encryption is unique and rarely seen in other available encryption web tools.

Overall, the web tool has a very clear description of what features it offers and the design is not clustered. However, it is quite broken, and it is confusing that there are other separate but similar encryption tools being offered by the same developer. The separate web tool is also promoted subtly as it is linked in an unlabelled icon. This web tool can be further improved by fixing the features and re-design the web tool to be more informative, such as displaying errors more clearly and setting a maximum limit attribute on the HTML tags to prevent the server from crashing due to unsupported file size.

2.2.2 TxtWizard

TxtWizard is simple text encryption and decryption web tool. On the homepage, users can immediately see the cryptography features including other tools which are for hashing, encoding, and compression.

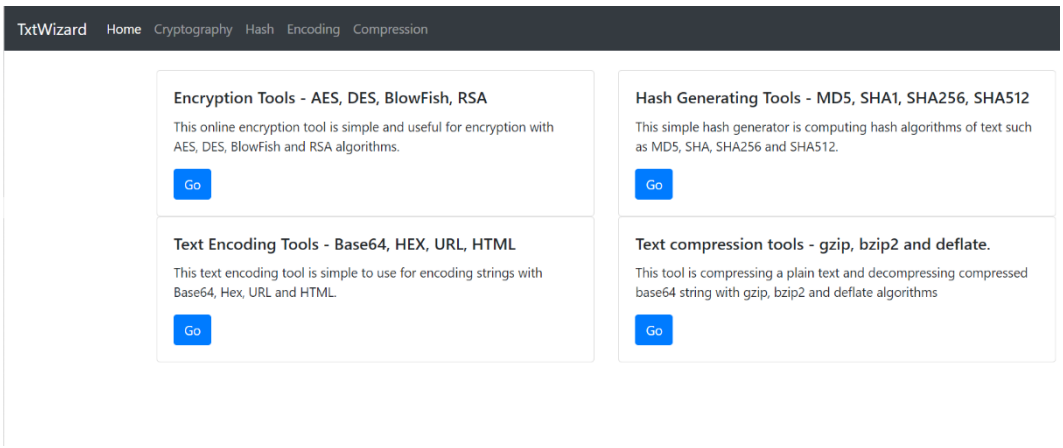


Figure 2.22: Homepage of TxtWizard

Source: wellknowngeek (2019)

There are two divisions on the homepage, which are "Encrypt" and "Decrypt". Each of these divisions have the same structure where users will provide or choose several items: encryption algorithm, mode, padding, plaintext or ciphertext, key and initialization vector (IV).

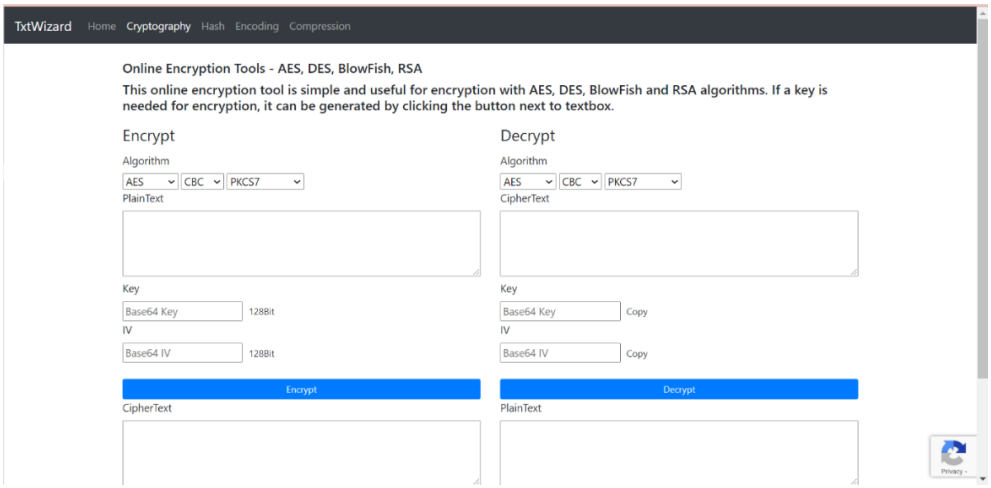


Figure 2.23: Cryptography page of the web-tool

Source: wellknowngeek (2019)

This web tool offers four encryption algorithms which are AES, DES, Blowfish, and RSA. The modes available are Cipher Block Chaining (CBC), Electronic Code Book (ECB), Cipher FeedBack (CFB), Output Feedback (OFB), None, and Propagating Cipher Block Chaining (PCBC). Key and IV are auto-generated by clicking a button that is labeled as the key and IV size of the chosen algorithm.

The process is simple: users only need to choose which algorithm to apply alongside its mode and padding. Then, users write down the plaintext in the provided box. The key and IV need to be auto-generated to work properly. Once users click 'Encrypt', the cipher text will appear alongside an unexplained text below it.

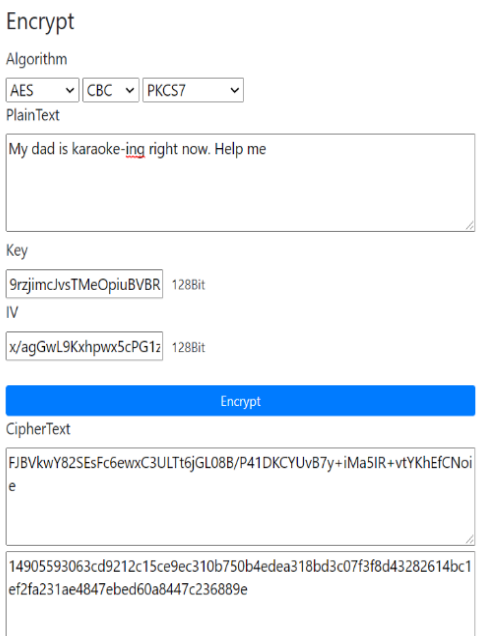


Figure 2.24: Encryption feature provided on the web-tool

Source: wellknowngeek (2019)

To decrypt the text, users just copy and paste or highlight the ciphertext and transfer it to the cipher text box under the 'Decrypt' division. The key and IV can be copied directly from the 'Encrypt' division simply by clicking the 'Copy' button beside both the text field of key and IV. Once users click 'Decrypt', the message will be decrypted.

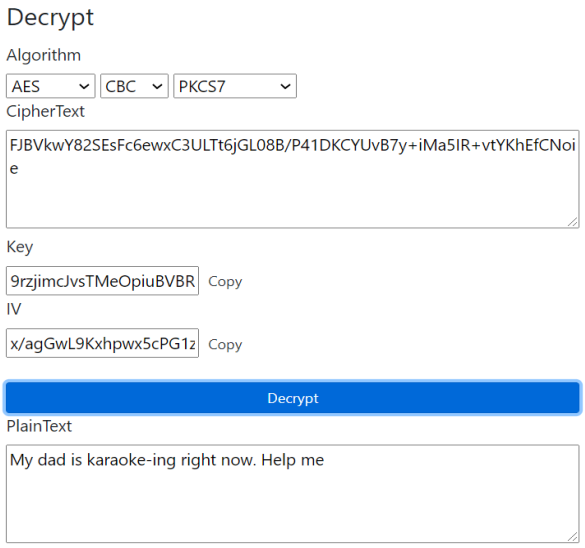


Figure 2.25: Decryption feature provided on the web-tool

Source: wellknowngeek (2019)

The other features provided by the web tool are hash generator, text encoding tool, and text compression tools. These tools are not involved nor correlate with the project hence it is not observed. The tools are similar in design to the encryption tool, where users enter a text and choose which method to use. After that, the result will be automatically shown.

To summarize, the website is very simple and easy to understand however there are some questionable design choices. There is no clarification regarding the key and IV, where users can only auto-generate it to use the tool. The button for the key and IV is also not visible as a button but rather as text. Also, the text field is editable hence a user might assume that the key can be self-written. Besides that, the RSA algorithm cannot be used in the web tool since there is no visible way of providing the key and IV. RSA is an asymmetric encryption algorithm hence the design should differ from the other symmetric algorithms provided. Another problem is that the tool seems to have problems in encrypting using padding options other than PKCS7. Other algorithms and mode have been tried and none of it works with other padding options.

2.2.3 ENCRYPT.One

ENCRYPT.One is an online text and file encryption and encryption that uses only the AES-256 algorithm. The cryptography features are done by using a full client-side encryption model. The homepage of the web tool shows a text field that users can type and below it is an area (or a button if the browser is small) for uploading files. Users can provide their passphrase to be used as the key or auto generated from the web tool itself. Users do not need to provide a passphrase nor for the passphrase to be auto generated. There is also a password generator provided, but it is not applied in the encryption process. The generated password can be used as the 'secret' text to be encrypted.

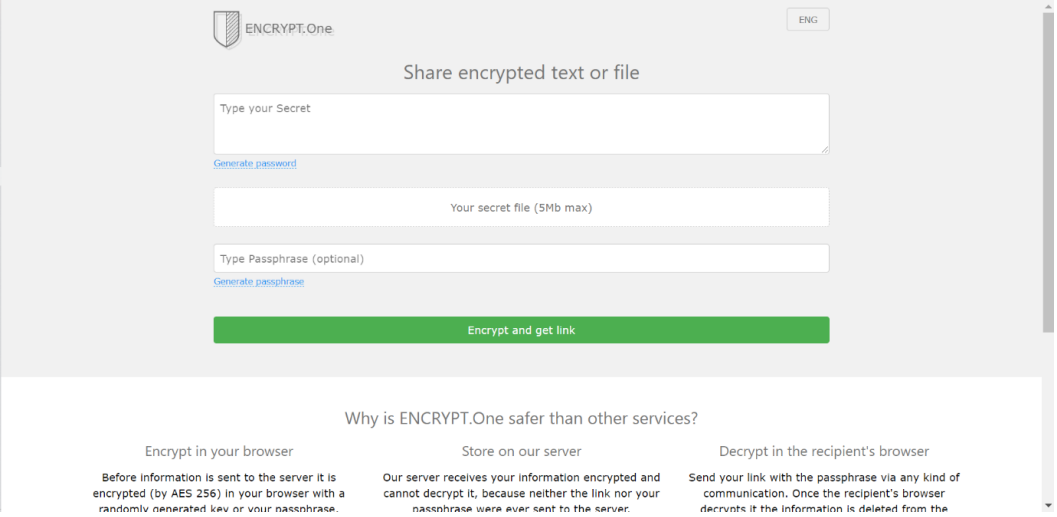


Figure 2.26: Homepage of ENCRYPT.One

Source: ENCRYPT.One (2016)

Users can upload both text and file at the same time to be encrypted as shown in Figure 29. It produces the same result at the end where a link is being generated to be shared and accessed once. After opening the link and the secret encrypted file, the provided link cannot decrypt the file anymore despite still being accessible. The link can be generated a lot of times thus, to share with several people, the user needs to generate several links and send each link to the different users to be used once.

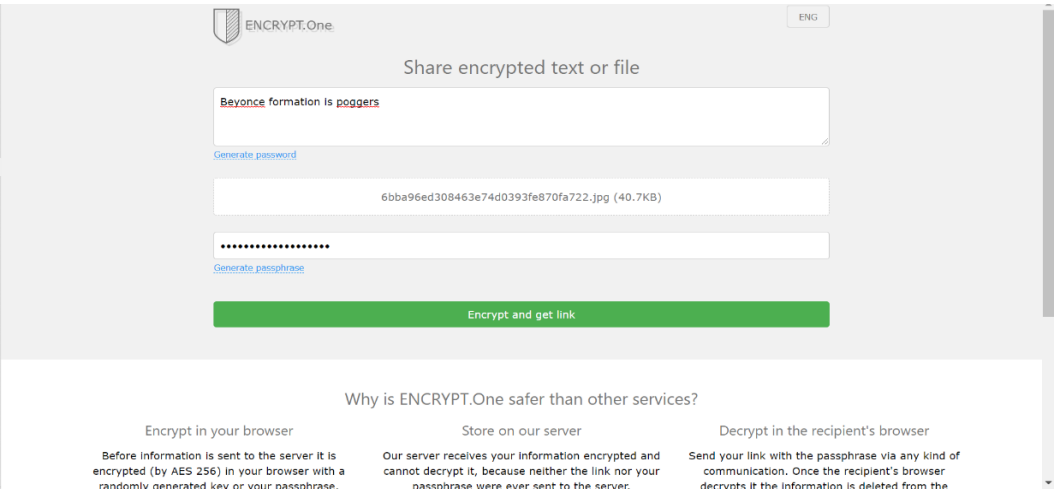


Figure 2.27: Encryption feature in the web-tool

Source: ENCRYPT.One (2016)

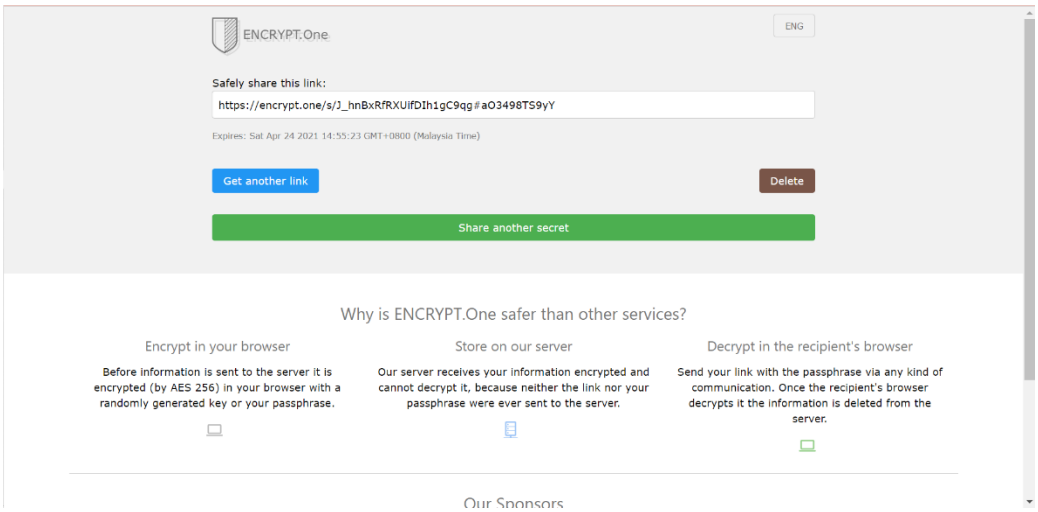


Figure 2.28: Link that can be shared and generated for access to decrypt file

Source: ENCRYPT.One (2016)

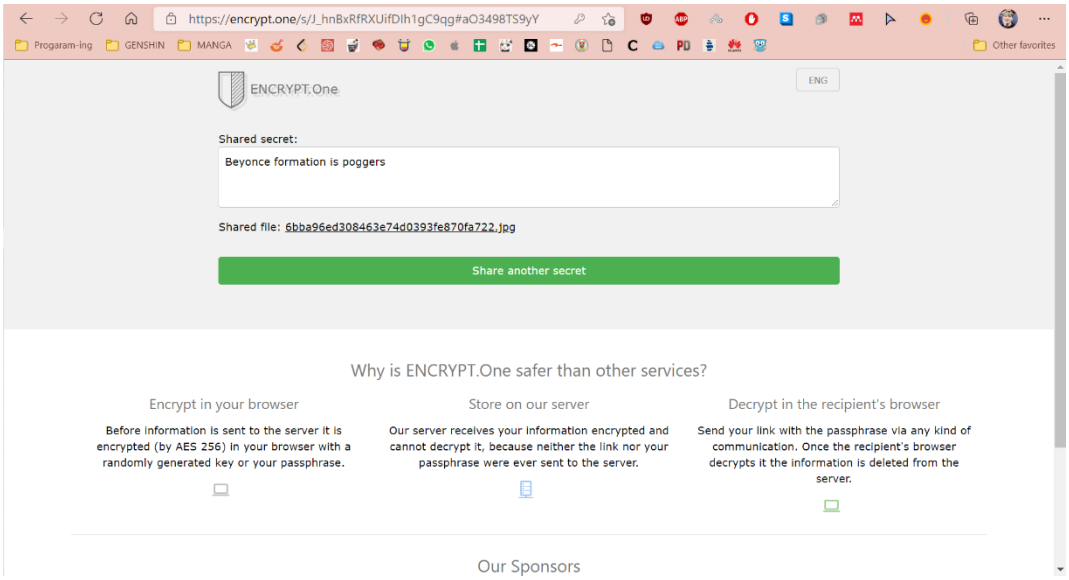


Figure 2.29: Decryption feature in web-tool

Source: ENCRYPT.One (2016)

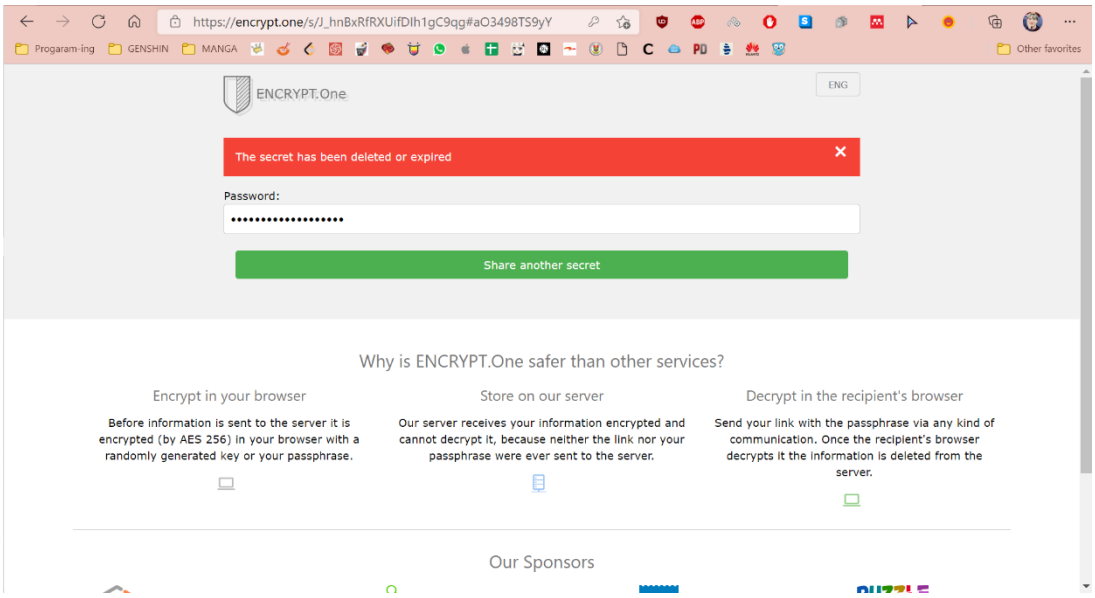


Figure 2.30: The same link accessed but in different tab browsers

Source: ENCRYPT.One (2016)

Users can also immediately delete the file after copying the sharing link. The encrypted file will then be immediately deleted and cannot be accessed anymore. If users choose to not include any passphrase, the file will be accessible without prior password prompting. The file will be kept in its servers for a week only, thus users can access the links to decrypt the files but only if the link has not yet been used. As of now, there is no other way to access the files other than the links given after encrypting.

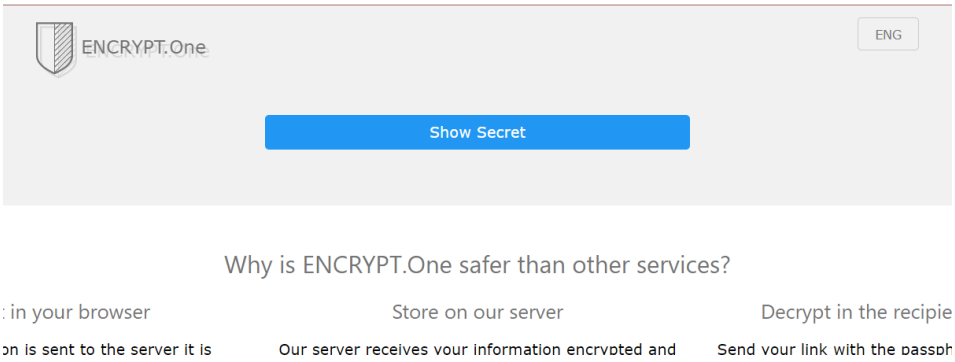


Figure 2.31: Encryption without secret passphrase provided

Source: ENCRYPT.One (2016)

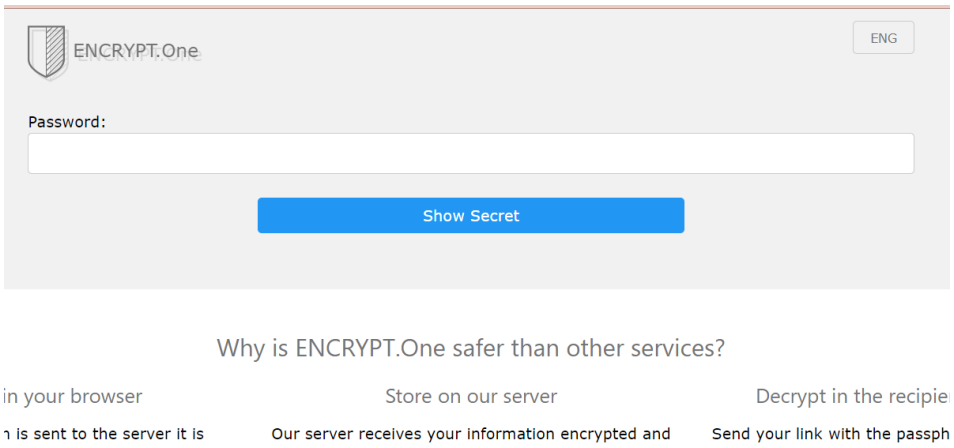


Figure 2.32: Encryption with secret passphrase

Source: ENCRYPT.One (2016)

This web tool is very easy to navigate through and the aesthetics are minimalistic. Unlike other encryption tools, this web tool does not overwhelm users with a variety of encryption algorithms, and modes for users to choose from. This makes it very user-friendly, especially for users that are just beginning to practice encryption. The downside is the decryption feature is only available for the web-tool users since users are given links instead of the encrypted file. As stated previously as well, there is no other way to retrieve the file other than through the generated links. Users cannot revisit the page that provides sharing links after closing the web tool. Thus, users must remember to copy several newly generated links to access or share them later within seven days.

2.2.4 EncDec

EncDec is an online text and file encryption tool that uses AES-CBC with PKCS5 padding. The web tool has a simple design, where the encryption and decryption area are on the homepage and displayed in a table form.

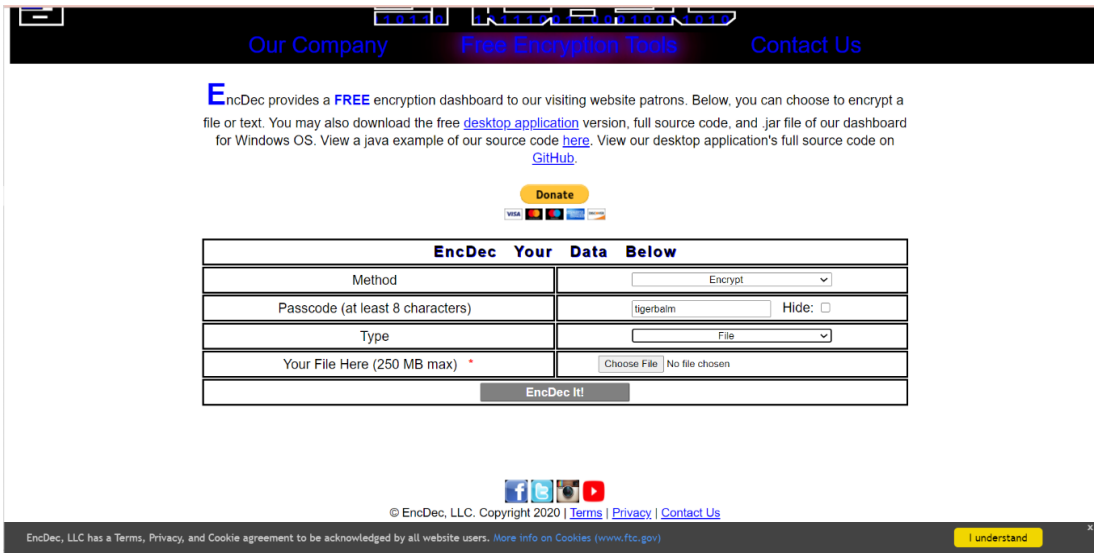


Figure 2.33: Homepage of EncDec web-tool

Source: EncDec (2020)

Users can choose two methods, which are 'Encrypt' and 'Decrypt'. Then, users are required to enter a passcode that has a minimum of eight characters. Next, users pick the type of files that are going to be encrypted, and the row below it changes depending on the choice. If 'File' is selected, a button for file selection will appear. If 'Text' is chosen, a text field will appear, and users can write in it. There is a limit to each type of file: 250 MB maximum size for files and 250 characters maximum for text.

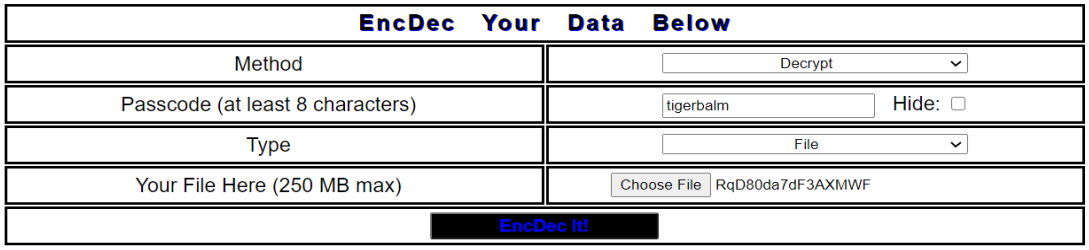


Figure 2.34: The limit size of data for a file

Source: EncDec (2020)

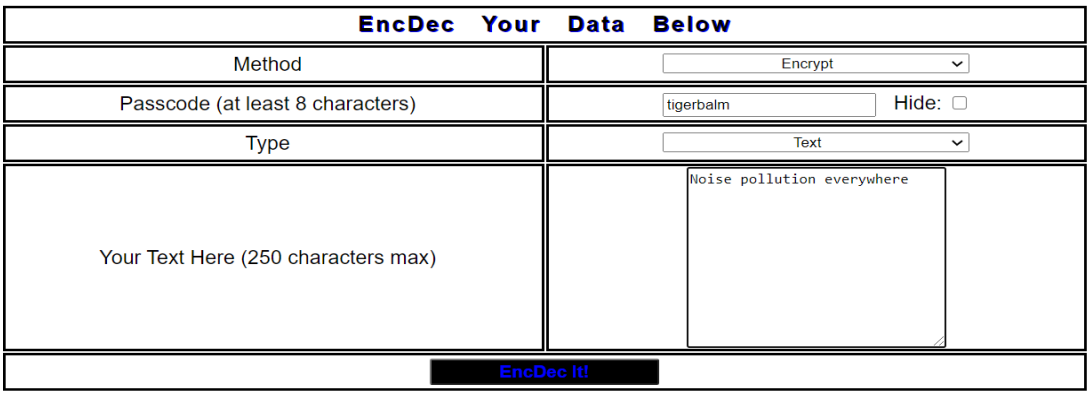


Figure 2.35: The limit size of data for text

Source: EncDec (2020)

Once all the required fields are fulfilled, users need to click on the button 'EncDec It!' and the file will be encrypted and downloaded into the device. Encrypted files do not have a file extension. Decryption also has the same process, and the decrypted file is automatically downloaded into the device.

Overall, the website serves its purpose as a simple online encryption tool. Despite the old-fashioned web design, it functions well and is still able to do cryptography as stated. It has no online file sharing features however, thus users are required to download the file into the device to share it.

2.2.5 Devglan

This web tool explains and teaches users how to use AES for encryption and decryption. It offers both text and file encryption, using AES with its modes (CBC and ECB). On the page, there will be two divisions which are for encryption and decryption.

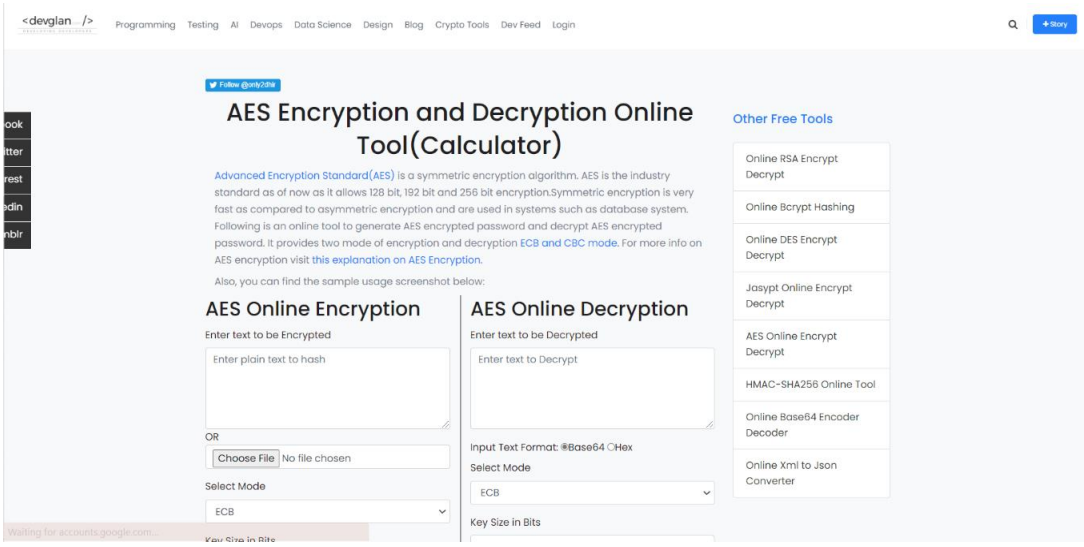


Figure 2.36: Homepage Devglan encryption tool

Source: Devglan (2019)

Users can enter a text or choose a file to be encrypted. After that, a mode must be selected. If the user selects CBC mode, a text field for initialization vector will appear below key size options and users need to fill it with an input of 16 characters in length. Then, users can choose the key size of the encryption algorithm, which are 128, 192, and 256. The secret key length depends on the key size chosen, where it increments in eight characters as the size increases. Initially, it is set to 128 key sizes, and the secret key needs to be 16 characters. This is because the developer does not apply any padding in the encryption tool. Lastly, users need to choose the output text format which is either Base64 or Hexadecimal. Once users click the 'Encrypt' button, an output of the encrypted file or text will be displayed in a text field below it. Users can only copy and paste the encrypted text.

The decryption tool works similarly except it does not provide an option to select files from the device. It is also not capable of decrypting an encrypted file properly except for a text file. For example, it can be seen in Figure 41, a decrypted encrypted image file will appear in weird symbols after encoding it and cannot be retrieved in its original form. It is not downloadable nor can be selected for copy and paste. However, it works fine with text: the decrypted text will be encrypted into Base64 or hexadecimal (depending on the format chosen) and users need to click 'Decode to Plain Text' to see the original text. The final output of text can be selected for copy and paste, unlike on non-text files.

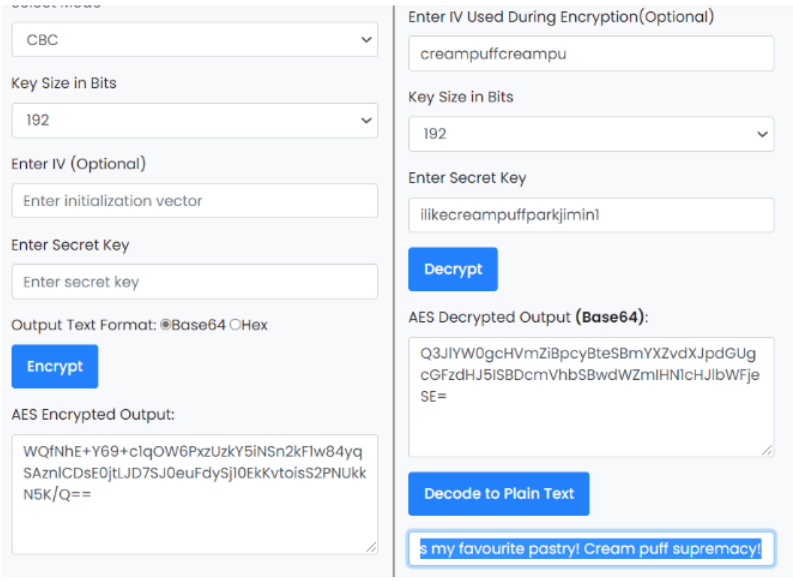


Figure 2.37: Encryption and decryption feature for text

Source: Devglan (2019)

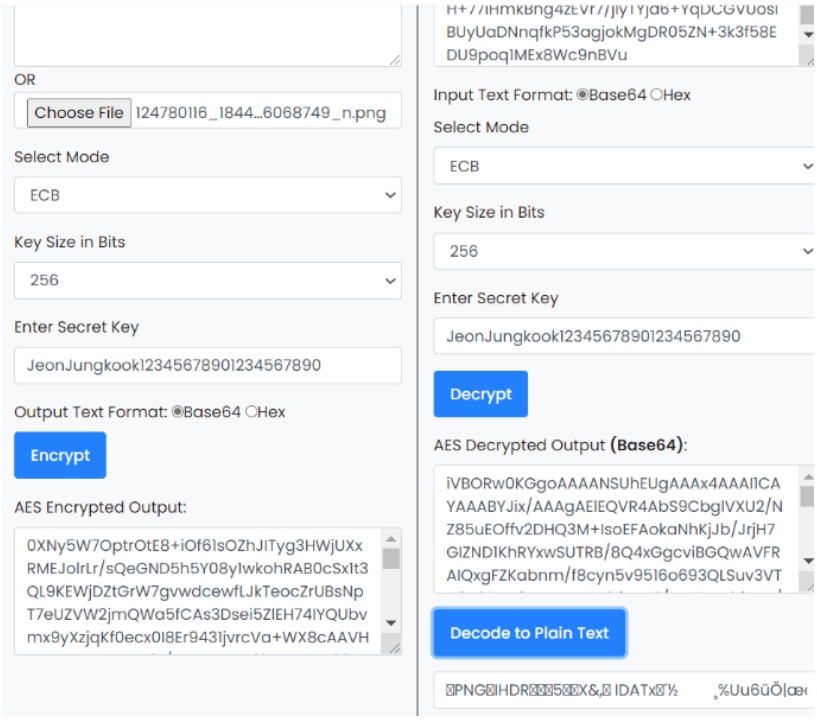


Figure 2.38: Encryption and decryption of an image file

Source: Devglan (2019)

In general, the encryption web tool works well in displaying AES encryption and decryption. The explanation on the website regarding AES and guidance on how

to use the tool is sufficient. Based on the comments, a lot of users are personally trying to understand the algorithm, and the tool is updated from time to time. However, it lacks features that most web tools possess, which are auto downloadable encrypted and decrypted files, and some sort of file sharing. It would be better if users can retrieve decrypted non-text files and download them. Padding would be an excellent addition to the algorithm as well since typing in long characters as the key will be cumbersome for some users to remember.

2.2.6 Comparison of Lightweight Symmetric Encryption Algorithm

Five web-based encryption tools have been observed by discussing the features provided, the interface design, the process flow, strengths, and limitations. The information is further categorized, including a proposed system to address the limitations.

Table 2.2: Comparison of five web-based encryption tools

Criteria	**Free Crypt**	**TxtWizard**	**ENCRYPT.One**	**EncDec**	**Devglan**	**Proposed**
Available features	1. Encryption 2. Decryption 3. Encrypted file analysis	1. Encryption 2. Decryption 3. Text encoding 4. Hash generating tool 5. Text compre	1. Encryption 2. Decryption	1. Encryption 2. Decryption	1. Encryption 2. Decryption	1. Encryption 2. Decryption 3. File retrieval

		ssion tool				
Supported File Encryption Format	File	Text	Text and file	Text and file	Text and file	Text and file
Encryption algorithm	AES, Serpent, SHA-1	AES, DES, Blowfish, RSA	AES-256	AES-CBC	AES	Lightweight symmetric encryption algorithm
Interface design	Simple and minimal	Very simple	Simple, clear, and minimal	Simple, clear, and old-fashioned	Simple, and clear	Simple, clear, and minimal
Multiple file selection	Yes	No	No	No	No	Yes
Functionality	Not satisfying. Some features were broken.	Not satisfying. Some features were broken.	Satisfying.	Satisfying.	Moderate.	Satisfying.
Usability	Easy to use	Easy to use	Easy to use with clear instructions	Easy to use with clear instructions	Easy to use with clear instructions	Easy to use with clear instructions

Speed of encryption	Fast	Fast	Fast	Fast	Fast	Fast
Online file sharing	Available through email	Not available	Available through link sharing	Not available	Not available	Available using a third-party API
Key generation	Auto-generated	Auto-generated	Self-written and auto-generated	Self-written	Self-written	Self-written and auto-generated
Additional features	1. Provides QR code to copy encrypted file key 2. Has separate web encryption tool that offers full client-side encryption	1. Provides various encryption modes (CBC, ECB, CFB, OFB, and PCBC) for the user to choose 2. Provides padding options for	1. Multiple links can be generated for the encrypted file 3. Stores the encrypted file for a week 4. User-friendly	1. Provides PKCS5 padding	1. Provides two encryption modes (CBC and ECB) 2. Allows users to pick a key size 3. Provides both file and text encryption	1. Registered users can retrieve their files in 14 days 2. Users can share encrypted files into their storage services 3. Padding and encryption mode will be included to help users encrypt easier and

		users to choose (PKCS7, and SSL3Padding)			4. Output text format can be chosen (Base64 or Hexadecimal)	more efficiently 4. Users can retrieve their keys within 14 days but must change it immediately to ensure confidentiality 5. User friendly
Limitations	1. File size supported is limited and very small (100 MB) 2. No available feedback features for users to send any feedback or complaints	1. Web-tool is insecure since there is no SSL certificate 2. Only provides text encryption and encryption	1. Encrypted file cannot be decrypted if the link has already been used 2. No other way to retrieve or generate links after closing the post-encryption tab (where it displays the sharing links)	1. No online file sharing provided	1. File decryption does not retrieve the original file in its original form 2. No padding thus users must provide	1. Not a full client-side encryption tool, hence users need to access it online to use

	3. Only file encryption and decryption is provided	3. No description of the additional features provided 4. No declaration on the key generation method (self-written or auto-generated) 5. Some of the additional paddings does not work	5. Decryption can only be accessed with the links provided and it cannot be used links		a key that fits the key size 3. Files cannot be retrieved and downloaded into the device	

The proposed system will provide a web-based encryption tool, where the encrypted files will be saved for 14 days so that users can share and view the file

securely, with extra layer of protection. Users can choose to provide the passphrase or let the system auto-generate it. If it is auto-generated, users need to only submit their password. Their password is used to generate a master key, and the passphrase (if users choose to not provide passphrase). The file can be also shared with other registered users, and these users will receive an email notification about it. A sharing session link will be sent so that they can access it through email, not necessarily need to be logged in but do need to know the access password set by the sender. The sharing session will only last 2 days, and automatically will be deleted. If possible, file sharing features can be expanded by allowing users to share their encrypted files to third-party applications such as Google Drive. It can be possible by implementing the API provided by Google. Another consideration would be to provide a way to help users retrieve their encrypted files if they forget their password or key.

CHAPTER 3

METHODOLOGY

3.1 Introduction

The methodology of this project is constructed based on the waterfall approach. This approach is chosen as it fits with the timeline required for this project and it being a sequential life cycle model. The phases of this project development will be in seven phases. The seven phases in this methodology and are explained as below:

i. **Phase 1: Literature Review**

   The aim for this phase is to study in-depth efficient and lightweight encryption algorithms that are best suited for supporting file encryption in web platforms, in terms of computation speed and storage size. The literature review will be carried out by searching reputable sources such as journals, articles, reports, and research papers online through several known platforms which are Google Scholar, ScienceDirect, ICM Digital Library, ResearchGate, and IEEE Xplore. Quantitative search will be the strategy for this research phase as known lightweight encryption algorithms will be identified and studied to choose which are suitable to be included in this project. Five lightweight symmetric encryption algorithms will be identified by using the purposive sampling technique for further investigation. These algorithms will be identified by searching already known lightweight block ciphers that exist within the industry or academic field. The accessibility of the algorithms will be considered as well as will be used during the experimentation and implementation phase. Their performance in supporting real-world web-based file encryption applications will be further investigated in Phase 3.

ii. **Phase 2: User and system requirements gathering and analysis**

For this phase, questionnaires will be used to collect user and system requirements. Questionnaires are used to collect these data mainly due to the broader target scope being lecturers from a faculty. The sampling method that will be used is simple random sampling as the data will be collected from lecturers in the Faculty of Computer and Informatics, and the questionnaire will be shared randomly online using email. Sample size, *n* would be around 49 lecturers, according to the sample size formula used below (Hamberg and Tiahrt, 1975):

$$n = N \times \frac{\frac{Z^2 \times p \times (1-p)}{e^2}}{[N - 1 + \frac{Z^2 \times p \times (1-p)}{e^2}]}$$

where N (population size) is 56, e (margin of error) is 5%, p (sample proportion) = 0.5 and z (z-score) is 1.96 (95% desired confidence level). The questionnaire contains two sections, with a total of 17 questions. There are two open-ended questions, and the others are close-ended questions. The first section focuses on user experiences; investigating the lecturers' current experience of using web-based encryption tools. The second section focuses on the limitations of the current existing web-based encryption tools to identify the main features that are expected in a web-based encryption tool. After obtaining the results, the data will be analyzed through statistical analysis with the output presented in a visual graph and chart.

iii. Phase 3: Implementation of Lightweight Encryption Algorithm

This phase aims to and select the best performing lightweight encryption algorithm with fast computation speed and consumes less storage size. Three lightweight symmetric encryption algorithms selected in Phase 1 will be implemented and compared to their performance by evaluating them with several different sized document files. Running environments would be in C/C++ in web application platform as offered feasibility of open-source encryption algorithms. Visual Studio 2019 will be used as the platform. It is a commonly used IDE, and it includes facilities that allow developers to monitor application performance such as CPU usage, memory usage, and others. The computation speed will be measured by recording the execution time of each

algorithm meanwhile the storage size or memory usage will be observed through the diagnostic tools built-in in Visual Studio 2019. Another criterion to be tested is the confidentiality and integrity of data. The Avalanche effect will be used to dictate how secure the algorithms are to protect the encrypted files. The Avalanche effect is a concept in cryptography where a slight change of the input value in a message or key, will cause a big change towards the ciphertext (Witoolkollachit, 2016). According to the Strict Avalanche Criterion (SAC), a good and secure cryptographic algorithm should have an avalanche effect of above 50%. This can be used to measure data confidentiality. Meanwhile, data integrity can be measured by comparing the difference of image file sizes, before and after encryption, including after decrypting the encrypted images. The images will also be compared visually. At the end of the experimentation, the best candidate would be the algorithm that shows the least usage of memory size with the capability of encrypting large files within a small amount of time, including having an avalanche effect above 50% and minimal compression of files.

Table 3.1: Minimal desktop specifications for experiment implementation

Desktop specifications	Details
Operating Systems	Windows 11 Home Single Language
Version	21H2
CPU	Intel ® Core (TM) i5-8265U CPU @ 1.60GHz 1.80GHz
RAM	20GB
Disk space	20GB
System type	x64-based operating system, x64-based processor

iv. Phase 4: Design of web-based file encryption tool

This phase aims to design a web-based file encryption tool that helps in protecting the confidentiality and integrity of data in cloud storage at rest.

System design will be done using Data Flow Diagram (DFD) and Entity Relationship Diagram (ERD). The main module that will be embedded with the lightweight encryption algorithm is the "file encryption and decryption" module as that module handles the cryptographic process. Other modules include the user account management, file management, and sharing sessions management. The design of the interface will be leaning towards a minimalistic and easy-to-understand design where the goal is that users can understand the website without needing a comprehensive guide. It is expected to achieve a secure and systematic DFD and ERD design, including a simple user interface design for this web-based encryption tool.

v. Phase 5: Implementation of web-based file encryption tool

This phase is the development of the web-based encryption tool with lightweight encryption. The main languages that will be used for this web development is PHP, HTML5, CSS3, and JavaScript. PHP is used for back-end development and application of the selected encryption algorithm. JavaScript will be for front-end development with some HTML and CSS for the aesthetic. MySQL will be used as the database and will be accessed through a PHP development environment called XAMPP. It will be integrated with the browser via PHP programming. The server that will be used is Apache, as it is included in XAMPP thus making it easier to be integrated. After completing the web tool, a web hosting service will be used to set up and make the web tool available for public use. The web-hosting service will be Go Daddy Malaysia as its packages are affordable and it is a trusted well-known web hosting service. It is expected for the web-based encryption tool to be fully functional, especially the features involving cryptography as these are the main features of this web tool. Besides that, the web tool must be easily accessible, easy to use, and has a smooth user interface.

vi. Phase 6: Testing and Evaluation

This phase aims to test and evaluate the performance of the developed lightweight web-based encryption tool in terms of user experience using the System Usability Scale (SUS). SUS is used compared to other methods since it provides a high-level subjective view on the usability of a system. The web tool should provide a better experience to users for this project to be

successful. Before SUS, unit testing and integration testing will be done to indicate if the web tool is ready to be tested by users. Testing and evaluation will be done towards the lecturers of the Faculty of Computer and Informatics with a sample size of around 15 lecturers. Lecturers will be asked to test the web-based encryption tool and answer a SUS form, alongside following an instruction sheet to simultaneously do User Acceptance Test (UAT). They will be asked to provide any feedback, if necessary, at the end of the form to identify any features that are having issues or not executing as expected. The result of this phase will be analysed using statistical analysis and be presented using a graph and chart. It is expected to achieve good feedback regarding the functionality and design of the web tool.

vii. Phase 7: Report writing and documentation

This is the last phase of this project. A report and proper documentation will be written and submitted.

3.2 Methodology

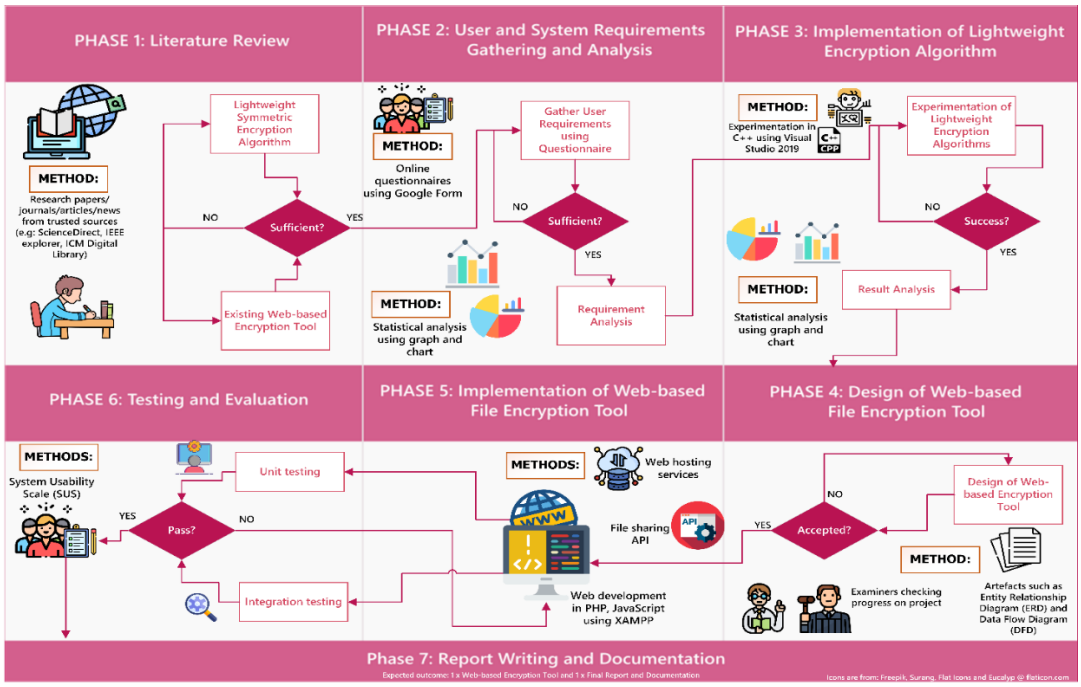


Figure 3.1: Methodology diagram

3.3 Conclusion

The methodology is separated into seven phases and follows the waterfall approach. The first phase includes a literature review, where important topics will be further studied to identify the best lightweight encryption algorithm and implementation of file sharing and data integrity. The second phase involves distributing a questionnaire to target users to identify user requirements and evaluate the key features of this web application project. The third phase is the implementation phase where the lightweight encryption algorithm will be implemented in an experiment to pick the best algorithm to be used in the project. The fourth phase involves the system design of the project, which are presented using diagrams such as DFD, and ERD. The fifth phase is the development of the web-based file encryption tool, where the best lightweight algorithm from the experiment done in phase 4, will be implemented into the web application. The sixth phase is the testing and evaluation, in which target users are asked to test and evaluate the web application using SUS to see if the system meets the requirements. Lastly, the seventh phase is report writing and documentation that will be submitted once finished.

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 Introduction

This chapter discusses the analysis of this project's system and design, where the data collections from the user requirements gathering will be analyzed and presented using graph and charts. After that, the system design will be presented through artefacts such as ERD, DFD, data dictionary and User Interface (UI) design.

4.2 Data Collections

Questionnaires were given out to target users, which are lecturers of FKI via Google Form – an online survey administration tool. The form was distributed to the target respondents via email and personal messaging platforms. The questionnaire consists of two sections: Section A User Experience and Section B Expectations and Suggestions. Participants were required to answer a total of 17 questions, where Section A has seven questions and Section B has 10 questions.

4.3 Questionnaire Findings

The questionnaire was sent to 56 lecturers of the Faculty of Computer and Informatics and the target population is 49 lecturers. However, the target population is not achieved and only 35 lecturers were able to participate. This could be due to many reasons, one of it is that some lecturers were too occupied to spare time to

answer the questionnaire. There could be a possibility that the lecturers were not interested in this topic. The questionnaire was held out for six days, thus it could be due to the short amount of data collection time.

4.3.1 Respondents Experiences with Web-based Encryption Tools

Section A of the questionnaire aims to observe current user experience in using web-based encryption tools that can be extracted as user requirements. Figure 4.1 shows that most respondents (97%) have never used a web-based encryption tool. However, one respondent stated a website which is a browser-based file encryption and decryption provided by Meixler Technologies Incorporated (MTI). It is further asked if the web-tool provides multiple file selection, and the result showed the feature is not provided. These limitations can be solved by including multiple file selections on the encryption tool.

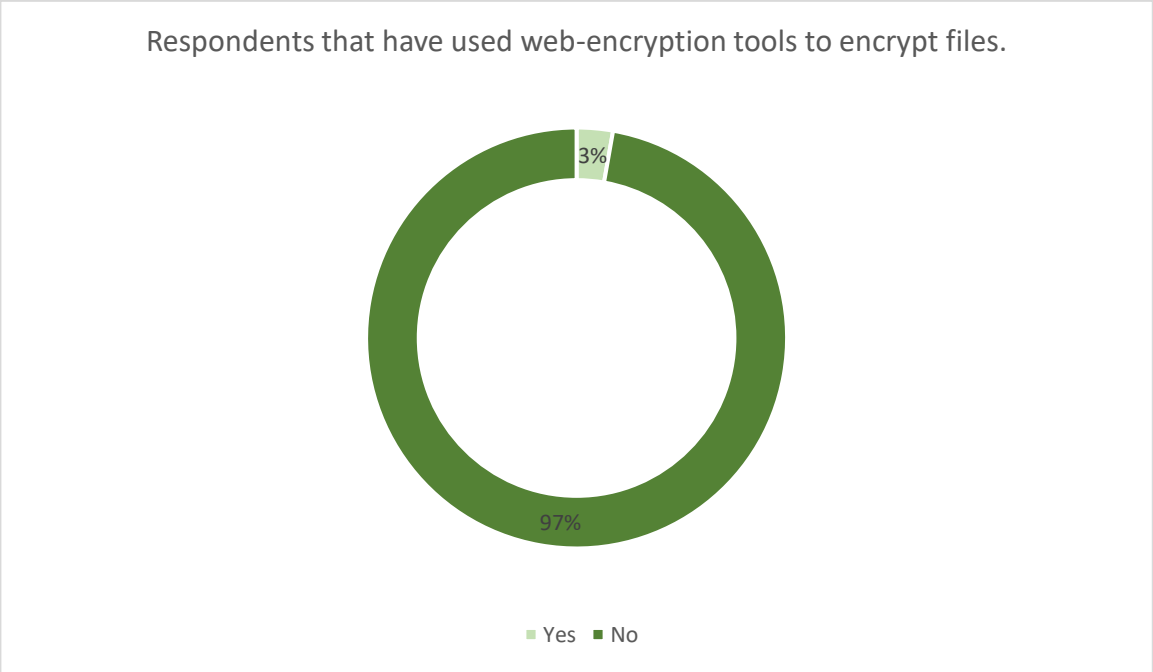


Figure 4.1: Respondents that have used web-encryption tools

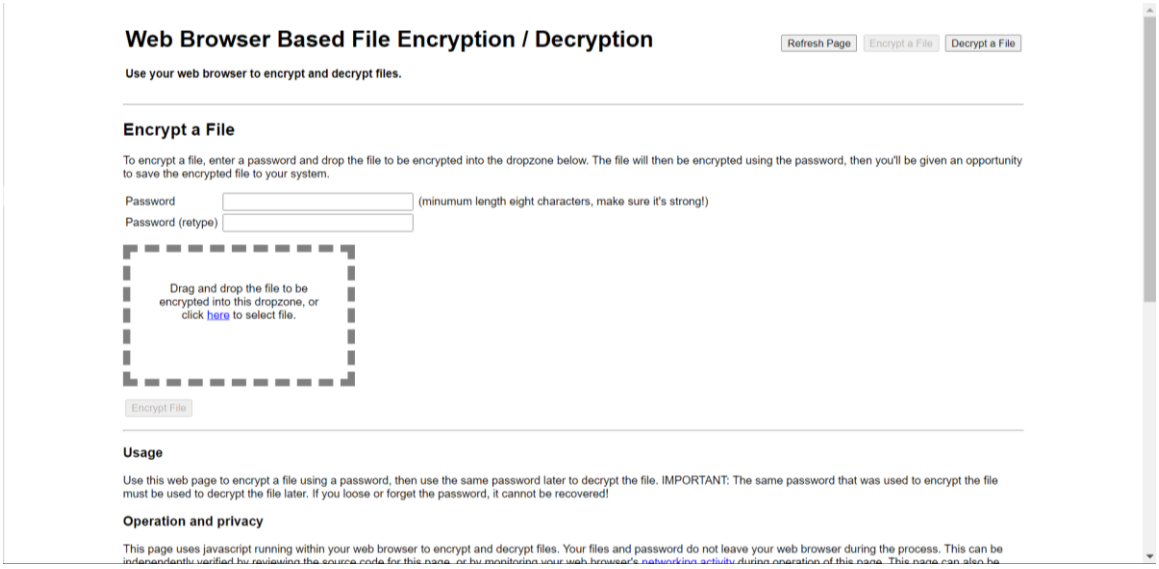


Figure 4.2: Web Browser Based File Encryption/Decryption by MTI

Source: MTI (2021)

Respondents are then asked about the type of files they often encrypt. Figure 4.3 reveals that the most encrypted files are document files (61.8%), and images (16.3%). The average time taken for most respondents to encrypt files is under 5 minutes (48.5%) as can be seen in Figure 4.2. The longest time taken by several respondents (11.4%) to encrypt files is more than 20 minutes. The solution for these limitations can be solved by implementing a lightweight encryption algorithm into the encryption and decryption process.

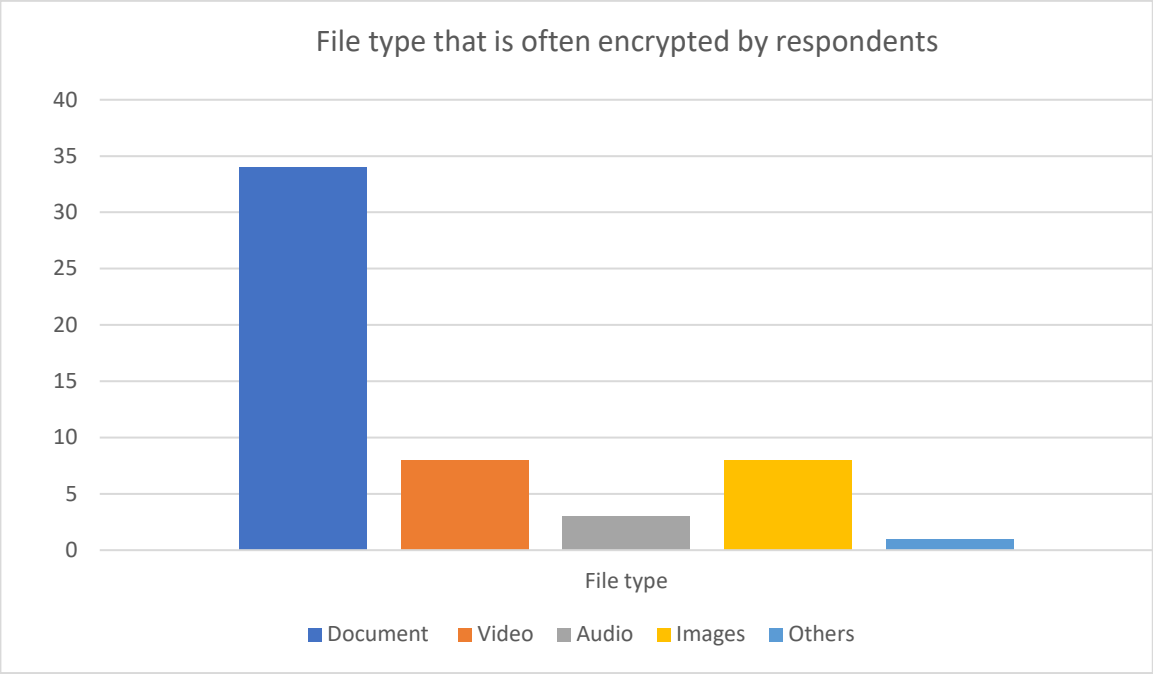


Figure 4.3: File type that is often encrypted by respondents

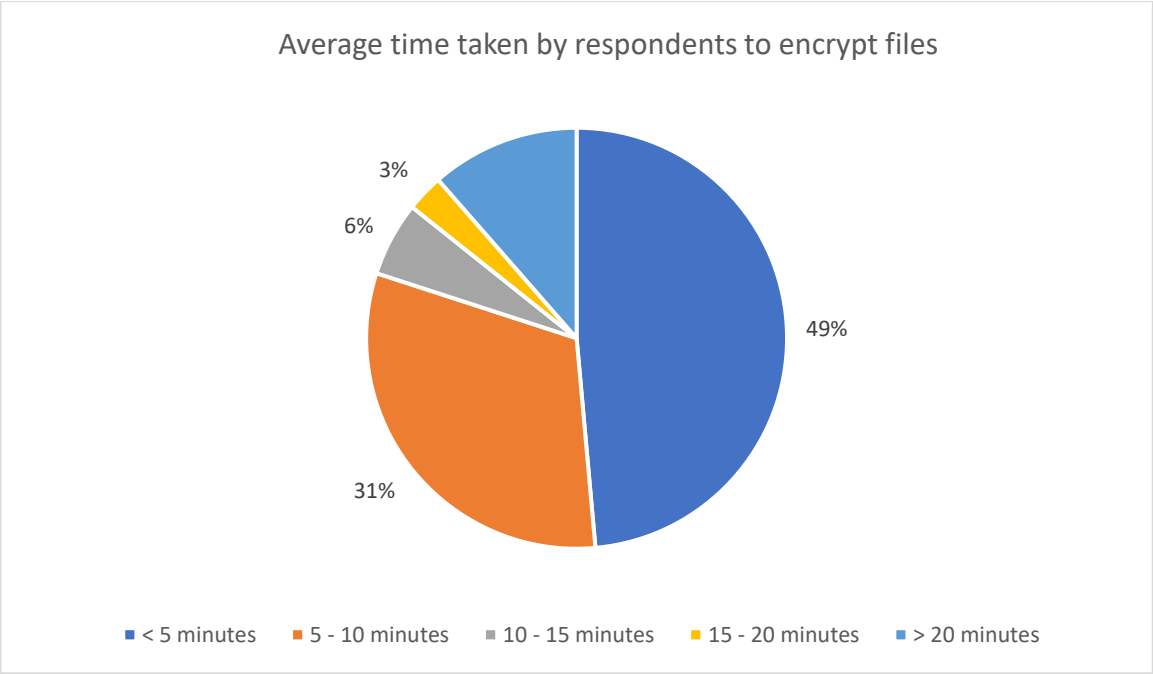


Figure 4.4: Average time taken by respondents to encrypt files

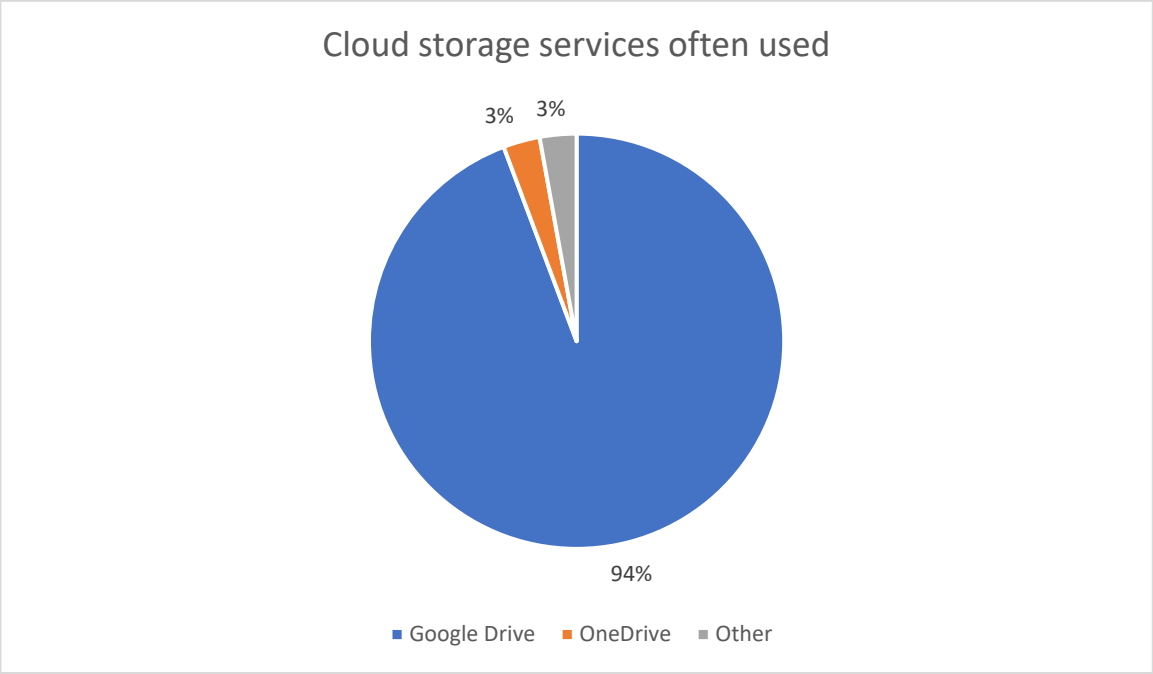


Figure 4.5: Cloud storage services often used by respondents

This project includes file sharing with storage services thus, respondents are asked regarding their most used storage services. Based on Figure 4.5, Google Drive is the most often used storage services among respondents (94%). Most respondents (71.4%) stated that their current method of uploading encrypted documents is convenient. The web-based encryption tool can include Google Drive API to allow users to export encrypted files into their Google Drive account.

The last question on this section asks respondents regarding the limitations in web-based encryption tools. Figure 4.6 shows that most respondents (42%) agreed that current web-based encryption tool lack in online file sharing, followed by slow encryption for multimedia files (31%). The limitation on online file sharing can be solved as it is stated above that an API can be included to allow users to export their encrypted files into their Google Drive account, or an online account can be made so that users can share files amongst themselves.

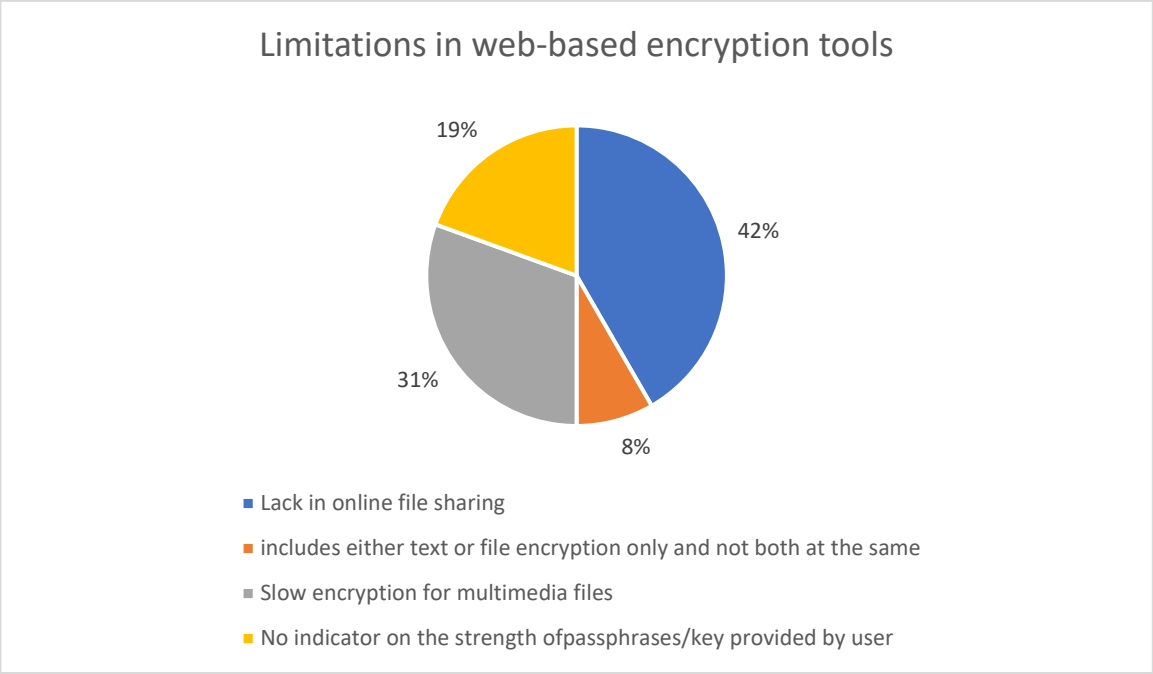


Figure 4.6: Limitations in web-based encryption tools

Section B involves the user expectations towards web-based encryption tools, where it is to investigate the preferred features to be included including expected performance of the web-tool. The first two questions are to find out user expectations of the system. Referring to Figure 4.7, respondents are expecting file encryption/decryption the most (68.6%), followed by text encryption/decryption (42.9%) and key generator (45.7%). The least expected feature is file retrieval (22.8%). Based on the findings, the main features that can be considered during development is file encryption/decryption, file sharing and key generator.

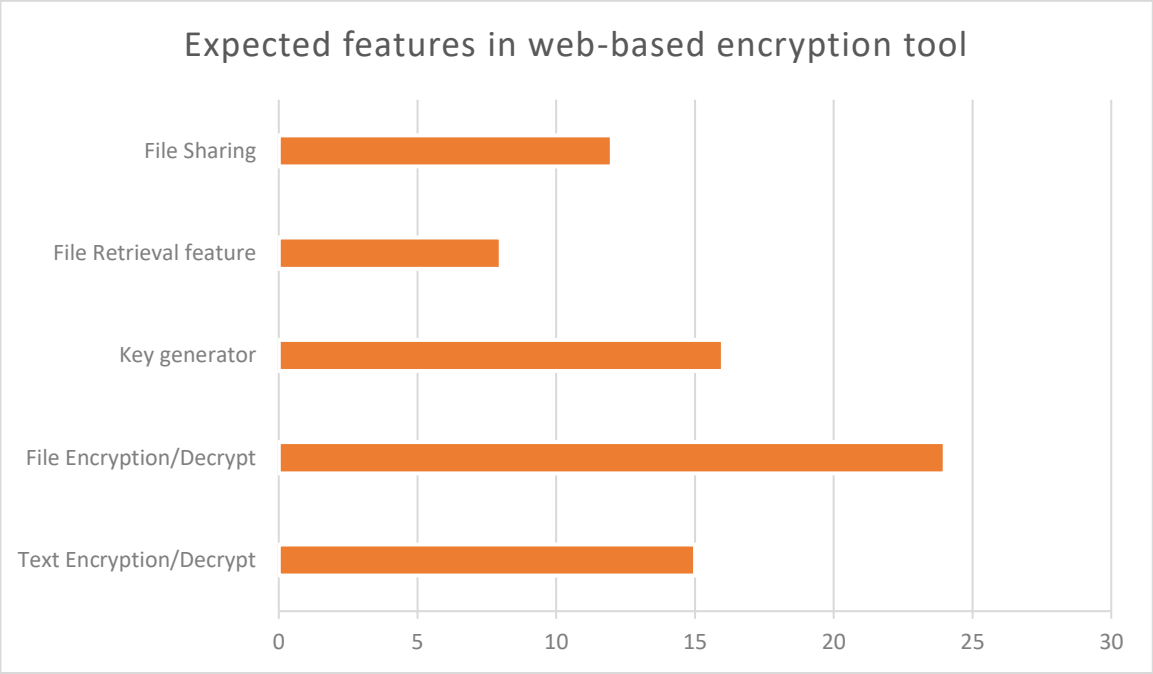


Figure 4.7: Expected features in web-based encryption tool

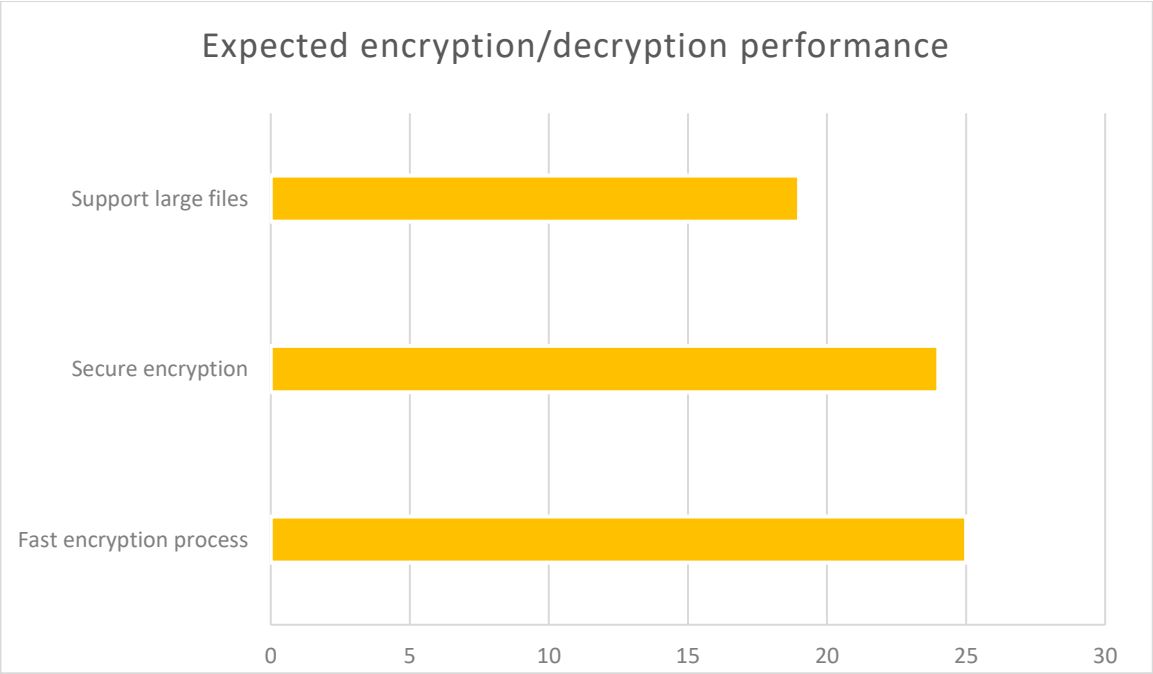


Figure 4.8: Expected performance of encryption and decryption service

The next question asked respondents on the expected performance for encryption and decryption process. The results can be used as criteria in measuring the performance of the web-tool during testing phase. Based on Figure 4.8, each of the options are chosen by all respondents which suggests that users expect the web-tool to achieve all the expected performance on the encryption and decryption process. These requirements can be achieved by implementing the appropriate lightweight encryption algorithm into the encryption/decryption process.

The next several questions are included to identify the preferred features to be included in the system. The questions are meant for these features: key generator, multiple file selection, retrieval feature, file storage and key storage. For the key generator, most respondents prefer for it to be auto generated (68.6%) rather than self-written (34%). However, most respondents (91.4%) agree for both options to be included. For multiple file selection, most respondents (85.7%) prefer the multiple encrypted files to be compiled into a ZIP folder to be downloaded. Next, majority of the respondents agree that a retrieval feature is needed in the system (99%), and the feature needs to have a time limit (74%). Respondents prefer to have an account to retrieve the files (62.8%), rather than through a query using File ID (40%). Lastly, most respondents agree that there should be a time limit for retrieving keys of encrypted files (88.5%). Based on these findings, the system will include both ways for users to provide the key. Besides that, since the system can include multiple files selection, where the encrypted files will be compiled into a zip file for users to download. A retrieval feature can also be added to meet user requirements, including a time limit for the files to be retrieved. Key retrieval can be added with a time limit as well. The system will allow users to have an online account to do the mentioned requirements.

4.4 System Design

This section will be divided into three subtopics, which are Entity Relationship Diagram (ERD), Data Flow Diagram (DFD), Data Dictionary and User Interface Design.

4.4.1 Entity Relationship Diagram (ERD)

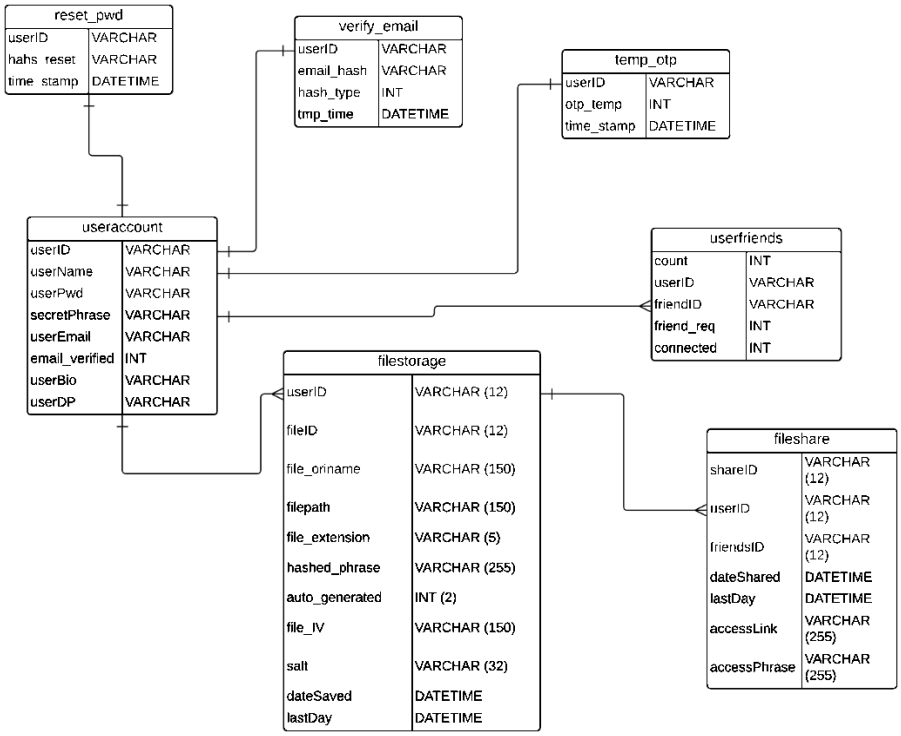


Figure 4.9: The ERD of the system

4.4.2 Data Flow Diagram (DFD)

This section will include artefacts for system design such as Content Diagram, DFD Level 0, and DFD Level 1.

4.4.2.1 Content Diagram

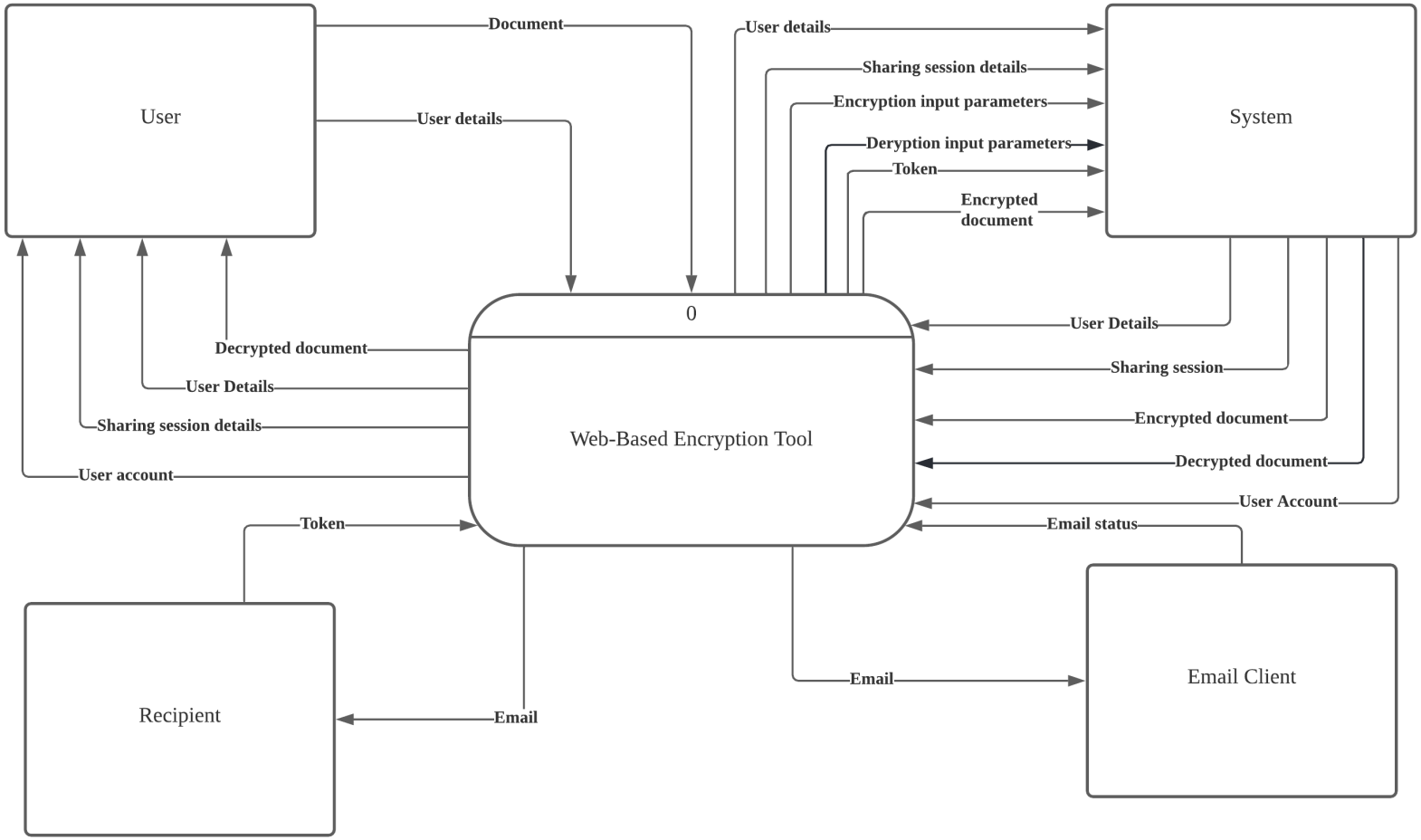


Figure 4.10: The Content Diagram

4.4.2.2 Data Flow Diagram (DFD)

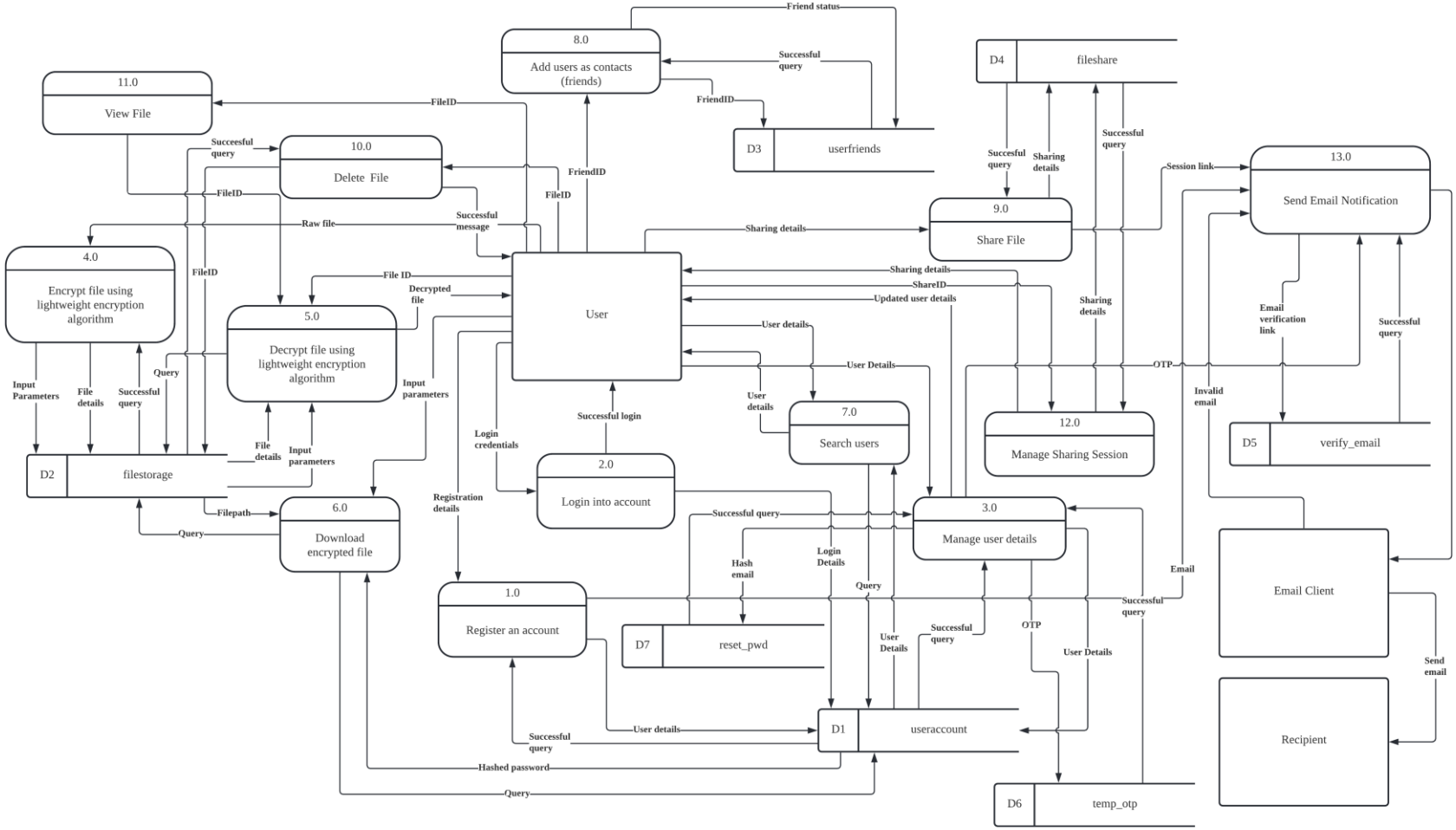


Figure 4.11: The DFD Level 0

4.4.2.3 Data Flow Diagram Level 1 (DFD Level 1)

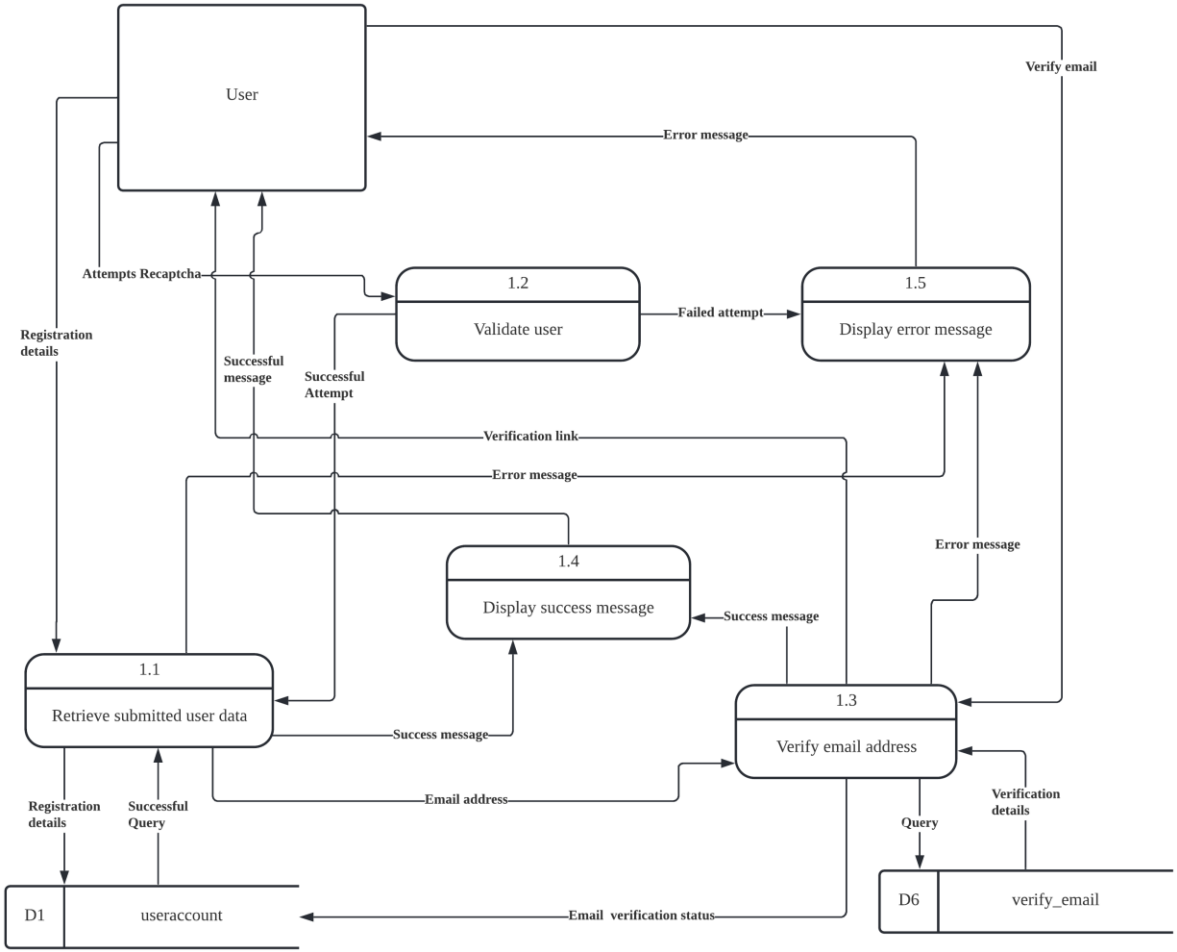


Figure 4.12: Data Flow Diagram Level 1 1.0

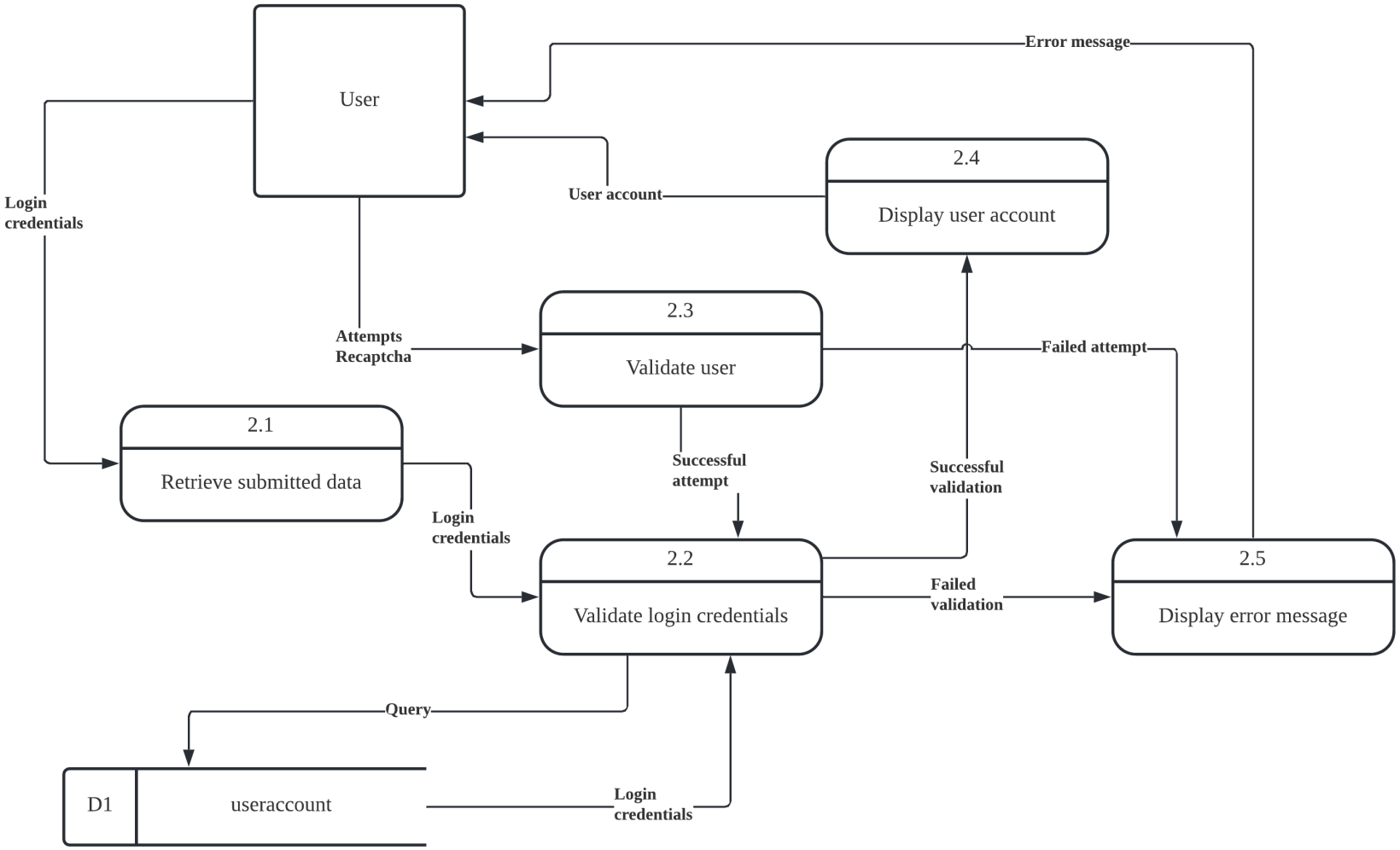


Figure 4.13: Data Flow Diagram Level 1 2.0

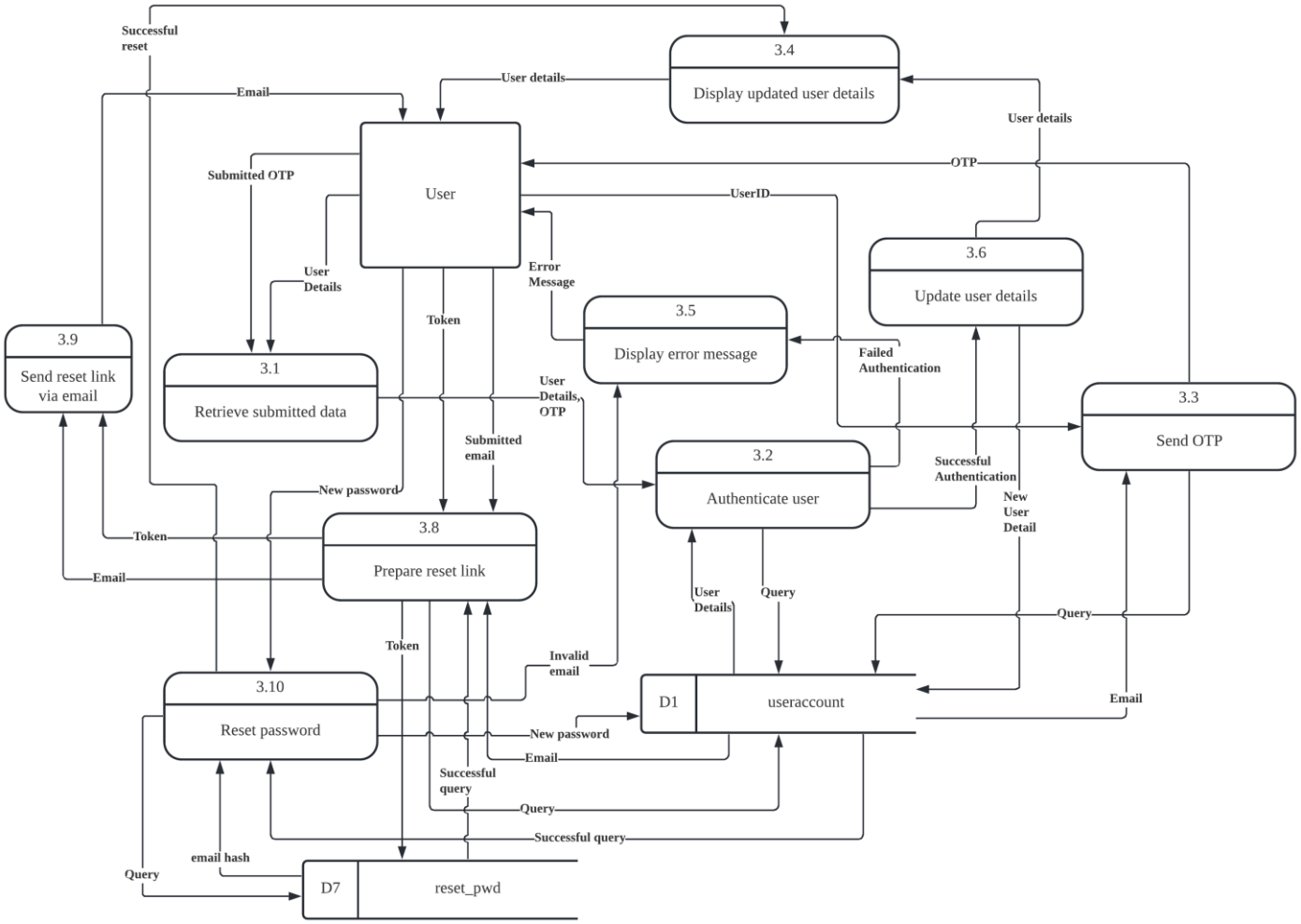


Figure 4.14: Data Flow Diagram Level 1 3.0

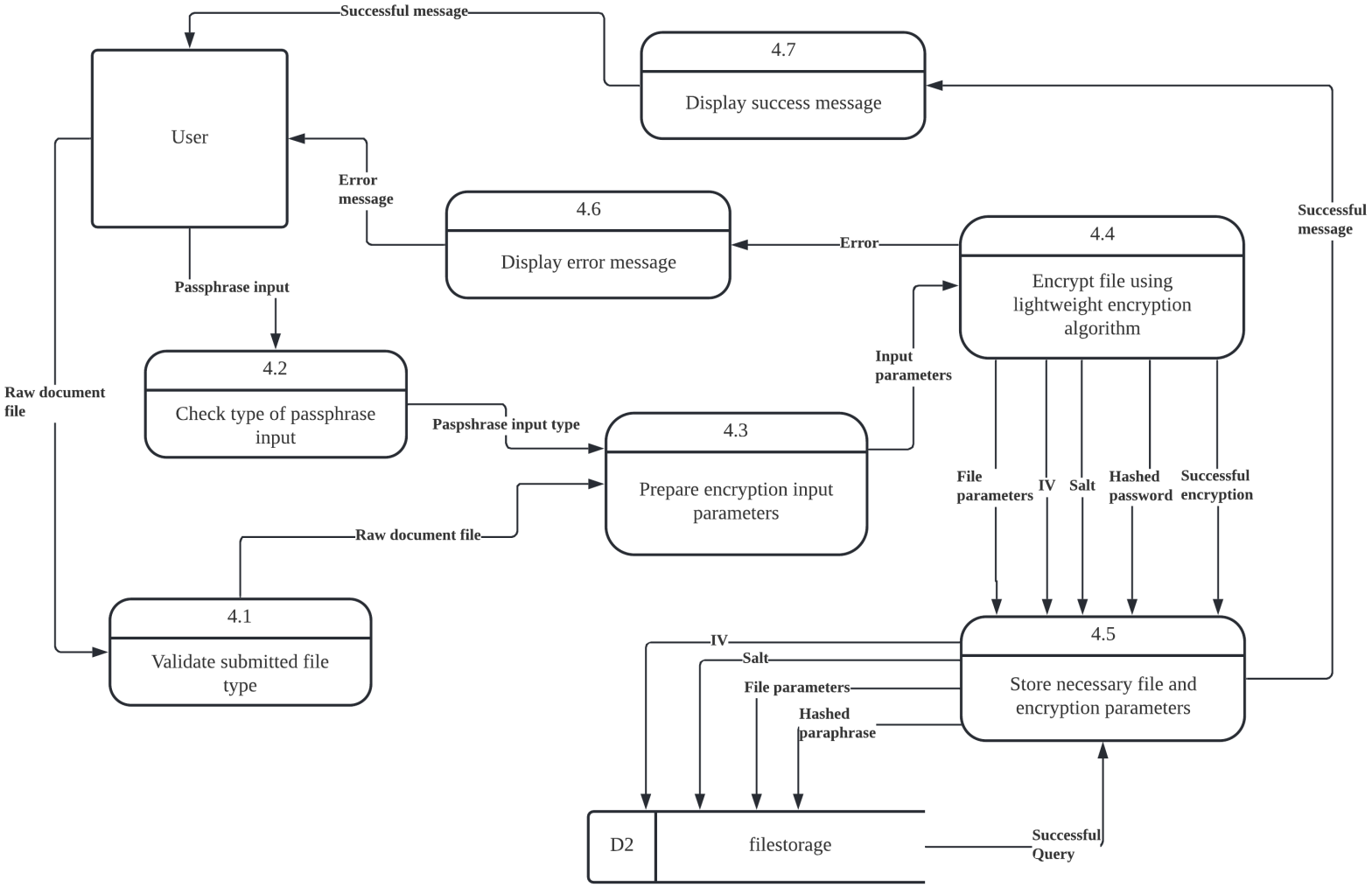


Figure 4.15: Data Flow Diagram Level 1 4.0

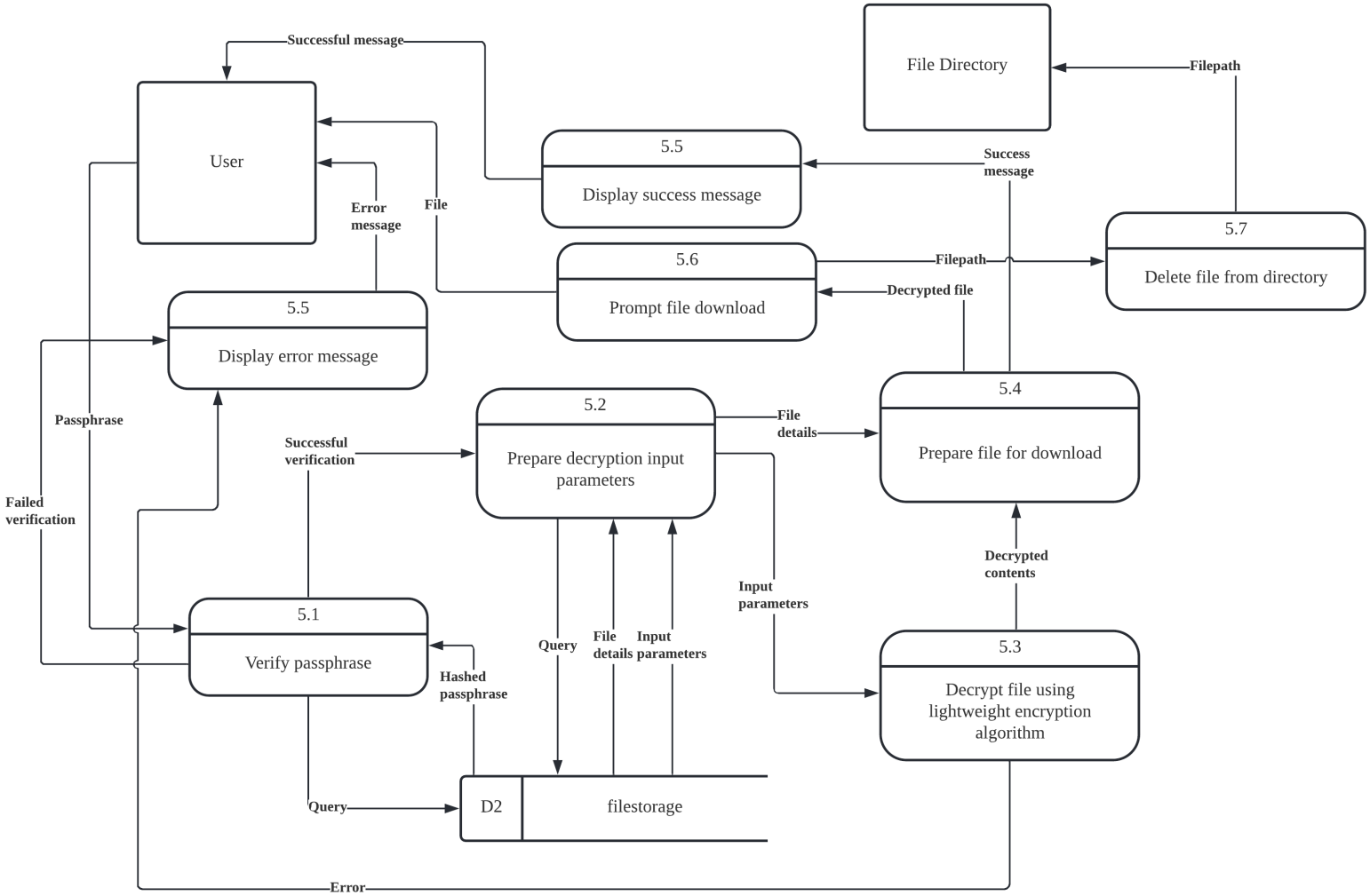


Figure 4.16: Data Flow Diagram Level 1 5.0

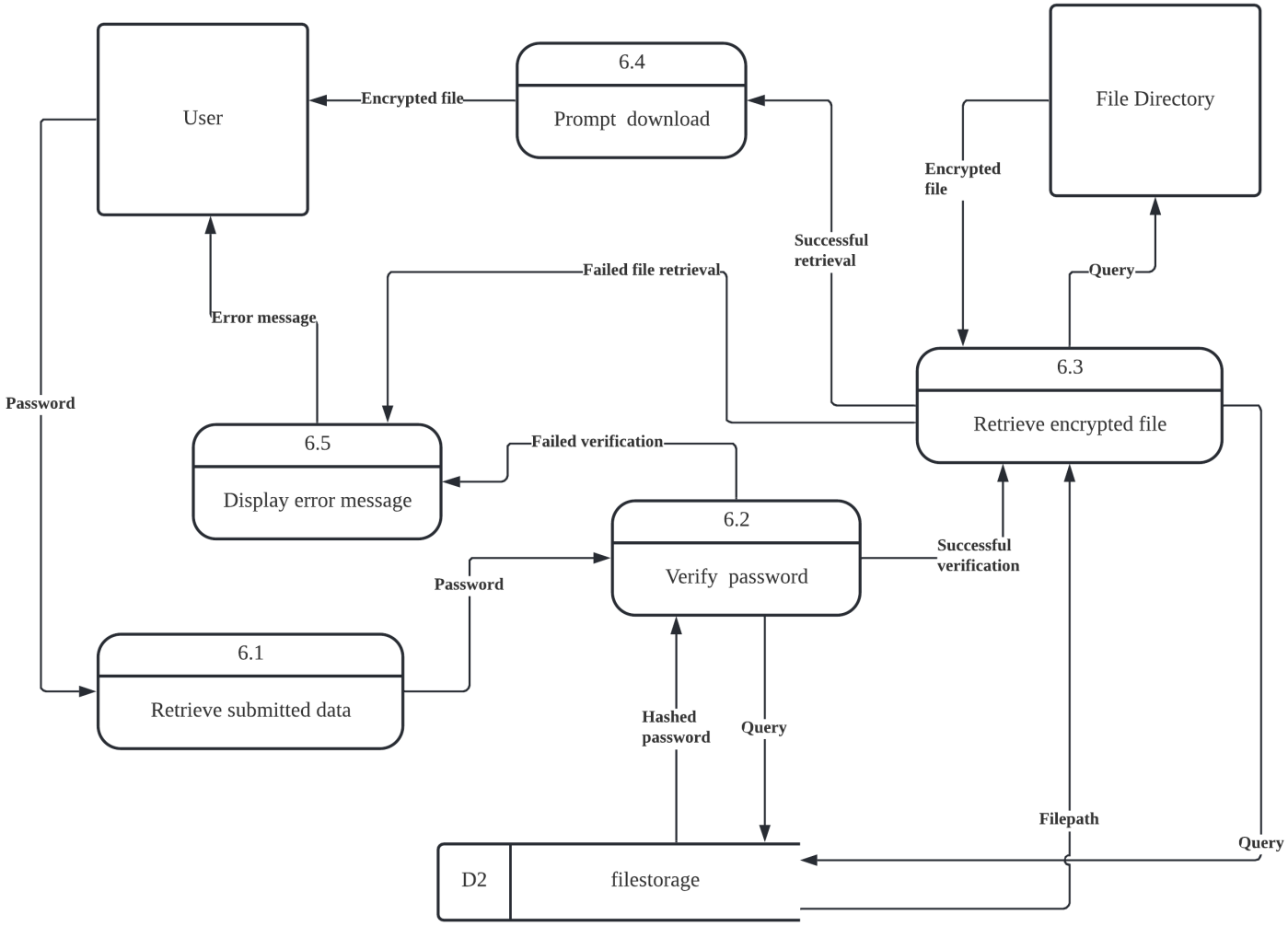


Figure 4.17: Data Flow Diagram Level 1 6.0

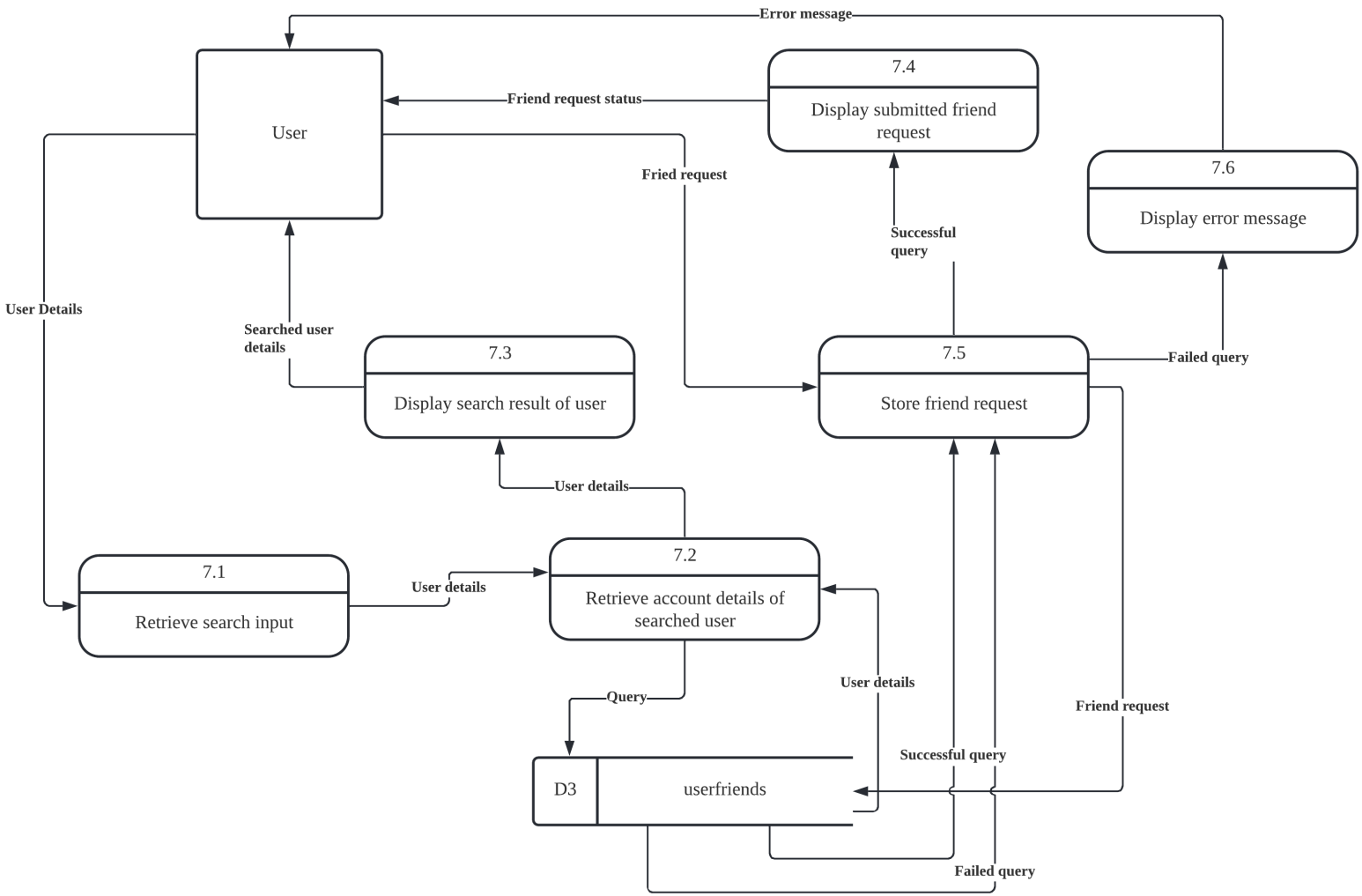


Figure 4.18: Data Flow Diagram Level 1 7.0

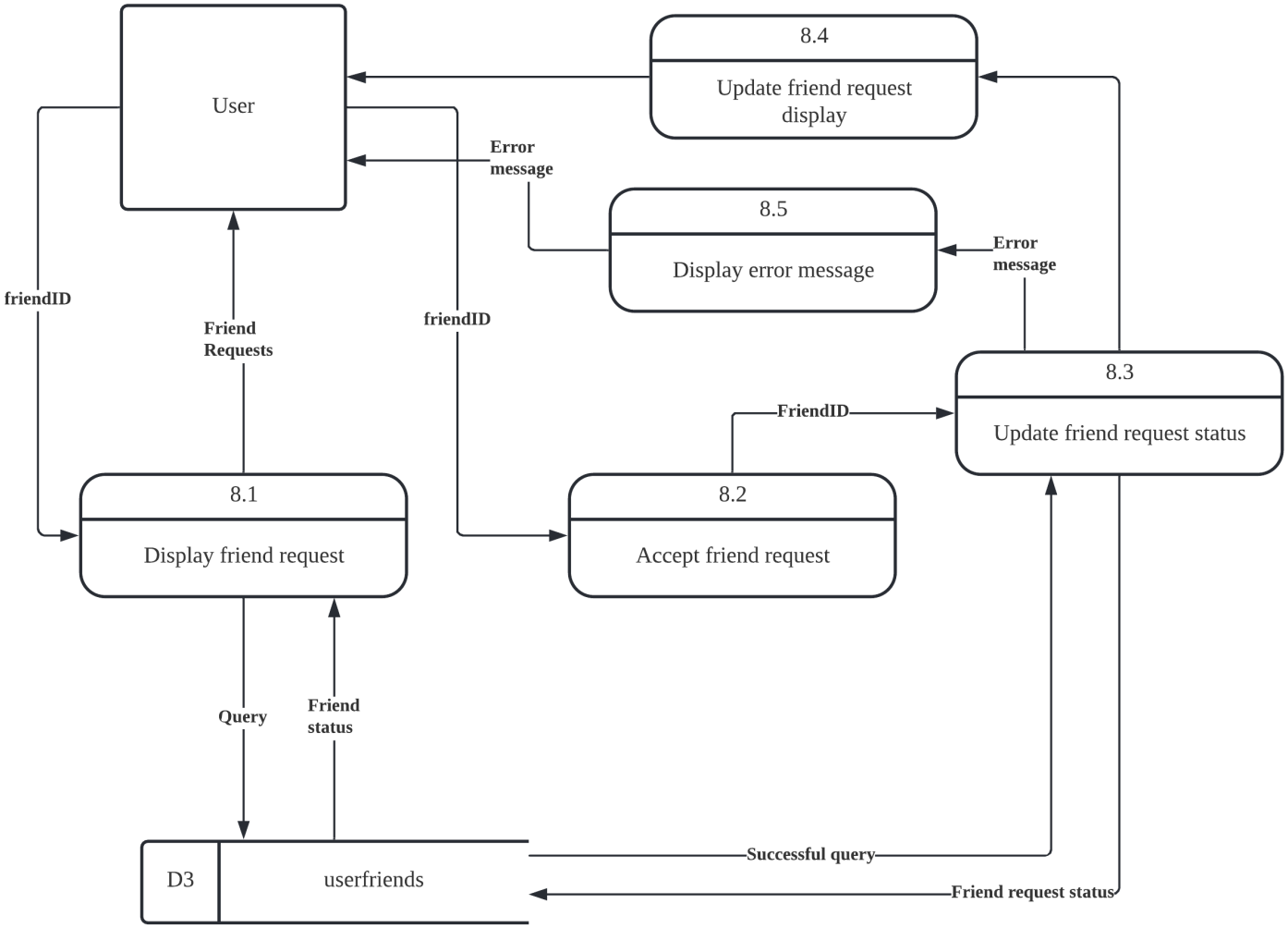


Figure 4.19: Data Flow Diagram Level 1 8.0

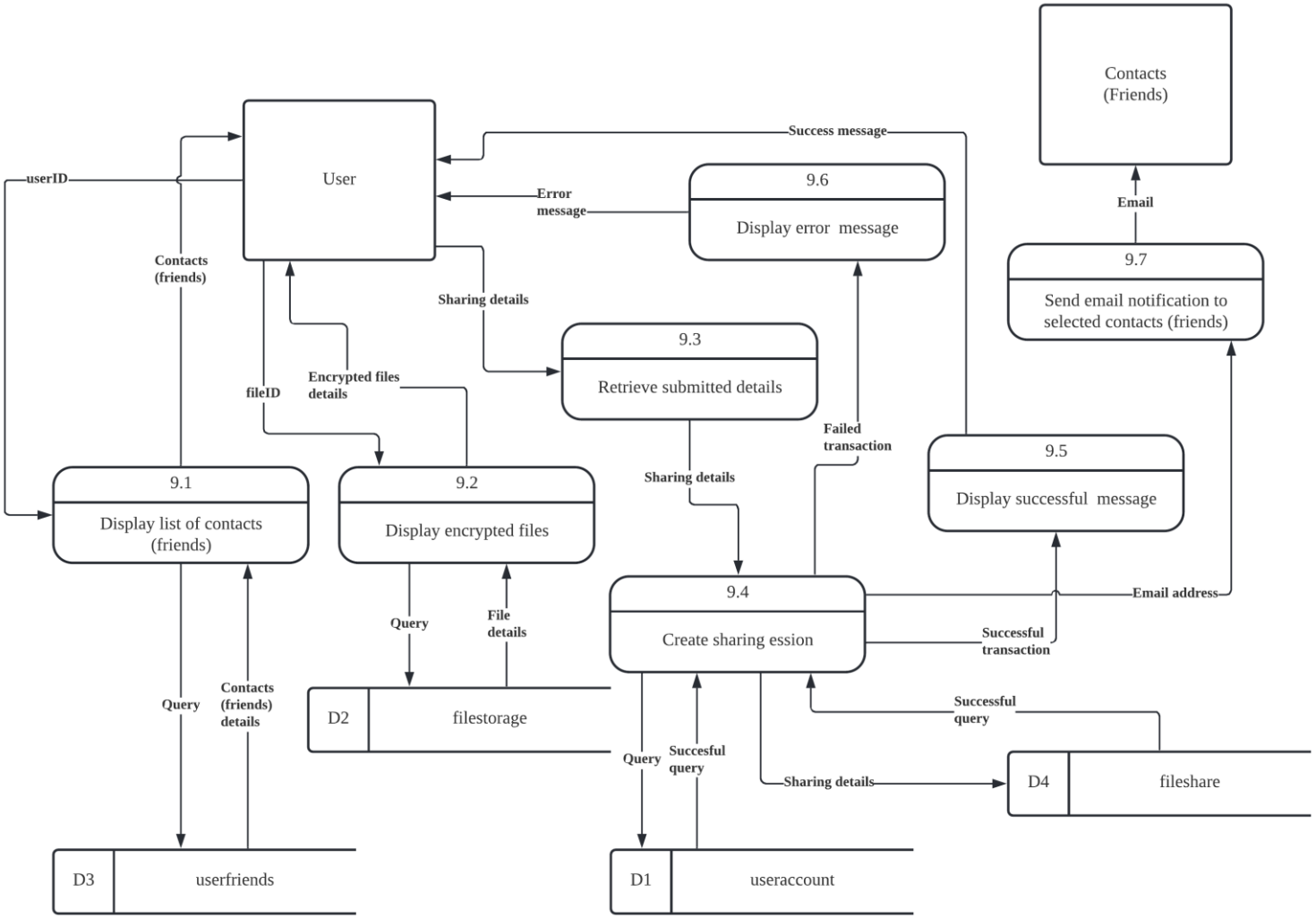


Figure 4.20: Data Flow Diagram Level 1 9.0

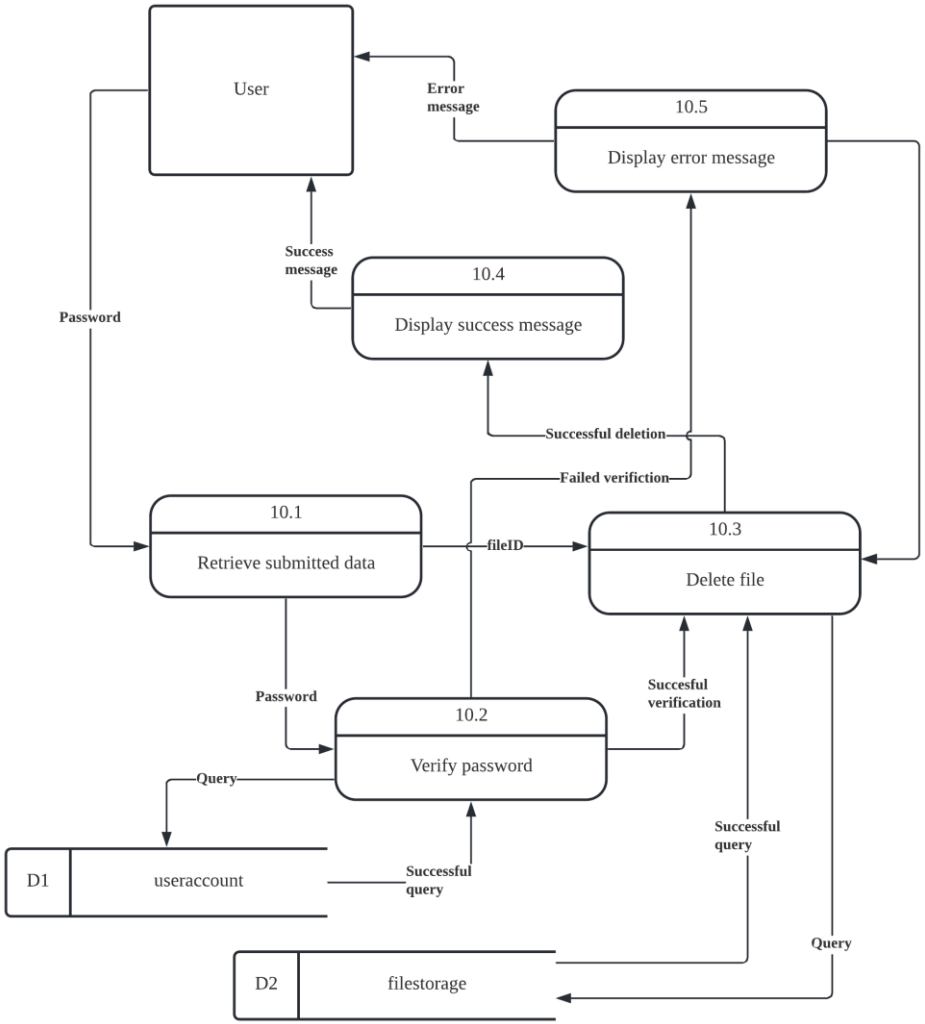


Figure 4.21: Data Flow Diagram Level 1 10.0

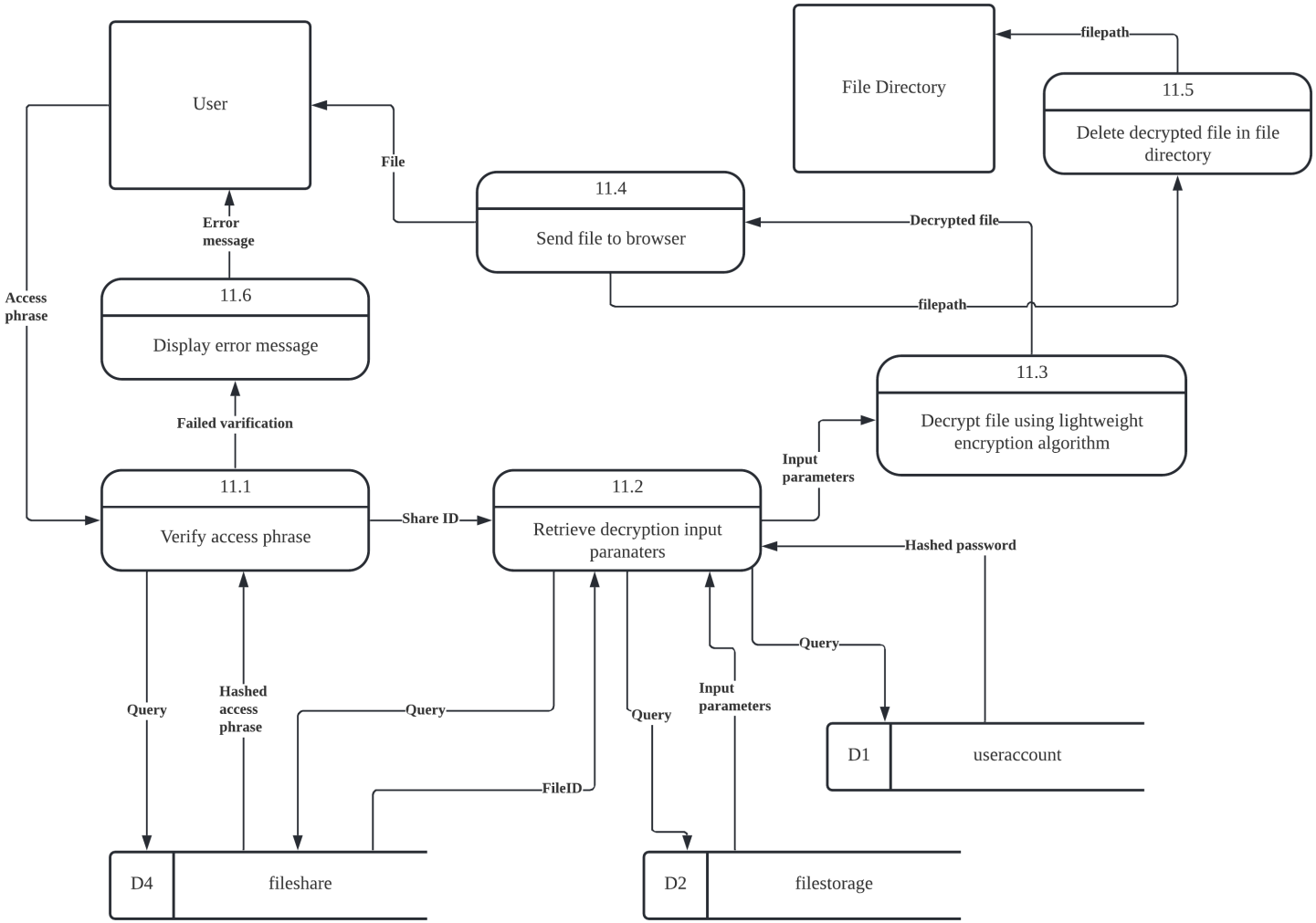


Figure 4.22: Data Flow Diagram Level 1 11.0

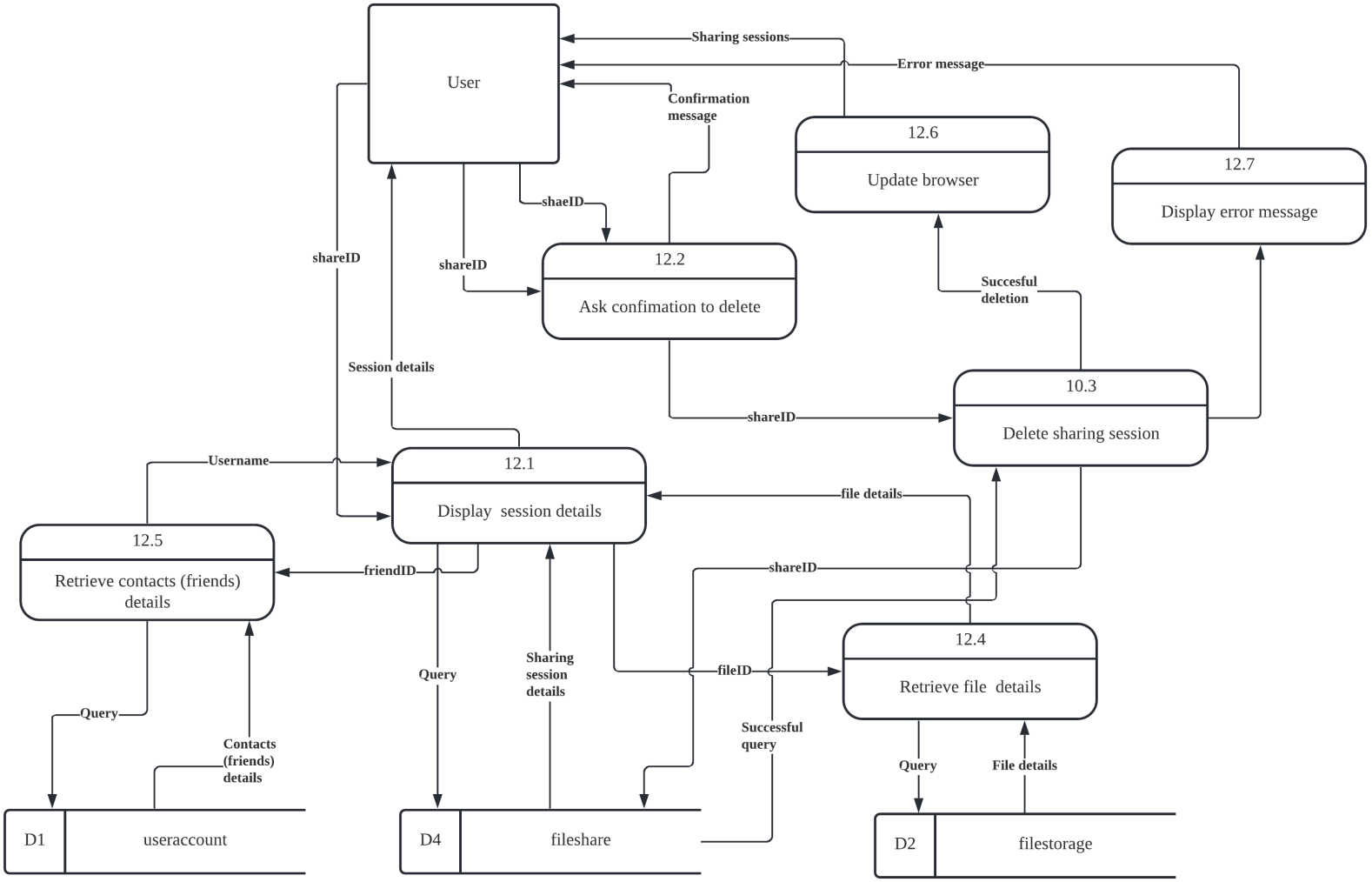


Figure 4.23: Data Flow Diagram Level 1 12.0

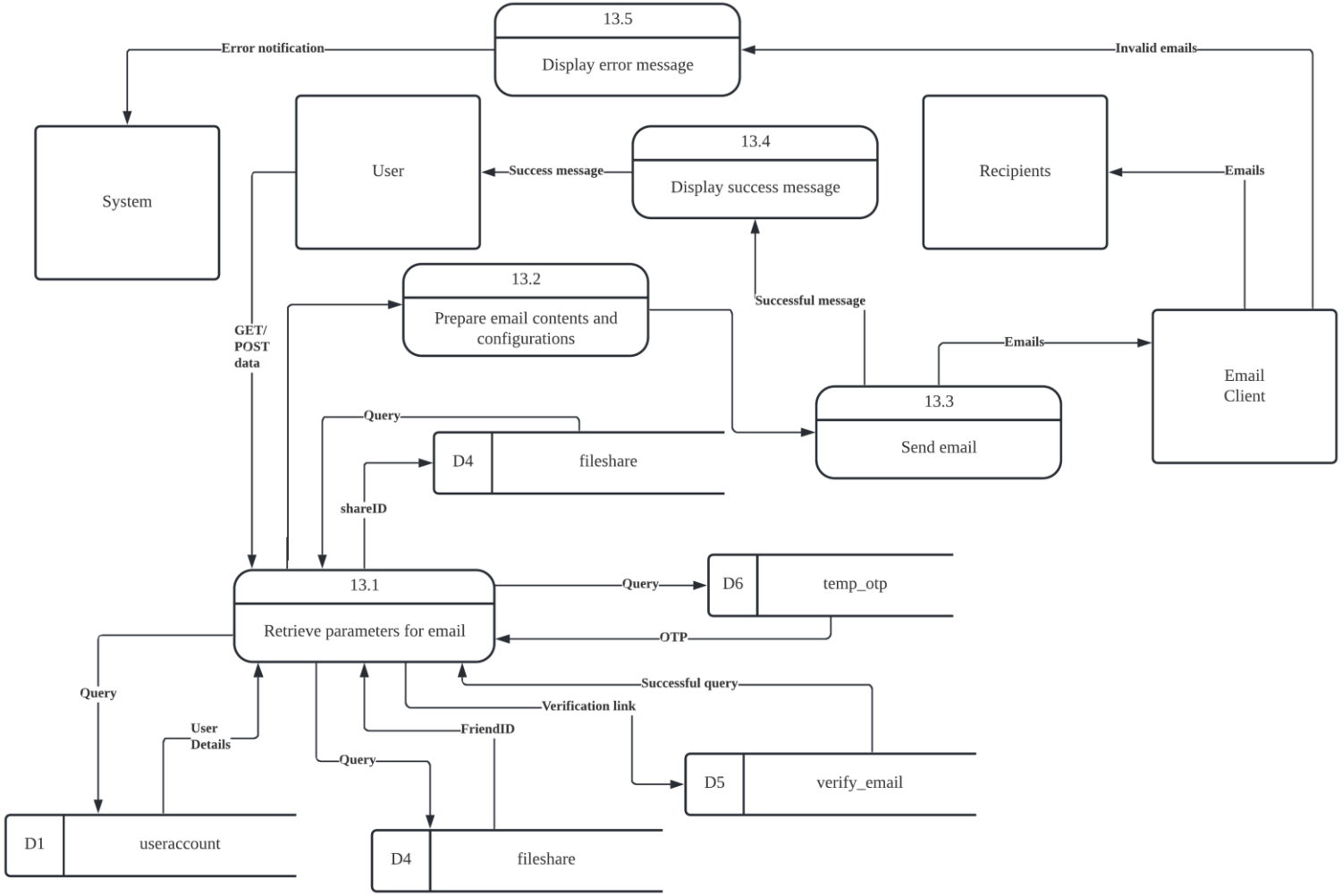


Figure 4.24: Data Flow Diagram Level 1 13.0

4.4.3 Data Dictionary

Table 4.1: Data Dictionary for useraccount table

Column	Type	Null	Default	Extra	Remarks
userID	varchar (12)	No			
userName	varchar (100)	No			
userPwd	varchar (512)	No			
secretPhrase	varchar (50)	No			
userEmail	varchar (50)	No	NULL		
email_verified	varchar (2)	No			Users verify email or not
userBio	varchar (255)	No			
userDP	varchar (150)	No			DP = Display Picture

Table 4.2: Data Dictionary for filestorage table

Column	Type	Null	Default	Extra	Remarks
userID	varchar (12)	No			->useraccount.user ID ON UPDATE NO_ACTION ON DELETE CASCADE
fileID	varchar (12)	No			
filepath	varchar (255)	No			
file_extension	datetime	No	current_timestamp()		
hashed_phrase	datetime	No			Acts as key that user to submit before decrypt file with master key
auto_generated	varchar (255)	No			1=auto-generate, 0=not auto-generated
file_IV	datetime	No			
salt	varchar (32)	No			
dateSaved	datetime	No	current_timestamp()		
lastDay	datetime	No			

Table 4.3: Data Dictionary for fileshare table

Column	Type	Null	Default	Extra	Remarks
shareID	varchar (12)	No			
fileID	varchar (12)	No			->filestorage.fileID ON UPDATE RESTRICT ON DELETE CASCADE
friendID	varchar (12)	No			
dateShared	datetime	No	current_timestamp()		
lastDay	datetime	No			
accessLink	varchar (255)	No			acc --> hashed shareID
accessPhrase	datetime	No			

Table 4.4: Data Dictionary for reset_pwd table

Column	Type	Null	Default	Extra	Remarks
hash_reset	varchar (255)	No			
userid	varchar (12)	No			
time_stamp	datetime	No	current_timestamp()		

Table 4.5: Data Dictionary for temp_otp table

Column	Type	Null	Default	Extra	Remarks
userID	varchar (12)	No			
otp_temp	varchar (255)	No			
time_stamp	datetime	No	current_timestamp()		

Table 4.6: Data Dictionary for userfriends table

Column	Type	Null	Default	Extra	Remarks
count	int (4)	No		auto_increment	
userid	varchar (12)	No			->useraccount.userID ON UPDATE NO_ACTION ON DELETE CASCADE
friendid	varchar (12)	No			
friend_req	int (4)	Yes	NULL		1=friend req sent, 2=friend req accepted, NULL=not friend req, 3=blocked
connected	int (4)	Yes	NULL		1=connected, NULL=not connected, 2=blocked

Table 4.7: Data Dictionary for verify_email table

Column	Type	Null	Default	Extra	Remarks
userID	varchar (12)	No			->useraccount.userID ON UPDATE CASCADE ON DELETE CASCADE
email_hash	varchar (255)	No			
hash_type	int (4)	No			1=email verification, 2=new email verification
Tmp_time	datetime	No	current_timestamp()		

4.4.4 User Interface Design

This section will showcase and explain the user interface design of the system. All the features or tasks can be categorized into two types: Guest, and User. Table 4.1 lists down all the included tasks or features within the system including the user types that has access towards it.

Table 4.8: List of features/tasks of each user type

Tasks/Features	Visitors	Users
Registration	/	/
Login	/	/
Manage profile	X	/
Change password	X	/
Change email	X	/
Deactivate account	X	/
Set secret phrase	X	/
Encryption	/	/

Decryption	/	/
Store files	X	/
Deleting files	X	/
Sharing files	X	/
Adding contacts	X	/

The system design is aimed to be very simple and easy to understand. The web application is named as 'Horusdocs' and has a logo with the picture of a falcon biting a key. The name is inspired by the Greek mythology God named 'Horpocrates', which is known as the god of silence, secrets and confidentiality. The name 'Horus' and the logo are taken from the Egyptian version of Horpocrates, which is 'Horus', that represents well-being, healing, and protection. This god is portrayed with a human-like body with a falcon head. It fits the theme of the web application, thus the word 'Horus' is used with 'docs', where it gives a meaning of protecting documents. Figure 3.2 shows the logo of this web application. The logo was made using Canva.

Figure 4.25: Logo of the web application

The landing page of the website shows a large banner, with a quick introduction on what the website offers. The top left shows the logo, which is

clickable and redirects user to the landing page. The top right shows two buttons, 'Sign Up' and 'Login', intended for the User Registration and Authentication module.

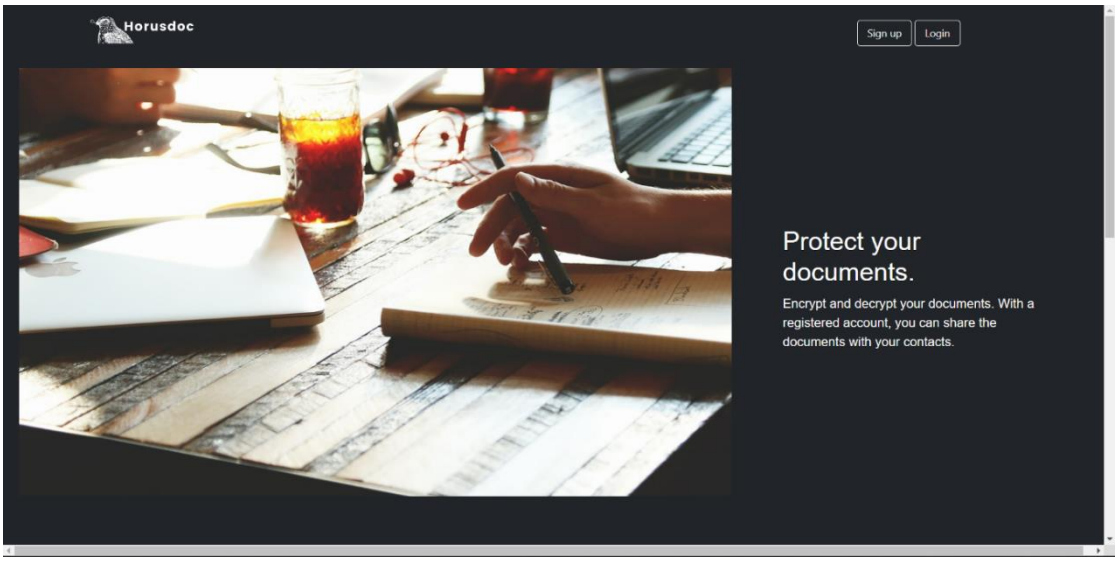


Figure 4.26: Landing page of Horusdoc

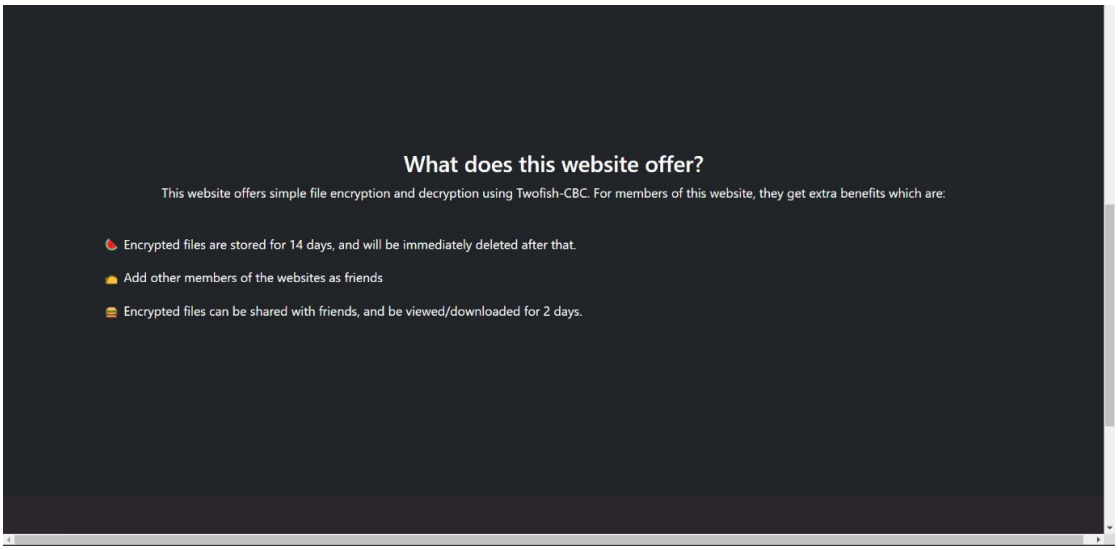


Figure 4.27: Quick introduction of web application

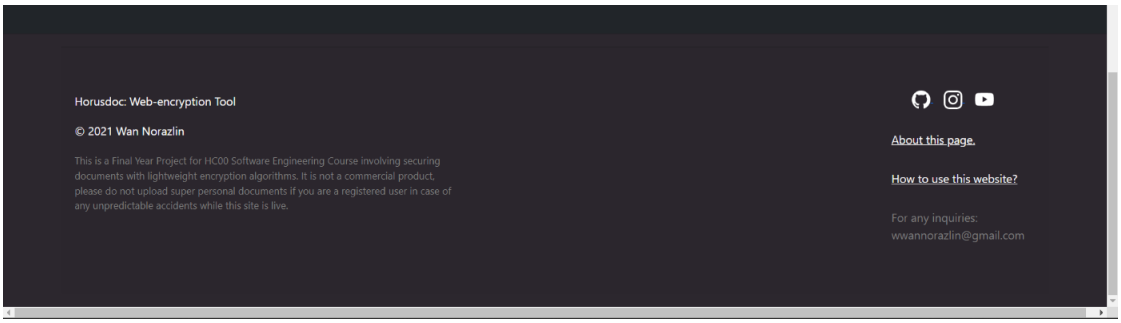


Figure 4.28: Website footer

As seen in Figure 4.35, the footer includes the name of the web application, a copyright label, a brief disclaimer about the website, social media handles, email and two links. The first link directs users to the 'About Page', where it explains users the background of the web application including the data usage as this website stores sensitive user information and documents.

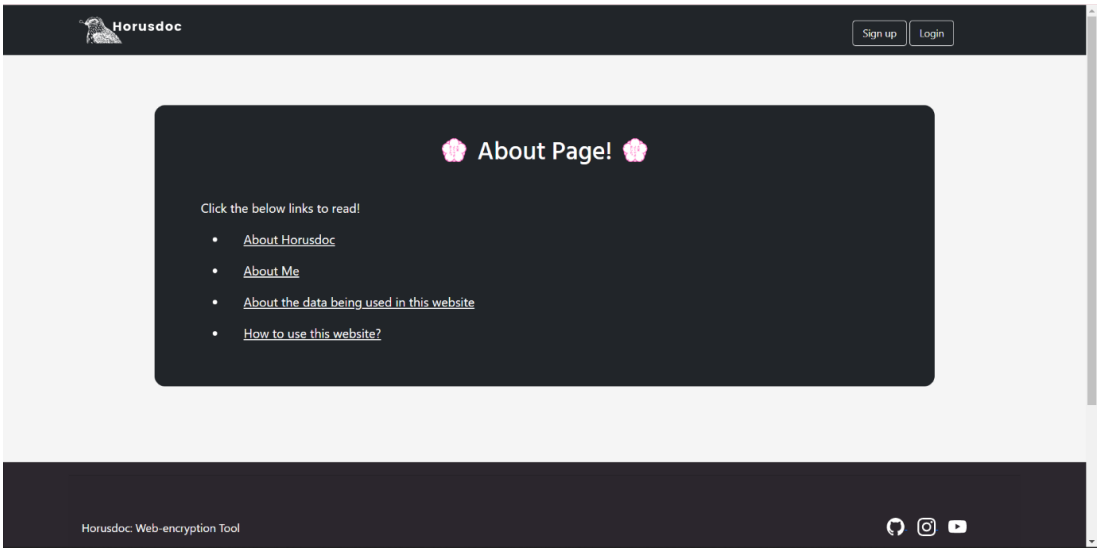


Figure 4.29: 'About Page' page

The 'Sign Up' page shows a form that users need to submit to make a valid account on the website. If users do not fill-in the form, a red outline will appear on the input elements. If users try to submit with empty inputs, the HTML customization message will show and prevent form from being submitted. If users are already registered, an alert will show above the form, and users will get an error message if

they submit the form with the registered email address. Once user submits the form with valid input, a confirmation page will appear.

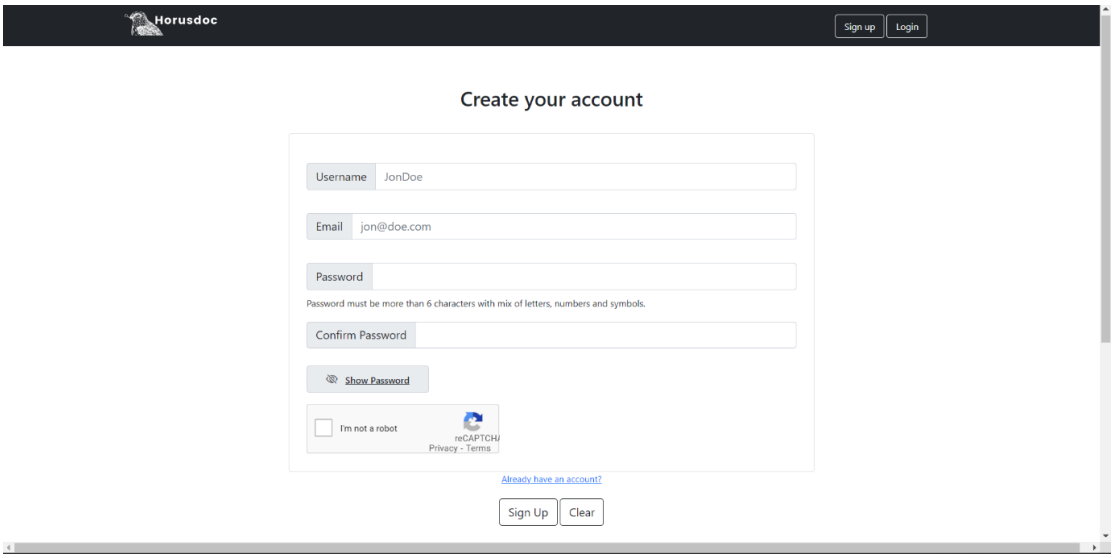


Figure 4.30: Registration page

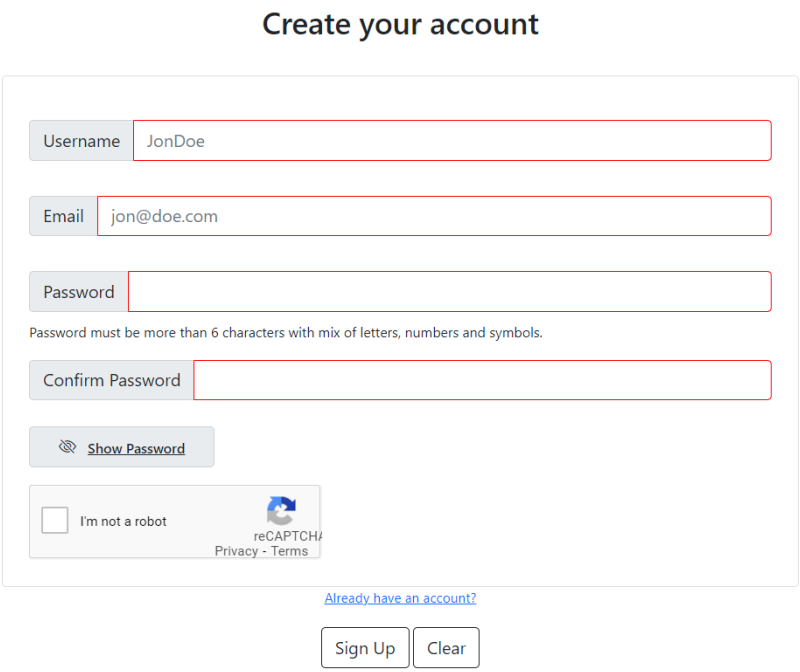


Figure 4.31: Input validation if input elements are empty

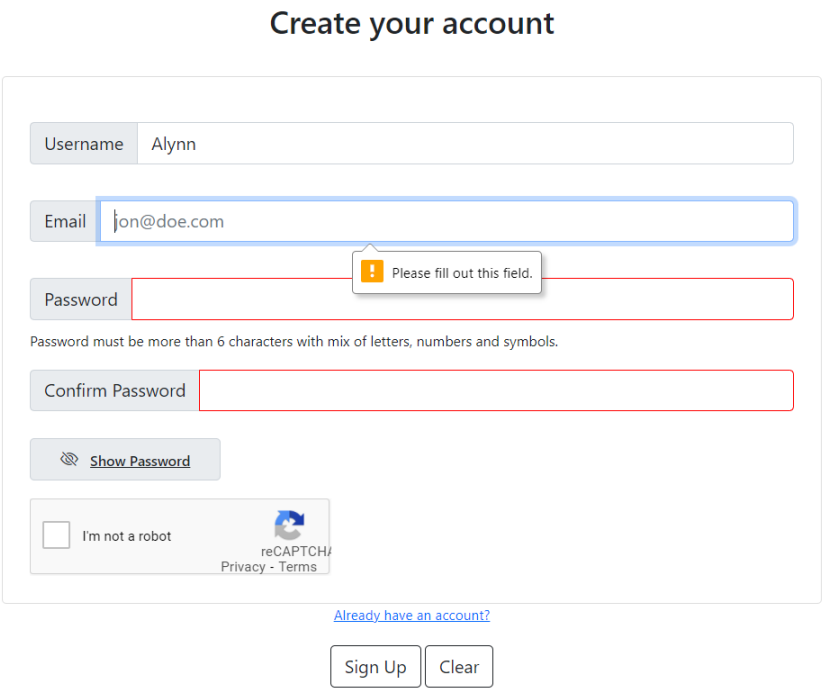
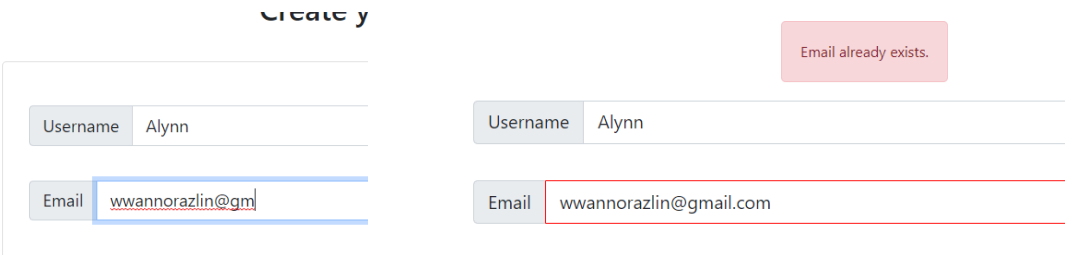


Figure 4.32: Input validation message when user tries to submit empty form



(a) (b)

Figure 4.33 (a) and (b): Input validation message if email is registered

Figure 4.34: Error message from server-side

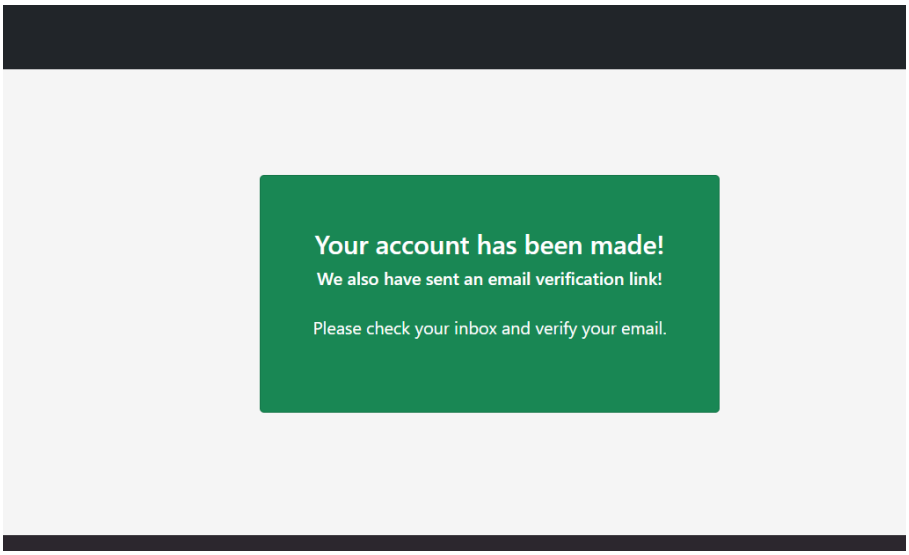


Figure 4.35: Confirmation message of user registration

Login page shows a form with three input elements, which requires email address, password and reCAPTCHA v2. Similar to the registration page, the input elements will have a red outline if it is empty, and the HTML input validation message will show up if user tries to submit an empty form.

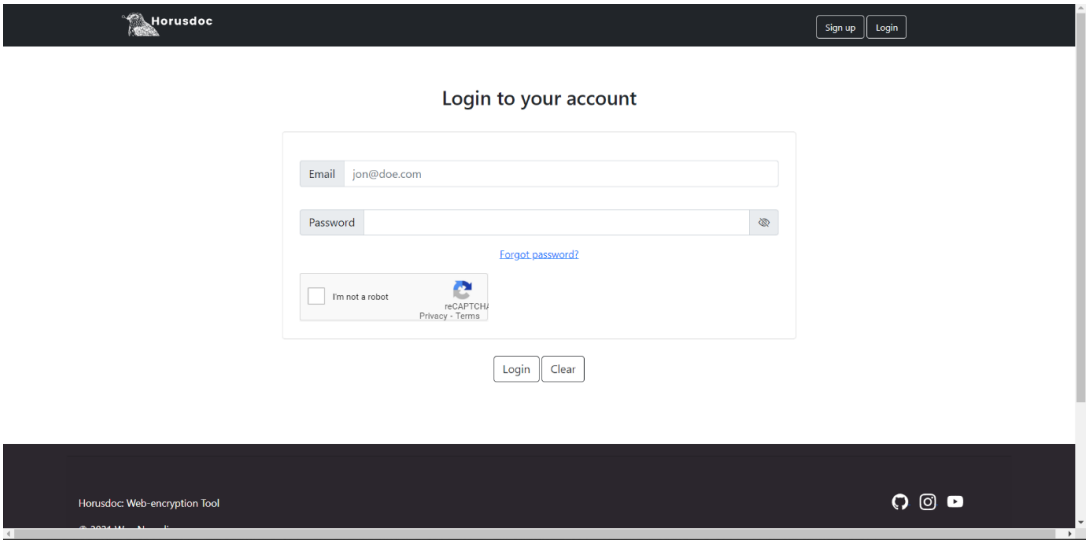


Figure 4.36: Login page

Once logged in, users will see the homepage of the online account as seen in Figure 4.44. The top navigation consists of the same buttons and links as in landing page but with an additional search bar and a logout button on the top right. The search bar is intended for users to search for other users to add as contacts (friends). This falls under the Contact List and Policy Management module, which allows users to only share file with users that they add as contacts or friends. Users need to enter either the name, email address or user identification number (userID) into the search bar. Figure 4.45 shows the search result of a user named 'DeerDear'. Once user sends a 'friend request', the button will change to 'Friend Request Sent'.

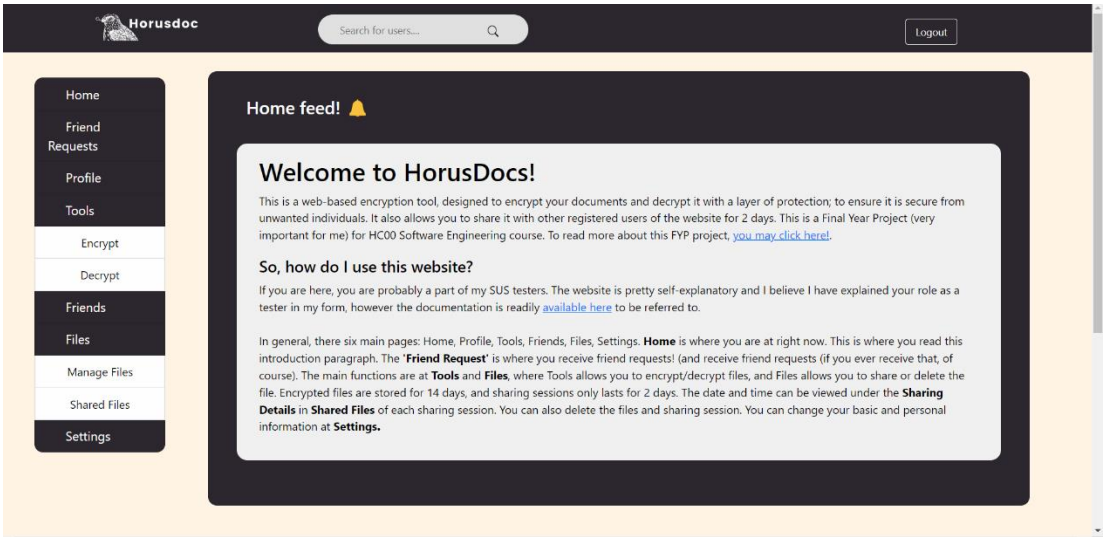


Figure 4.37: Homepage of user account

Figure 4.38: Search results of a user

Figure 4.39: Button change after user send friend request

The side navigation shows seven other functions or available activities, as seen in Figure 4.44. 'Tools' and 'Files' have two different pages dedicated for its functions. It is separated so that the website does not become clustered and complex. 'Home' shows the homepage of the user account. 'Friend Requests' shows any friend requests available for the currently logged in user, which can be seen in Figure 4.47. Similar to the button on 'Search Results' page, it changes after user clicks it to accept the friend request.

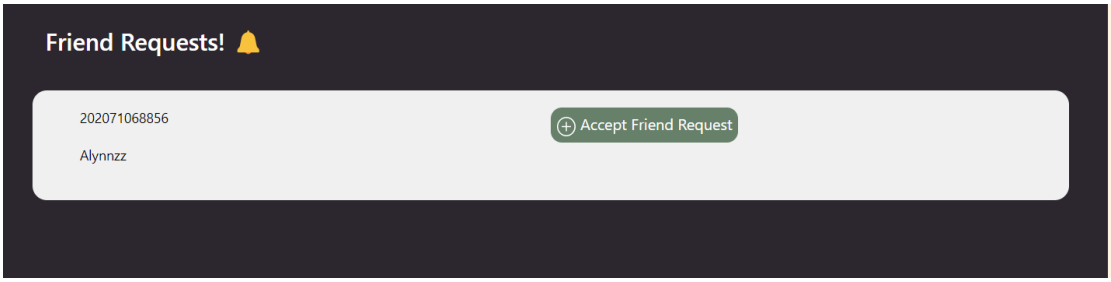


Figure 4.40: Friend request page

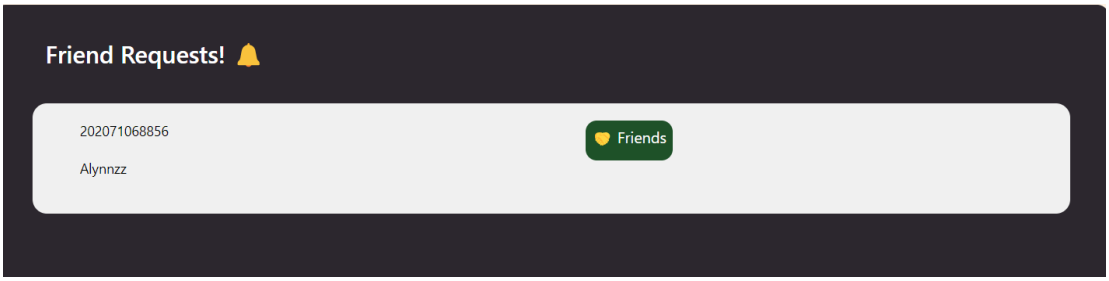


Figure 4.41: Accepted friend request

'Profile' shows the basic user information which are username, user ID, profile picture and biography, which can be seen in Figure 4.49. On this page, user can change their profile picture and biography. This page and 'Settings' page falls under the user account management module.

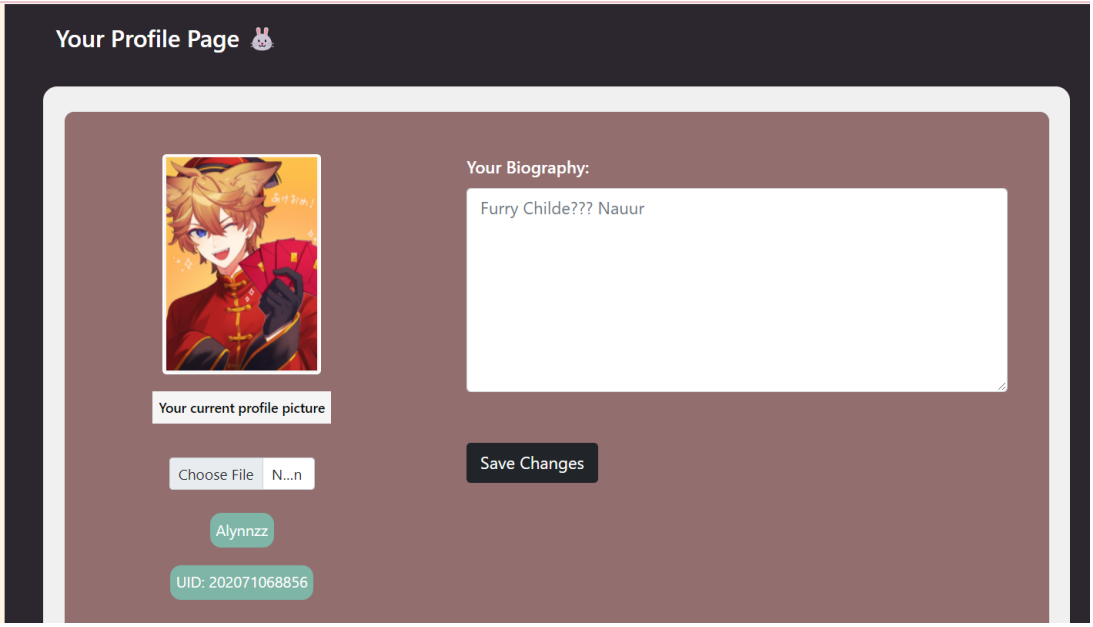


Figure 4.42: Profile page of user account

As mentioned above, 'Tools' has two navigation links under it, which are 'Encrypt' and 'Decrypt'. 'Encrypt' is intended for file encryption, and 'Decrypt' is for file decryption. This falls under the Web-based encryption tool module. Figure 4.50 and 4.51 shows the encryption and decryption pages, respectively. File encryption page contains a form that accepts a file input and a password-type input. Users can choose to not provide a passphrase and auto-generate it by checking the 'Please auto-generate key for me' checkbox. Encrypted files can be accessed for decryption on the 'Decrypt' page, and file sharing in the 'Manage Files' page.

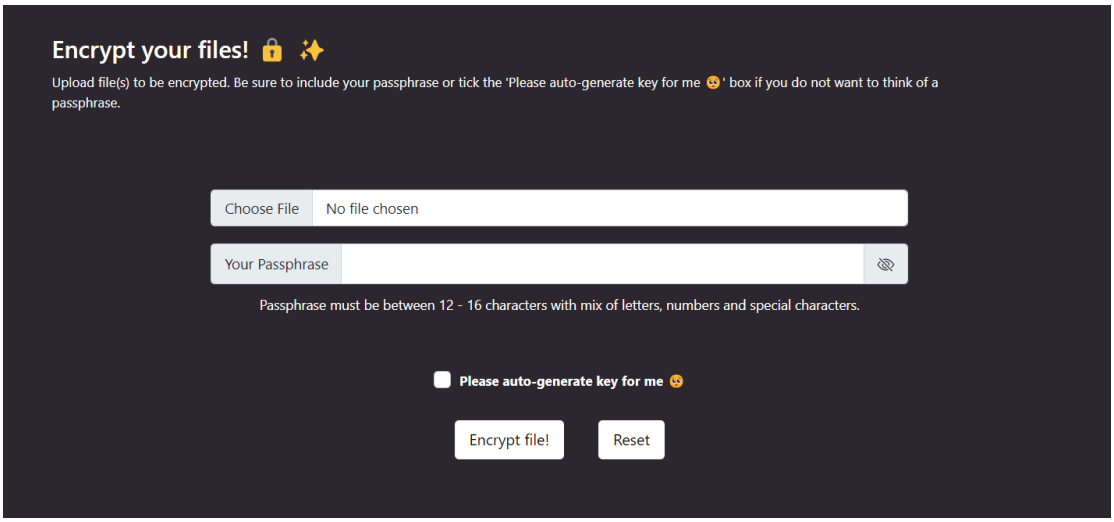


Figure 4.43: File encryption page

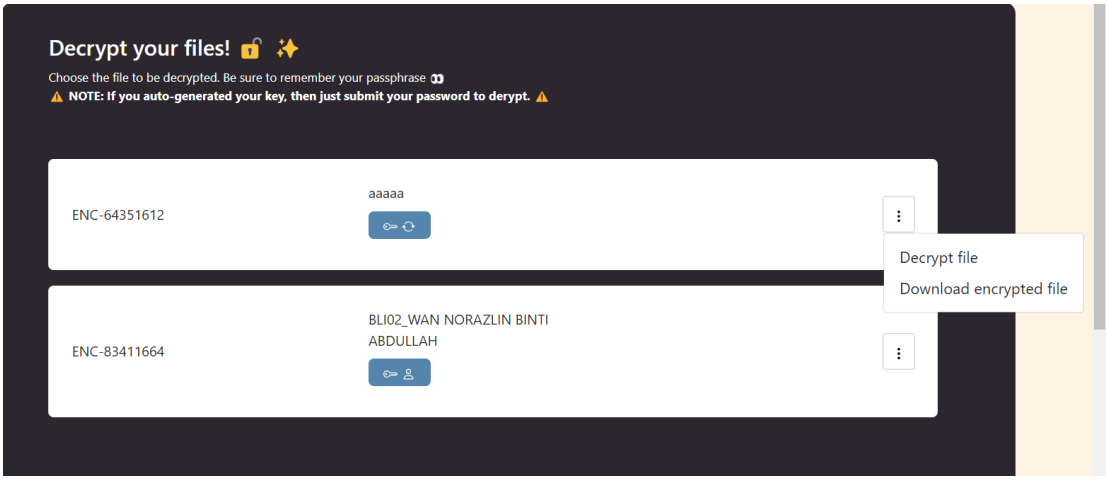


Figure 4.44: File decryption page

Decryption page displays all the stored encrypted files, with two functions available which are 'Decrypt file' and 'Download encrypted file', as shown in Figure 4.51. Upon clicking the 'Decrypt file' button, a modal will appear and prompts user for the passphrase. If user submits the correct passphrase/key, a download prompt will appear in the browser where user can choose to open or download the file. If an invalid key is submitted, an error alert will appear on top of the top navigation, stating the error that caused the decryption failure. For downloading encrypted file, user needs to submit their password. A valid password submission will prompt a download, while an invalid one will prompt an error alert similar to the decryption module.

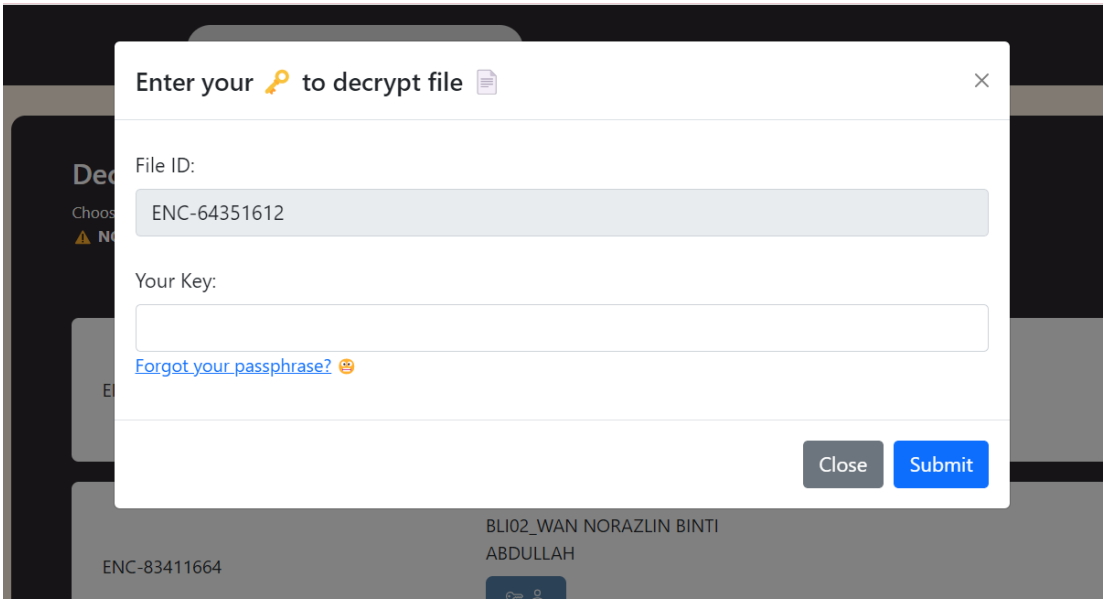


Figure 4.45: Modal prompting user to enter key or passphrase for decryption

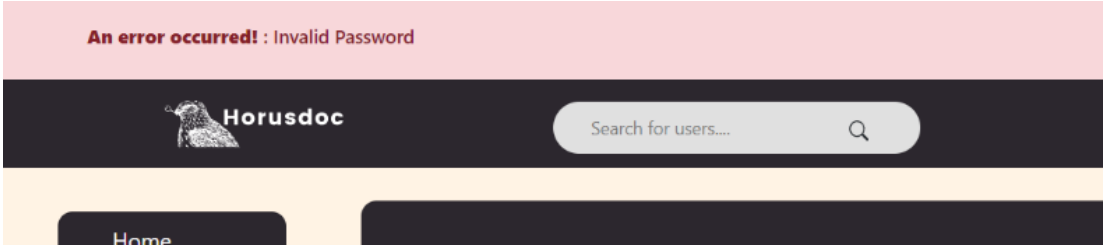


Figure 4.46: Error alert after invalid key submitted

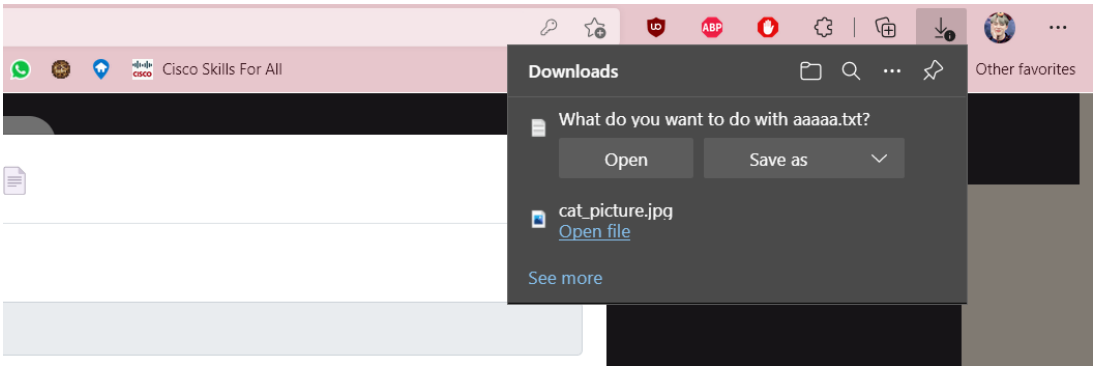


Figure 4.47: Download prompt after successful decryption

'Friends' page displays all the users that the currently logged in user is connected to, or friends with. It is just a simple display of the friend's name, userID and their profile picture, as seen in Figure 4.55.

Figure 4.48: Display of user's friends

'Files' has two navigation links under it, which are 'Manage Files' and 'Shared Files'. 'Manage Files' allows users to share the file and delete the file. This falls under the File Sharing Management. Once users click the 'Share File' button, a modal will appear to prompt user to choose contacts (friends) to share the file with. Users choose by clicking the circle tick box on the top left of the contact's details display. The tick box will have a flower in it if it is ticked, which can be seen in Figure 4.58. The 'Confirm' button will only be available only if the user has ticked at least one contact for sharing file.

After choosing the contacts and clicking the 'Confirm' button, user will be asked to provide an access password/phrase. This phrase is used as one of the verification methods to ensure that only the specific users can access the file. User is required to inform the selected contact themselves about the access phrase. Once submitted, a sharing session will be made, and user can view it under the 'Shared Files' page. The selected users will receive an email with the sharing session link and is/are informed on who is the one sharing the file.

File deletion can be done by clicking the 'Delete File' button, where users will be greeted with a modal stating that deleting it will permanently make it unavailable as seen in Figure 4.61. This modal does not prompt any input from user; thus, user

just need to click on the button to proceed. The file will be deleted from the filesystem.

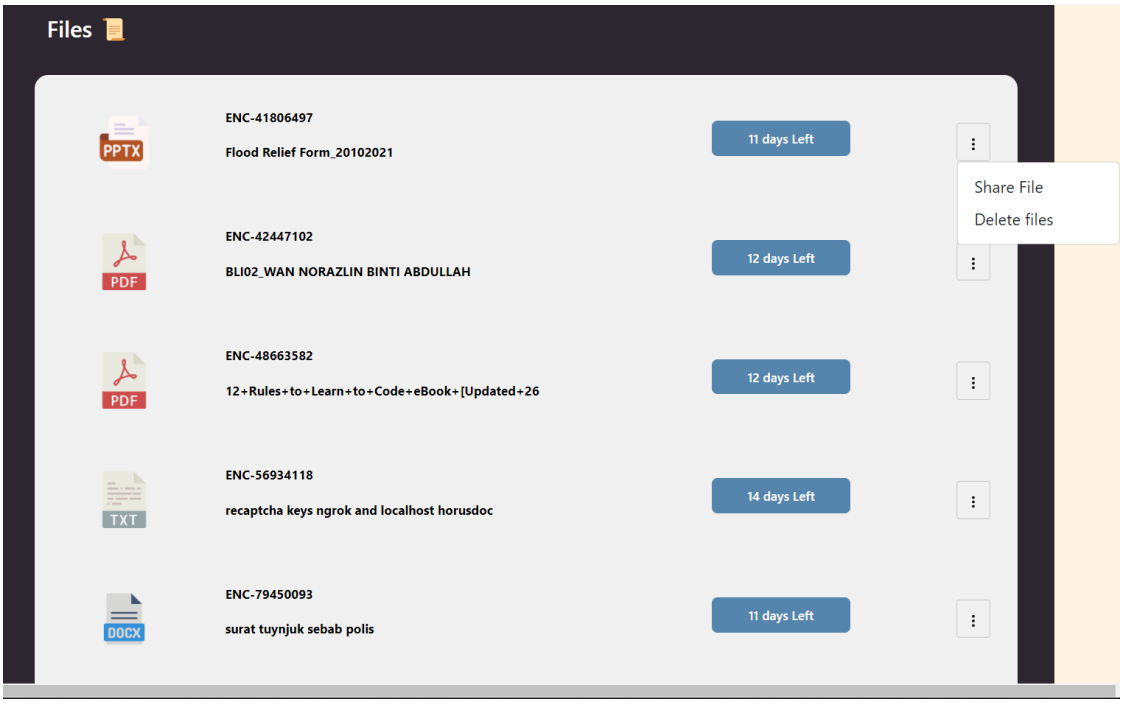


Figure 4.49: Page to manage stored encrypted files

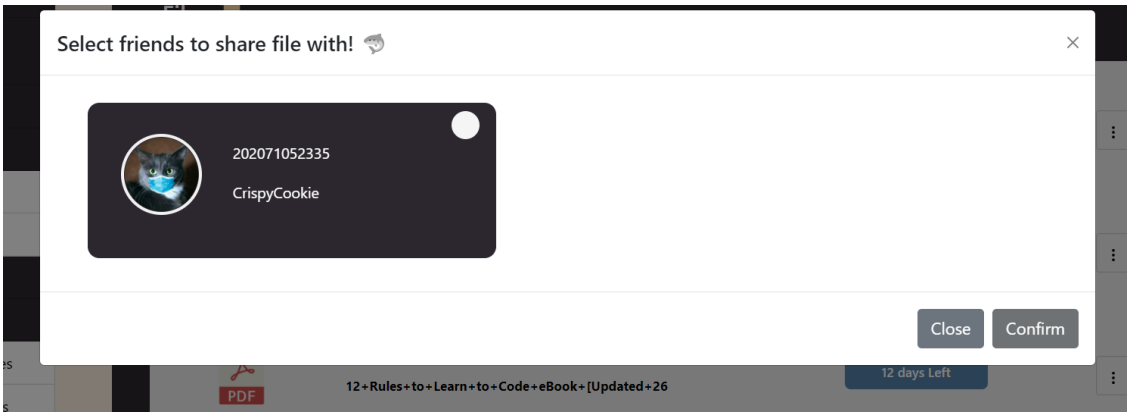
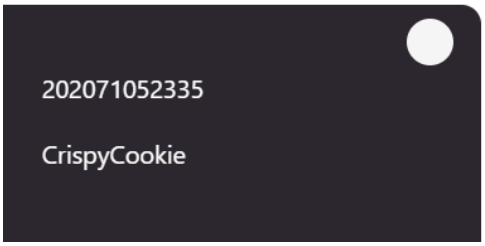
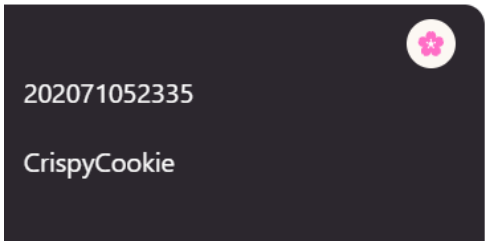


Figure 4.50: Prompt for user to choose contacts (friends) to share file with



(a)



(b)

Figure 4.51 (a) and (b): Tick box appearance once users tick it

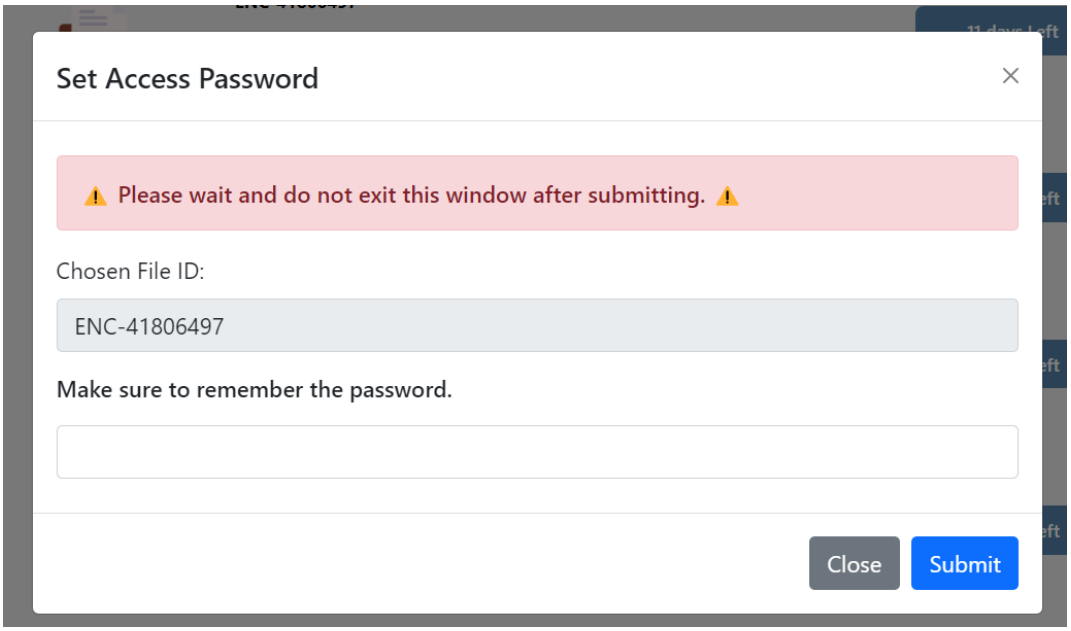


Figure 4.52: Prompt to submit access password/phrase

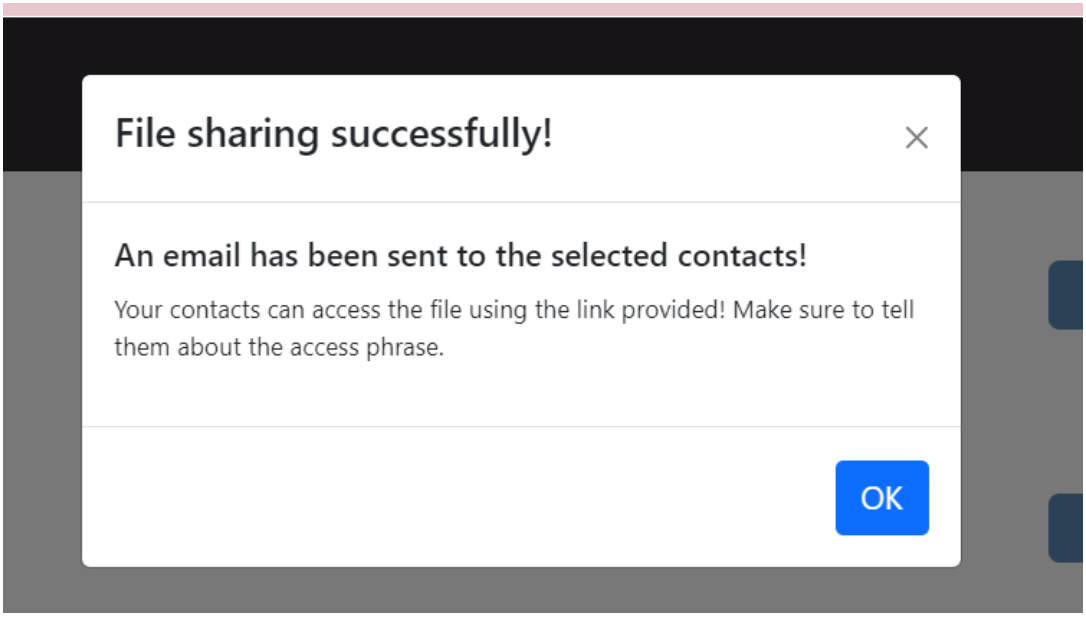


Figure 4.53: Confirmation message of email notification sent to selected contacts

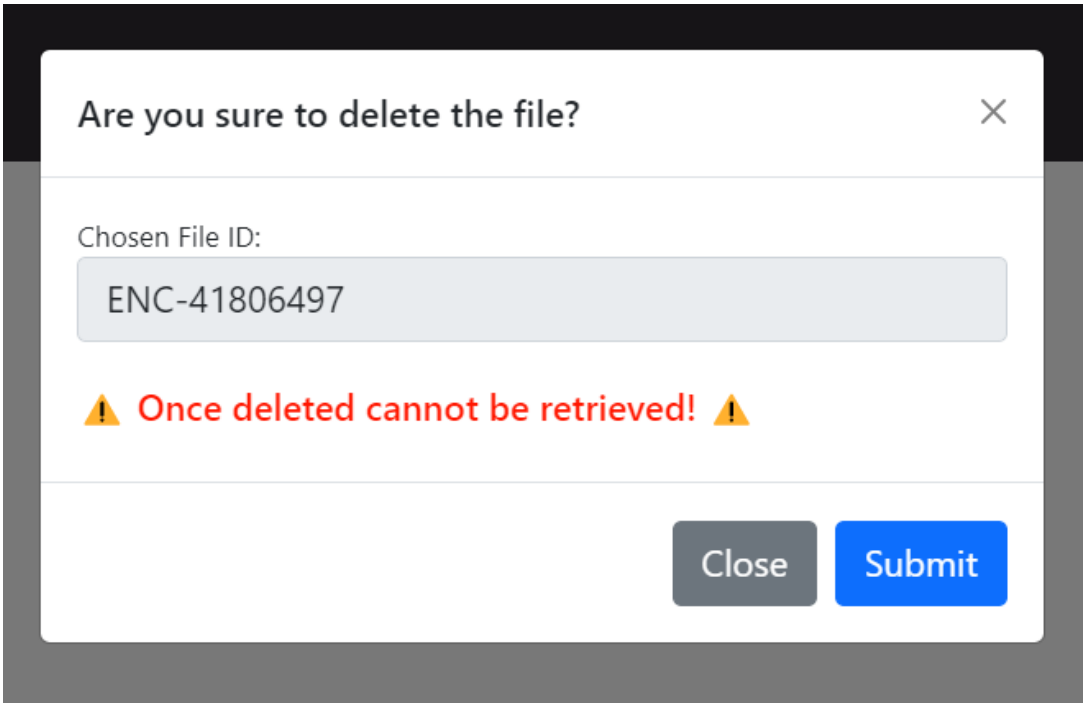


Figure 4.54: Delete file modal

Next, 'Shared Files' displays all the files that are currently shared to other users. This falls under the File Sharing Management. There are three activities that user can do: viewing file, deleting sharing session and viewing sharing session details. For viewing file, a new tab will be opened once users click the 'View File' button where a modal prompting users the access phrase/password will show up, which can be seen in Figure 4.63. If user submits an invalid access phrase/password, an error message will appear stating that it is wrong. Otherwise, user is allowed to proceed viewing or downloading the file. As of now, only PDF files can be viewed directly on browser, meanwhile other files are needed to be downloaded first prior viewing.

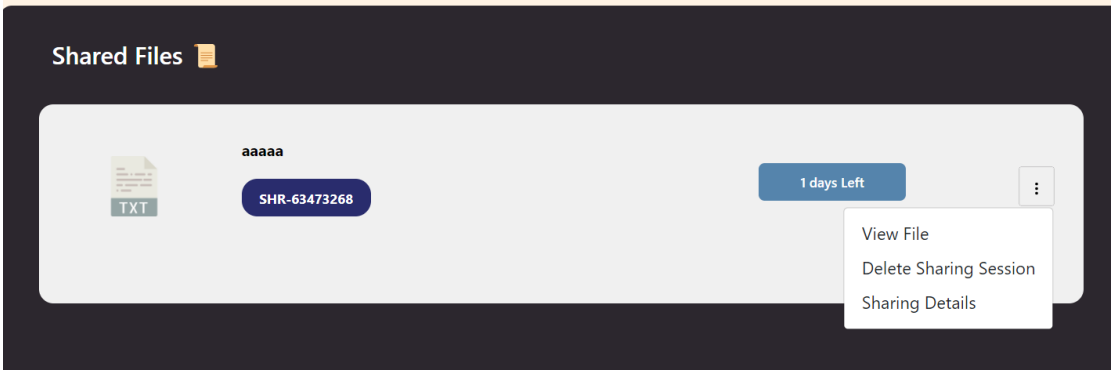


Figure 4.55: Shared Files Page

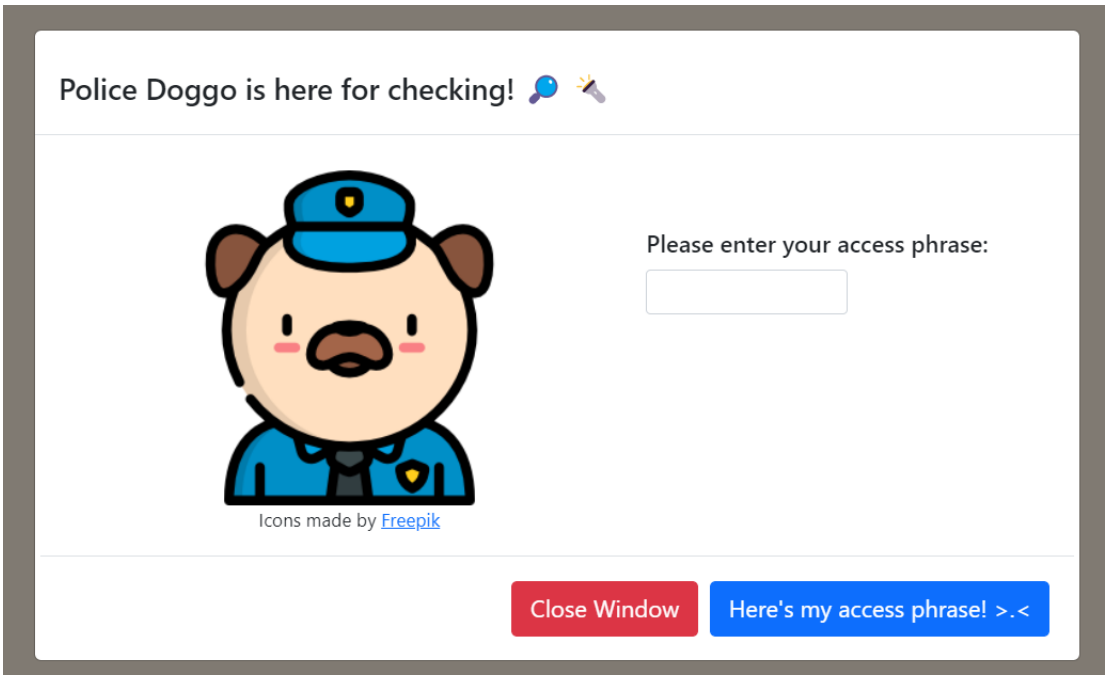


Figure 4.56: Modal that prompts user the access phrase/password

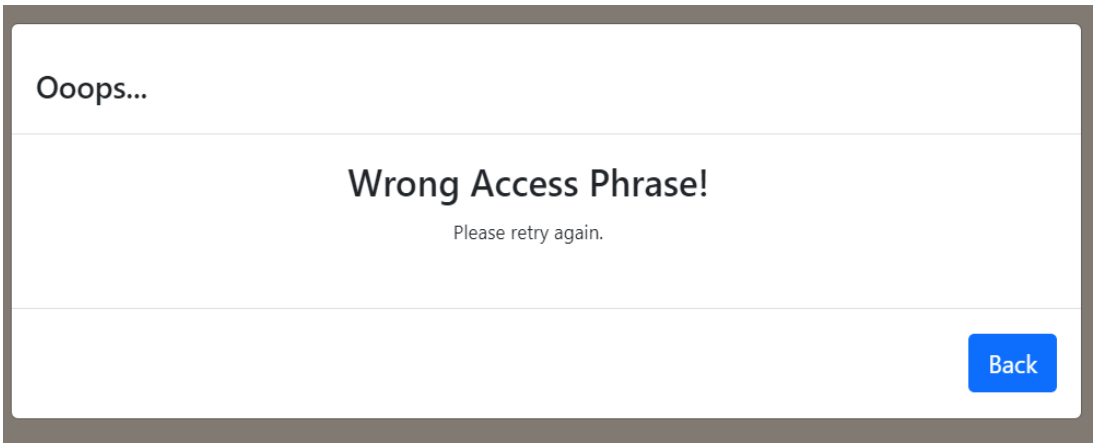


Figure 4.57: Error message for invalid access phrase

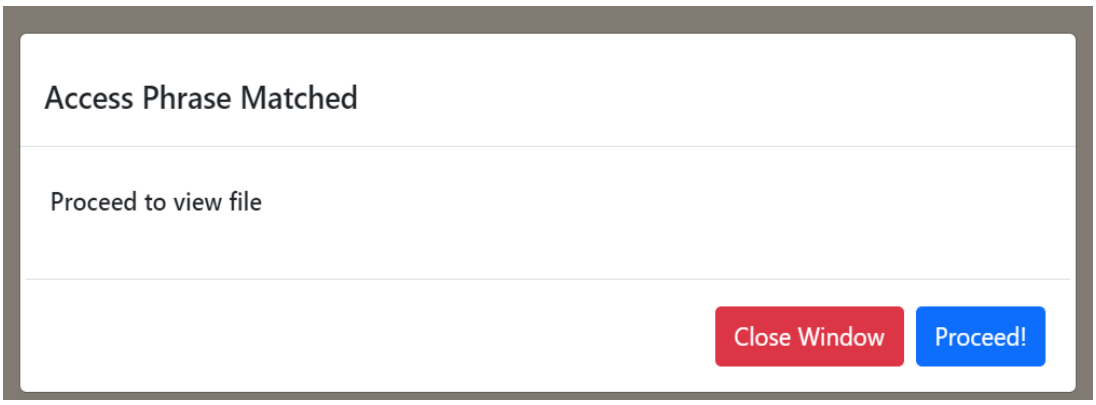


Figure 4.58: Modal for users to view/download the file

Lastly, the 'Settings' page is where users can change their user details such as username, email and password. This falls under the User Account Management module. There are several user details updates that requires an OTP (One-Time-Password) via email, and others only need to provide password. An example of details updates that user need to do OTP can be seen in Figure 4.69. For email address, an alert will be shown whether user has verified their email address or not.

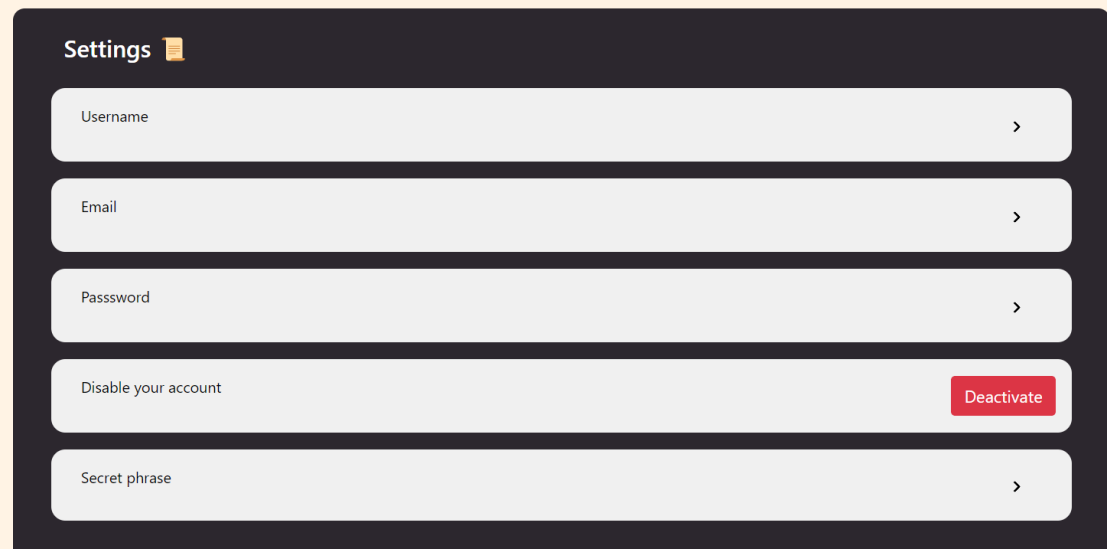


Figure 4.59: Settings page

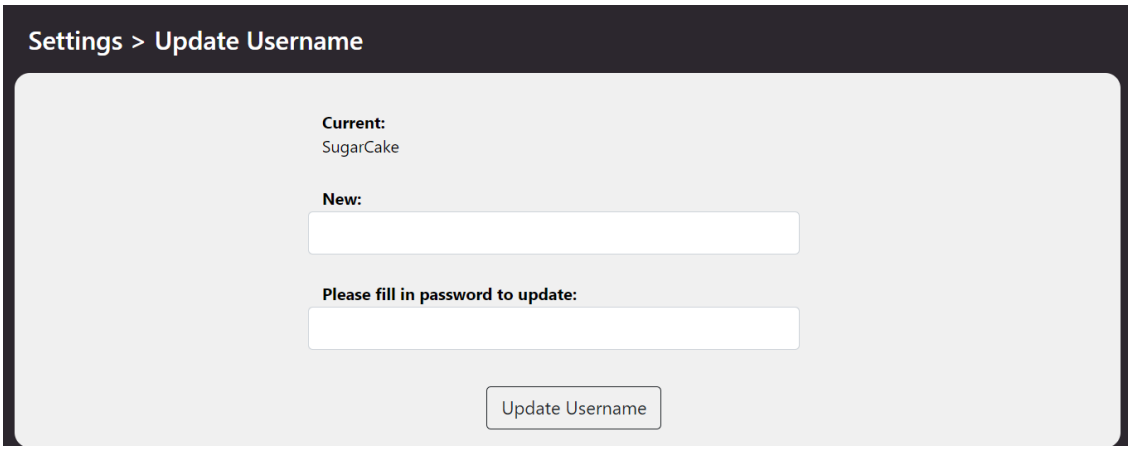


Figure 4.60: Page to change username

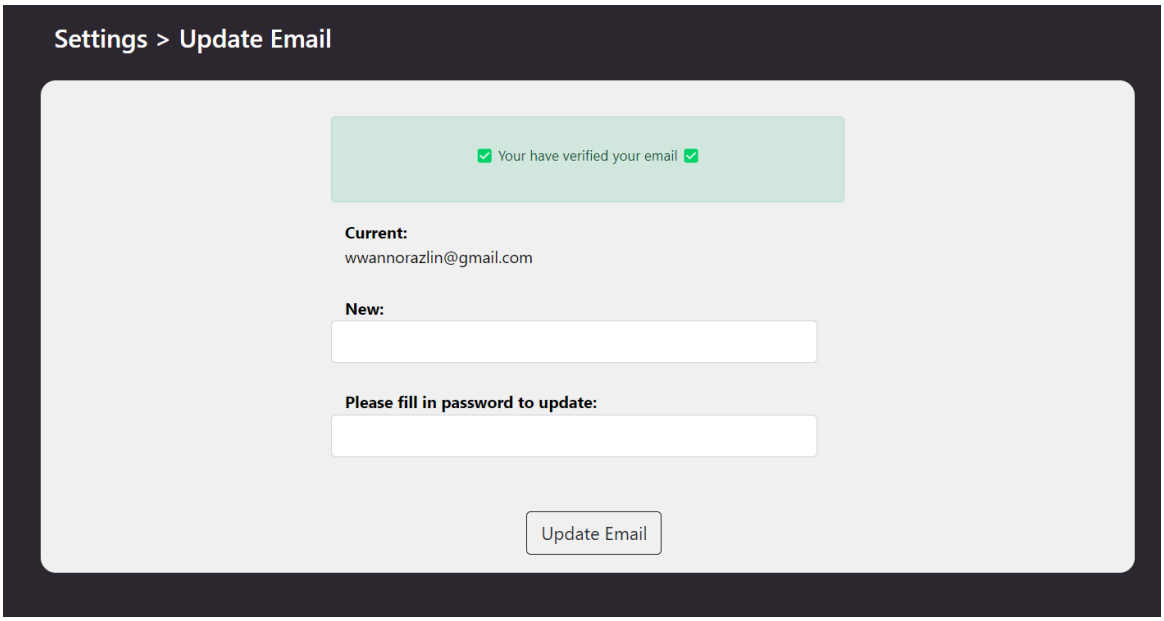


Figure 4.61: Update email address page

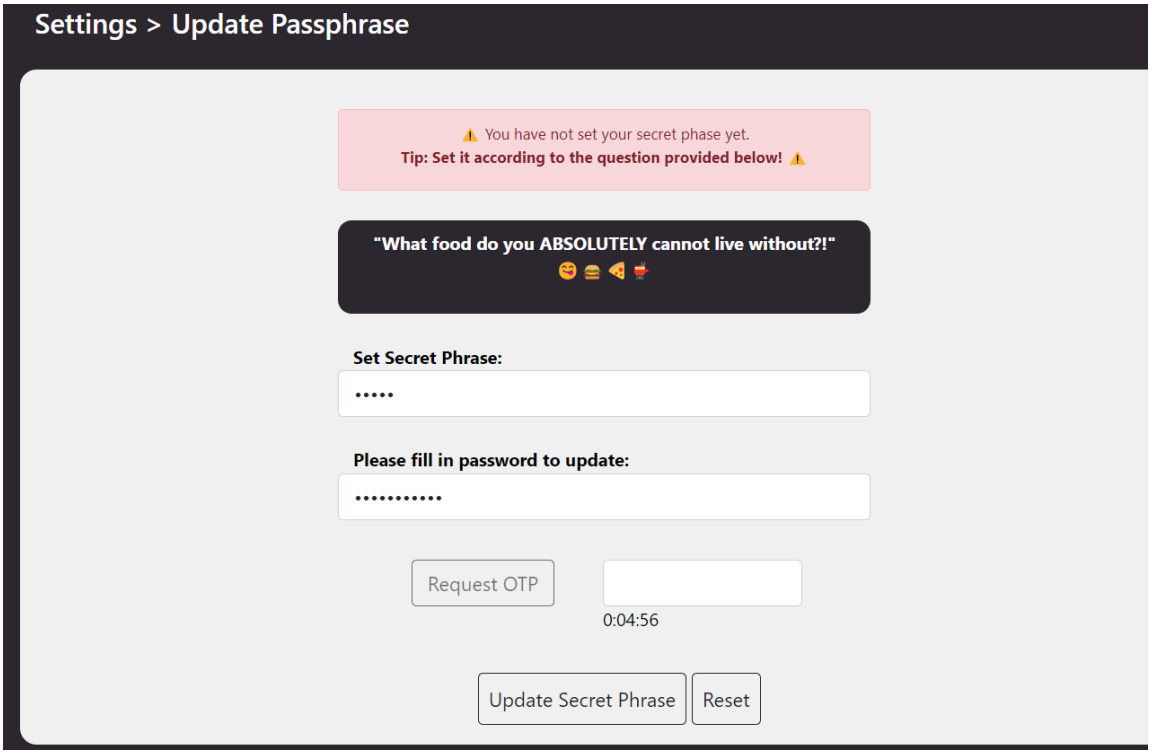


Figure 4.62: Update secret phrase page

4.5 Conclusion

This chapter involved analysing the user requirements by discussing the data collections and questionnaire findings. Besides that, the design is sketched and displayed using artefacts such as DFD and ERD to show the relationships of entities and the flow of data throughout the system. In addition, UI design is also included to give a clearer picture of the system and how it functions as a web application.

CHAPTER 5

EXPERIMENT IMPLEMENTATION

5.1 Introduction

This chapter involves the discussion on experiment implementation which includes the dataset and parameter settings, including the algorithm for experimentation.

5.2 Dataset and Parameter Settings

This section discusses the dataset being used for the experimentation, including an explanation regarding the parameter settings, to compare and select the best lightweight algorithm according to its computation speed and encryption file size.

The dataset comprises of nine types of document files, which are XML, ODS, CSV, DOCX, PPT, PPTX, XLSX and TXT. These documents are chosen based on the questionnaire results discussed in Section 4.2, where it showed that the respondents mostly encrypt document files compared to other file types such as images and videos. The sizes of these files vary but each size can be categorized into three types of sizes: small, medium, and big. The sample files are retrieved from various resources, which can be seen on Table 5.1.

Table 5.1: The datasets involved in the experiment

Small sizes			Medium sizes			Large sizes		
File type	**Size (KB)**	**Source**	**File type**	**Size (KB)**	**Source**	**File type**	**Size (KB)**	**Source**
XML	4.548	Kaggle	XLSX	130.05	Kaggle	DOCX	1285.36	Course files
TXT	17.9	Google	CSV	316.971	Kaggle	PPT	3985.92	Course files
ODS	27.655	Google	PPTX	898.927	Course files	PDF	4556.29	Course files

The language being used for this experimentation is in C++, and the integrated development environment (IDE) being used is Visual Studio 2019. The machine being used for this experiment is described below.

Table 5.2: Specifications of the machine being used for experimentation

Item	Value
System Model	HP Laptop 14s-cf1xxx
System Type	X64-based PC
CPU	Intel (R) Core (TM) i5-8265U CPU @ 1.60GHz, 1800 MHz, 4 Core(s), 8 Logical Processor(s)
RAM	12 GB
Operating System	Microsoft Windows 10 Home Single Language

The encryption algorithms that were tested and compared are Twofish, LEA and XTEA. Each key lengths that these encryption algorithms provide were tested with all the nine datasets. As seen in Table 5.3, XTEA only provides 128-bit key length, compared to Twofish and LEA with all three key lengths. Each encryption and decryption of dataset using the respective encryption algorithms with each of its key lengths, were being done in 100 trials to ensure that the time taken for each

encryption and decryption process can clearly be seen and compared in seconds. Block cipher mode used is CBC with PKCS#7 padding scheme. All the algorithms and schemes were adapted from the open sourced Crypto++ library.

Table 5.3: Encryption algorithms being tested

Encryption algorithms	**Key Lengths**		
	128-bit	**192-bit**	**256-bit**
Twofish	/	/	/
LEA	/	/	/
XTEA	/	X	X

The computation speed is measured by using the functions provided in the Chrono library, that handles time and clocks according to the system of the machine. The wall time and CPU (process) time is measured for every encryption and decryption process of each file and key lengths. The results are then output into a text file for easy viewing.

5.3 Algorithm Implementation

As mentioned above, the algorithm is being implemented with CBC mode alongside PKSC#7 padding scheme using the Crypto++ library. The library is designed so that developers will need to use any cipher mode to use a cipher for any encryption and decryption process. Thus, each algorithm is implemented using the same method. The padding scheme is also automatically applied into their functions; hence developers do not need to separately call or define the padding scheme prior encryption or decryption.

5.3.1 Twofish Algorithm Implementation

Before calling the Twofish class, the key length and IV needs to be declared its length. Figure 5.1 shows the initialization of key length and IV under Twofish using the SecByteBlock data type, provided by the library. Both values must be initialized properly prior any encryption to ensure the proper block padding is being used. The key length is changed according to the number of rounds. Since CBC_Mode is used, the encryption must include the key with IV.

```
SecByteBlock key((const unsigned char*)(password.data()), 32);
SecByteBlock iv((const unsigned char*)(fixed_iv.data()), Twofish::BLOCKSIZE);
```

Figure 5.1: Initialization of key length and IV for Twofish

In this experiment, the Twofish algorithm is implemented using CBC mode that is provided by the library. As seen in Figure 5.2 and Figure 5.3, the Twofish class is called using a function under the CBC_Mode class.

```
CBC_Mode< Twofish >::Encryption Encrypt;
Encrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.2: Calling Twofish class through CBC Mode function for encryption

```
CBC_Mode< Twofish >::Decryption Decrypt;
Decrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.3: Calling Twofish class through CBC Mode function for decryption

5.3.2 LEA Algorithm Implementation

The implementation of LEA algorithm is similar to Twofish. The key length and IV needs to be declared and initialized prior encryption or decryption process. Figure 5.4 shows the initialization of the key lengths and IV using the SecByteBlock data type. The key length is changed according to the number of rounds.

```
SecByteBlock key((const unsigned char*)(password.data()), 32);
SecByteBlock iv((const unsigned char*)(fixed_iv.data()), LEA::BLOCKSIZE);
```

Figure 5.4: Initialization of key length and IV for LEA

LEA algorithm is implemented using CBC mode that is provided by the library. As seen in Figure 5.5 and Figure 5.6, the LEA class is called using a function under the CBC_Mode class. Since CBC_Mode is used, the encryption must include the key with IV.

```
CBC_Mode< LEA >::Encryption Encrypt;
Encrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.5: Calling LEA class through CBC Mode function for encryption

```
CBC_Mode< LEA >::Decryption Decrypt;
Decrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.6: Calling LEA class through CBC Mode function for decryption

5.3.3 XTEA Algorithm Implementation

The key length and IV needs to be declared and initialized prior encryption or decryption process. Figure 5.7 shows the initialization of the key lengths and IV. Since XTEA only has 128-bit key length, hence only the value 16 is available to be used with XTEA.

```
SecByteBlock key((const unsigned char*)(password.data()), 16);
SecByteBlock iv((const unsigned char*)(fixed_iv.data()), XTEA::BLOCKSIZE);
```

Figure 5.7: Initialization of key length and IV for XTEA

XTEA algorithm is implemented using CBC mode that is provided by the library. As seen in Figure 5.8 and Figure 5.9, the LEA class is called using a function under the CBC_Mode class.

```
CBC_Mode< XTEA >::Encryption Encrypt;
Encrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.8: Calling XTEA class through CBC Mode function for encryption

```
CBC_Mode< XTEA >::Decryption Decrypt;
Decrypt.SetKeyWithIV(key, key.size(), iv);
```

Figure 5.9: Calling XTEA class through CBC Mode function for decryption

CHAPTER 6

DISCUSSION

6.1 Introduction

This section discusses the results from the experimentation implementation and presents it using bar graphs.

6.2 Discussion

This section will discuss on the results obtained from the experimentation, which consists of three main data: wall time, CPU time and encrypted file size. Wall time indicates the actual real time that has elapsed; as the name implies, it is the time that has passed on the clock on a wall. CPU time indicates the execution time where the CPU is dedicated to a process. The wall time will be larger than the CPU time, if the CPU is waiting for any task to be done before executing the needed task. However, if the CPU time is greater thus it indicates that the program is executed in parallel (Foo, 2011). Wall time and CPU time is measured to observe the computation speed of the encryption algorithms. Encrypted file size is measured to observe any changes in file size prior and after encryption and decryption. A significant change may indicate that the file may lost some data after the process.

In addition, XTEA is the only encryption algorithms among the three chosen encryption algorithms that only offers 128-key length. Thus, only two encryption algorithms are being discussed for results involving 192-bit and 256-bit key. Figure 6.1 until Figure 6.6 shows the results for encryption, meanwhile Figure 6.7 until

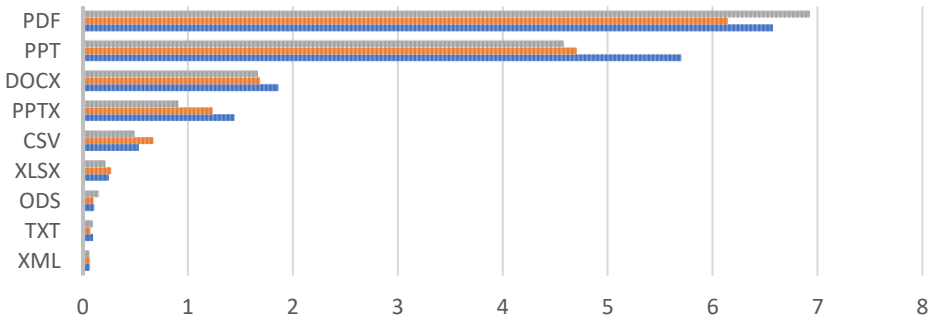
Figure 6.12 shows the results for decryption. After that, the differences in file sizes prior and after encryption including decryption, will be shown.

Firstly, the wall time for encryption will be analyzed. Figure 6.1 shows the results of the wall time of encryption process of each encryption algorithms using 128-bit key length. It can be observed that PDF seems to have the most extensive wall time in each algorithm as compared to other types of files, followed by PPT and DOCX. This shows that the document files can be considered as complex documents, hence takes more time for encryption process. For PDF, XTEA-128 seems to have longer wall time compared to the other two algorithms, and the shortest is Twofish-128. However, for PPT document, LEA-128 performed the longest as compared to the other two, with the shortest being XTEA-128. Overall, the encryption algorithm that showed the most persistent wall time is Twofish-128, as its wall time is often in between LEA-128 and XTEA-128. Twofish-128 has a longer wall time compared to the other two algorithms on CSV and XLSX. Figure 6.2 shows the results of wall time using 192-bit key length for each encryption algorithms using 192-key length. LEA-192 has the longest wall time for most document types as compared to Twofish-192. The longest wall time is around 6 seconds by LEA-192 during encryption of PDF files. Twofish-192 performed longer during encryption of CSV and XML documents. Figure 6.3 shows the results of wall time using 256-bit key for each encryption algorithm using 256-bit key length. It can be observed that LEA-256 performs slower in encrypting all the document files, with a significant gap that can be seen in the results of encrypting PPT file. Twofish-256 performed slower during the encryption of PDF and PPTX.

Next, the CPU time for encryption will be analyzed. Figure 6.4 shows the results of CPU time for encryption using 128-bit key length. XTEA-128 can be seen performing the slowest compared to the other two encryption algorithms, for most document files. The fastest and most consistent performance is by Twofish-128. Figure 6.5 shows the results of CPU time for encryption using 192-bit key length. LEA-192 showed the longest CPU time compared to Twofish-192. For the results of CPU time of encryption using 256-bit key length, it can be observed in Figure 6.6 that Twofish-256 performed better than LEA-256 for majority of the document files. However, Twofish performed slower on PDF and PPTX. Meanwhile, LEA-256 performed the slowest on PPT and DOCX. Overall, the results of CPU time are like

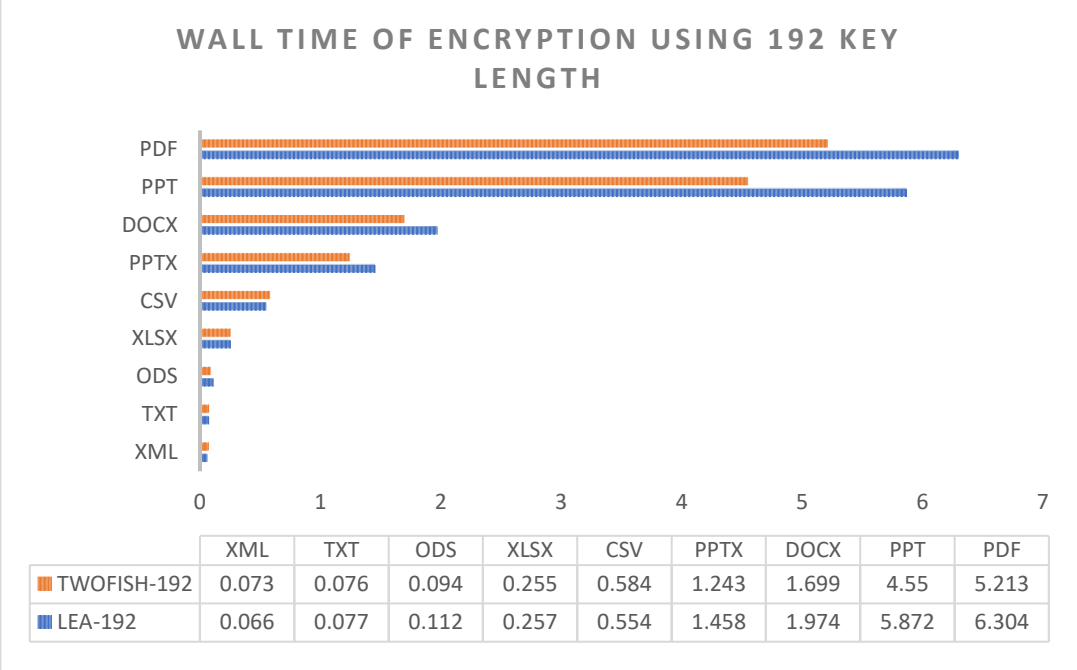
wall time, where XTEA seems to be performing the slowest on 128-bit key length, and Twofish outperforming LEA on the other two key lengths.

Based on the findings, it is expected for XTEA-128 to take a longer time in encrypting a heavy file as it undergoes 64 processing rounds, despite being the simplest algorithm among the three encryption algorithms. LEA-128 has a longer wall time on certain files, and it could be due to it having the lowest computation speed among the others, which is stated in Section 2.1.6. The results shown for 192-bit and 256-bit key length shows that LEA performed slower on majority of the files, and this could be because of the number of processing rounds and the RAM usage. LEA does 28 rounds and 32 rounds on 192-bit and 256-bit key length respectively, as compared to Twofish with the constant number of processing rounds which is 16. Besides that, LEA also produces more RAM footprint compared to Twofish, hence would explain the slow performance on encrypting the files.



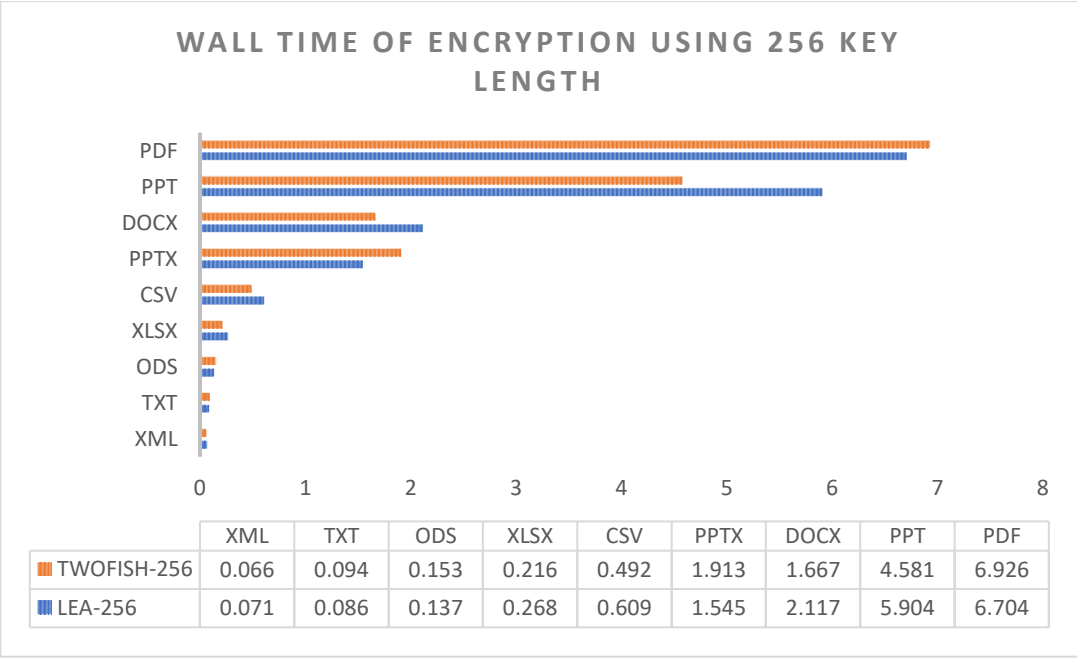
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
XTEA-128	0.066	0.094	0.153	0.216	0.492	0.913	1.667	4.581	6.926
TWOFISH-128	0.068	0.076	0.105	0.273	0.675	1.241	1.69	4.704	6.143
LEA-128	0.07	0.0957	0.1123	0.2467	0.5327	1.4483	1.863	5.7017	6.572

Figure 6.1: Wall time of encryption using 128-key length



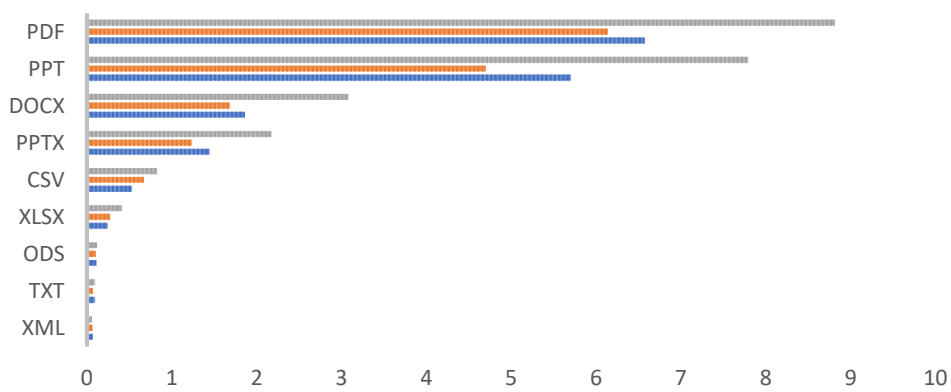
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-192	0.073	0.076	0.094	0.255	0.584	1.243	1.699	4.55	5.213
LEA-192	0.066	0.077	0.112	0.257	0.554	1.458	1.974	5.872	6.304

Figure 6.2: Wall time of encryption using 192-key length



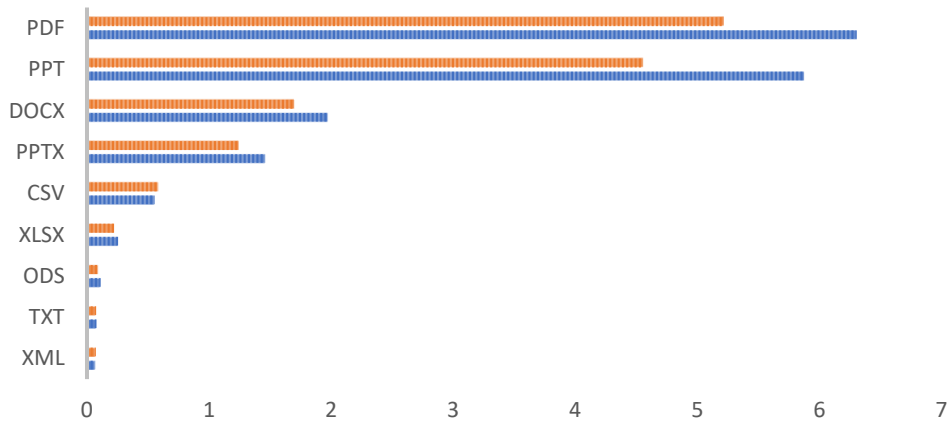
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-256	0.066	0.094	0.153	0.216	0.492	1.913	1.667	4.581	6.926
LEA-256	0.071	0.086	0.137	0.268	0.609	1.545	2.117	5.904	6.704

Figure 6.3: Wall time of encryption using 256-key length



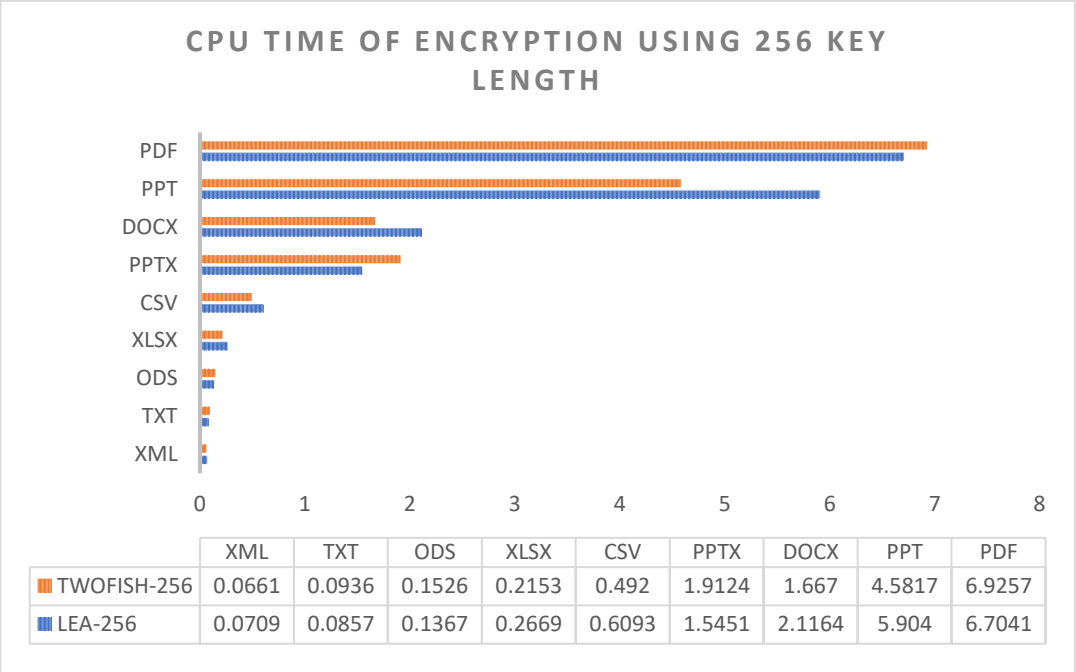
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
XTEA-128	0.0602	0.0971	0.1195	0.4083	0.8303	2.1747	3.0869	7.7946	8.818
TWOFISH-128	0.0679	0.0756	0.1053	0.2737	0.6744	1.241	1.6892	4.7045	6.1429
LEA-128	0.06999	0.0958	0.1123	0.2472	0.5329	1.448	1.8632	5.7016	6.5714

Figure 6.4: CPU time of encryption using 128-key length



	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-192	0.0737	0.0763	0.0925	0.2248	0.5839	1.2434	1.6989	4.5506	5.2136
LEA-192	0.066	0.0766	0.1126	0.2566	0.5546	1.4587	1.9737	5.872	6.3045

Figure 6.5: CPU time of encryption using 192-key length



	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-256	0.0661	0.0936	0.1526	0.2153	0.492	1.9124	1.667	4.5817	6.9257
LEA-256	0.0709	0.0857	0.1367	0.2669	0.6093	1.5451	2.1164	5.904	6.7041

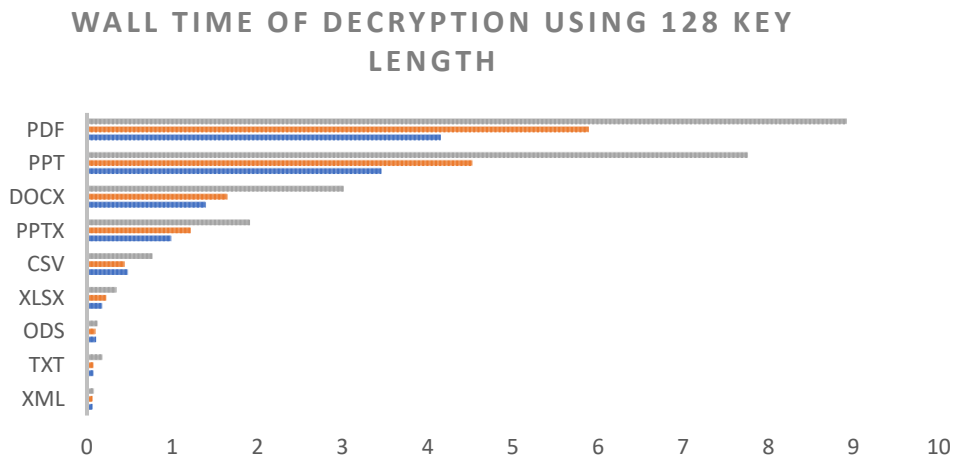
Figure 6.6: CPU time of encryption using 256-key length

All encrypted documents were saved to test the speed of decryption among the three encryption algorithms. Firstly, the wall time will be analyzed. Figure 6.7 shows the results of wall time during decryption using 128-bit key length. XTEA-128 performed the slowest, especially on PDF and PPT, where the gap between XTEA-128 and Twofish-128 is clearly significant. Surprisingly, LEA-128 performed better compared to the others, as opposed during encryption. As seen in the result for PDF and PPT, the gap between LEA-128 and Twofish-128 is considerably significant. Figure 6.8 shows the results of wall time for decryption under 192-bit key length. Twofish-192 appeared to be the slowest in decrypting as compared to LEA-192. Lastly, the results for wall time during decryption using 256-bit key length can be seen on Figure 6.9. It can be observed that both algorithms performed similar in speed, with each algorithm being slow at the same amount of document files.

Next, the CPU time of decryption using each key length will be analyzed. Figure 6.10 shows the CPU time for decryption using 128-bit key length. It can be observed that XTEA-128 performed the slowest compared to others, followed by Twofish-128. The results for decryption using 192-bit key length can be observed at Figure 6.11, where Twofish-192 performed the slowest compared to LEA-192. Lastly, Figure 6.12 shows the results of CPU time for decryption using 256-bit key length.

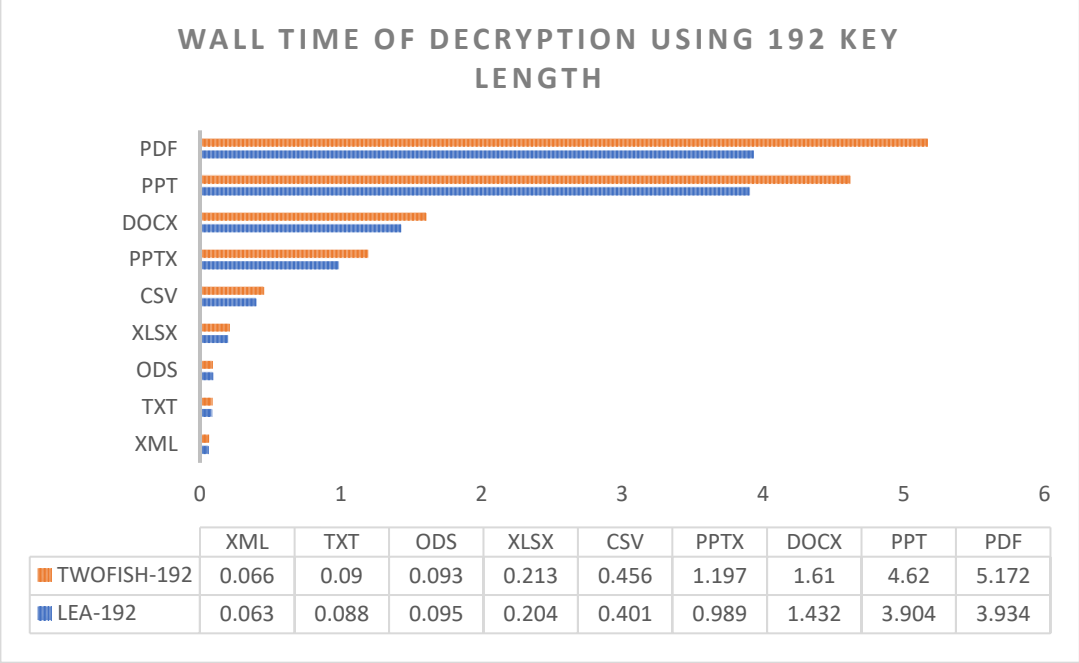
The performance from both Twofish-256 and LEA-256 are similar, where both are slow at the same amount of certain document files. Overall, the results of CPU time for decryption are similar with the wall time.

Based on the findings, it is expected for XTEA-128 to perform much longer on decryption as it has much more processing rounds compared to others. Besides that, XTEA has the slowest computation speed among the others, even though it consumes the least RAM usage. LEA performed the fastest among the three on most of the files in each key lengths, and this could be due to its decryption algorithm having the same structure as its encryption algorithm. Twofish however displayed a longer time consumption when compared with LEA, and that could be due to its complex key scheduling.



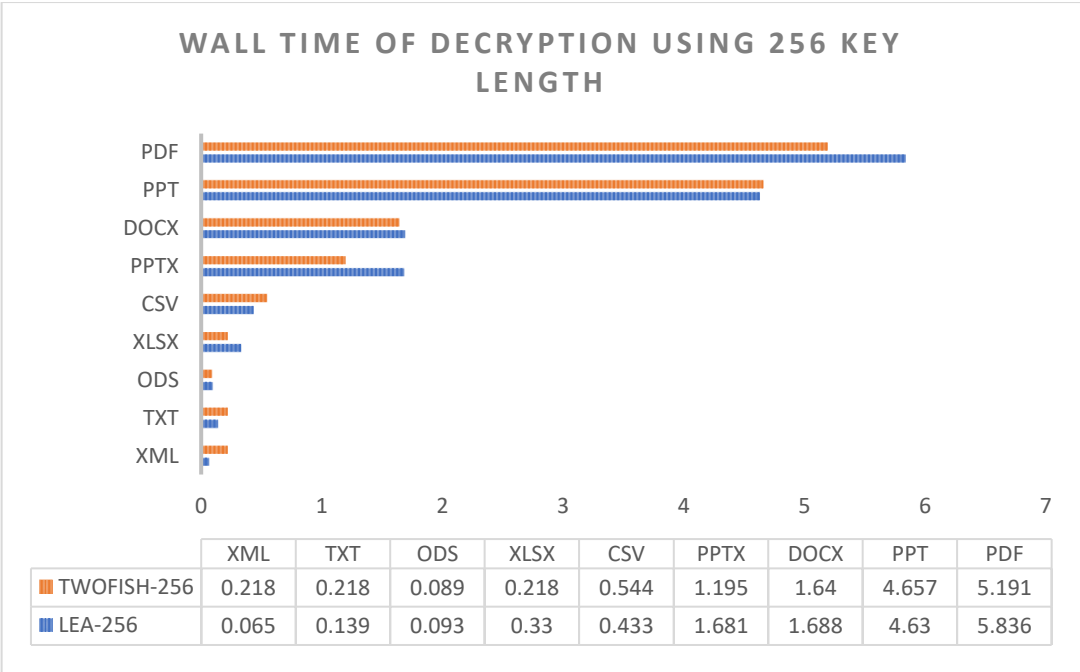
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
XTEA-128	0.086	0.185	0.131	0.349	0.773	1.917	3.018	7.759	8.923
TWOFISH-128	0.068	0.083	0.1	0.23	0.445	1.219	1.647	4.525	5.89
LEA-128	0.066	0.081	0.107	0.184	0.484	0.992	1.395	3.456	4.153

Figure 6.7: Wall time of decryption using 128-key length



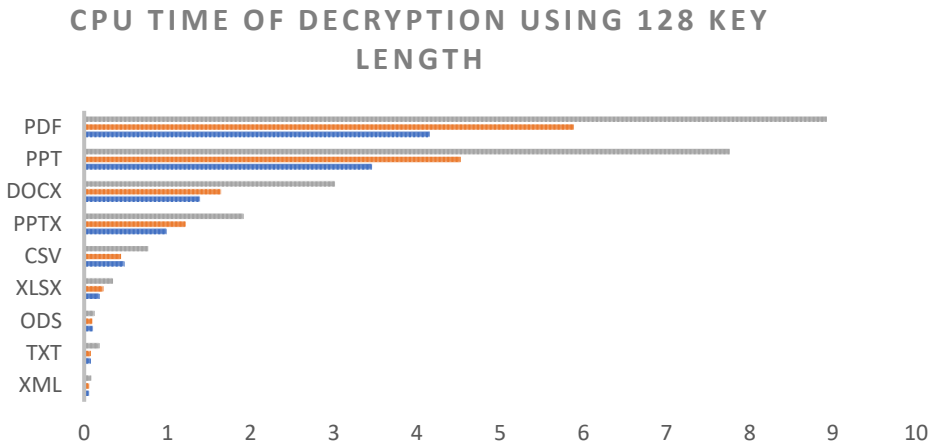
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-192	0.066	0.09	0.093	0.213	0.456	1.197	1.61	4.62	5.172
LEA-192	0.063	0.088	0.095	0.204	0.401	0.989	1.432	3.904	3.934

Figure 6.8: Wall time of decryption using 192-key length



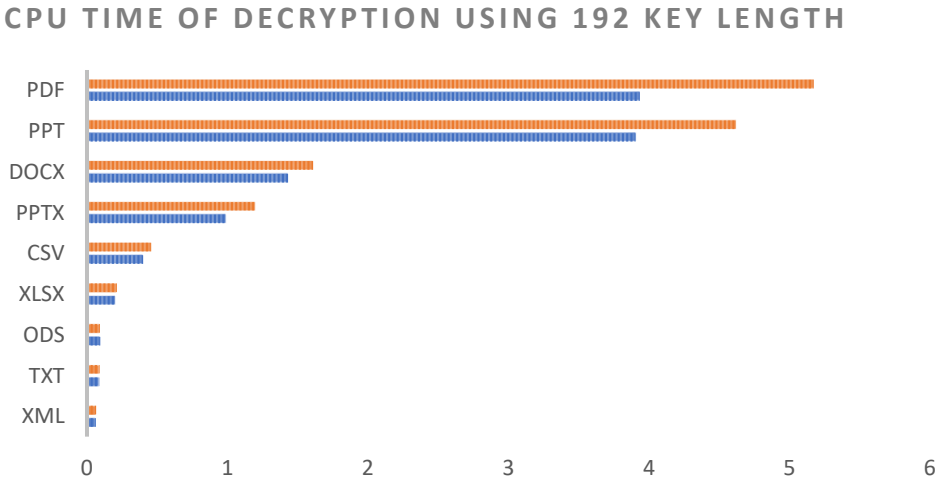
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-256	0.218	0.218	0.089	0.218	0.544	1.195	1.64	4.657	5.191
LEA-256	0.065	0.139	0.093	0.33	0.433	1.681	1.688	4.63	5.836

Figure 6.9: Wall time of decryption using 256-key length



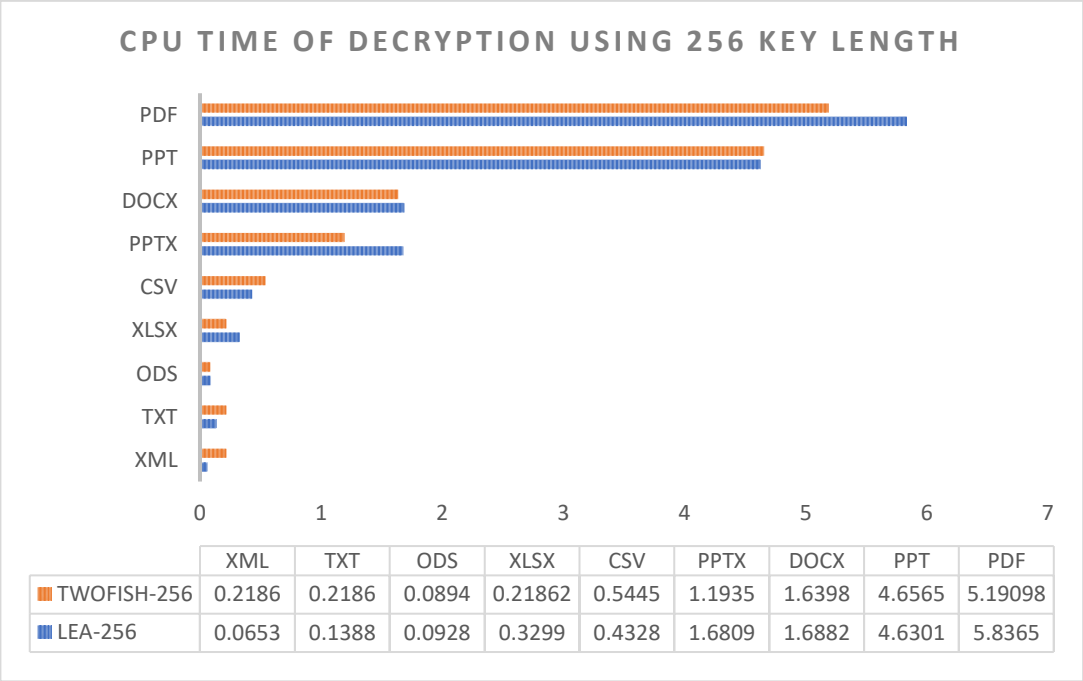
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
XTEA-128	0.0857	0.1855	0.1293	0.3491	0.7736	1.9168	3.0182	7.7589	8.9234
TWOFISH-128	0.0681	0.0833	0.0994	0.2301	0.444	1.2194	1.6471	4.5254	5.8904
LEA-128	0.0658	0.0812	0.1073	0.1845	0.4843	0.9919	1.3958	3.4564	4.1523

Figure 6.10: CPU time of decryption using 128-key length



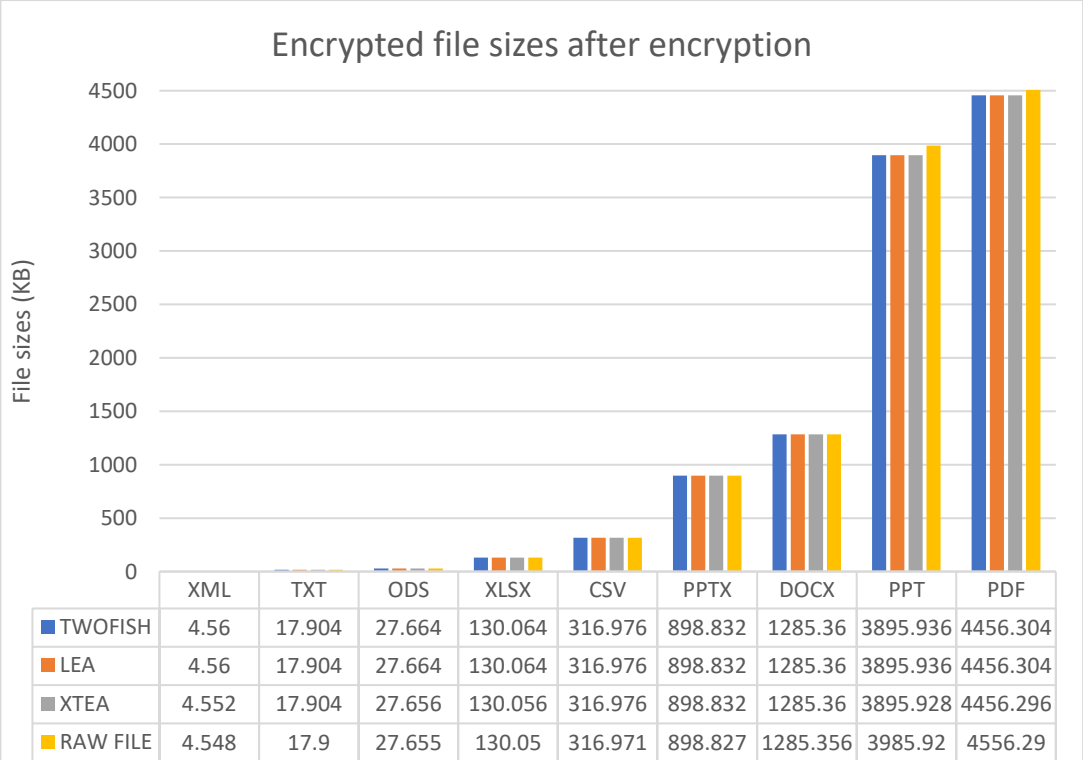
	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-256	0.0656	0.0908	0.0931	0.2135	0.4561	1.1971	1.6101	4.6194	5.1725
LEA-256	0.0633	0.0885	0.0951	0.2045	0.4006	0.9897	1.4322	3.904	3.934

Figure 6.11: CPU time of decryption using 192-key length



	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH-256	0.2186	0.2186	0.0894	0.21862	0.5445	1.1935	1.6398	4.6565	5.19098
LEA-256	0.0653	0.1388	0.0928	0.3299	0.4328	1.6809	1.6882	4.6301	5.8365

Figure 6.12: CPU time of decryption using 256-key length



	XML	TXT	ODS	XLSX	CSV	PPTX	DOCX	PPT	PDF
TWOFISH	4.56	17.904	27.664	130.064	316.976	898.832	1285.36	3895.936	4456.304
LEA	4.56	17.904	27.664	130.064	316.976	898.832	1285.36	3895.936	4456.304
XTEA	4.552	17.904	27.656	130.056	316.976	898.832	1285.36	3895.928	4456.296
RAW FILE	4.548	17.9	27.655	130.05	316.971	898.827	1285.356	3985.92	4556.29

Figure 6.13: Encrypted files after encryption

The encrypted file size of each of the document files are recorded to be compared with the raw size of the document types. The results obtained showed that ciphertext produced by LEA and Twofish are the same sizes for all key lengths. Thus, the results for all algorithms will be presented as such in Figure 6.13. LEA and Twofish produces the same sizes of ciphertext, which are a bit larger than XTEA. The difference of ciphertext sizes between the algorithms is very minimal, and this could depend on how much data is stored in the header of the file (Kobayashi and Kiya, 2018).

CHAPTER 7

IMPLEMENTATION

7.1 Introduction

This section discusses the development progress of this project, involving the components being developed with figures and step-by-step explanation.

7.2 Implementation

This section is divided into two sections, which discusses on the development environment and components or functionalities that are currently available in this system.

7.2.1 Development Environment

The development of this project is done on a local web server using XAMPP. XAMPP allows developers to build websites offline on a local server on the developer's computer, and it includes Apache, MySQL, PHP, and Perl (VMWare, 2021). It is very convenient for this system as the language mainly used for this system is PHP, and the MySQL database will be used for all database related functions. VS Code is used as the text editor, and GitHub Bash is used as the CLI to run and install npm and composer lines.

The libraries and packages being adapted into this project, are installed using npm and composer. The libraries used in this project are backward-timer, PHPMailer, HTMLPurifier, and phpseclib.

7.2.2 Components

The components in this project can be explained according to the modules exists in the system. First off, the User Registration and Authentication module involves registration and login process as an account is needed to encrypt/decrypt files and share it with other users.

Registration page can be accessed by clicking on the 'Sign up' button on the top right of the landing page as shown in Figure 7.1. The registration page contains a form for users to fill for account registration.

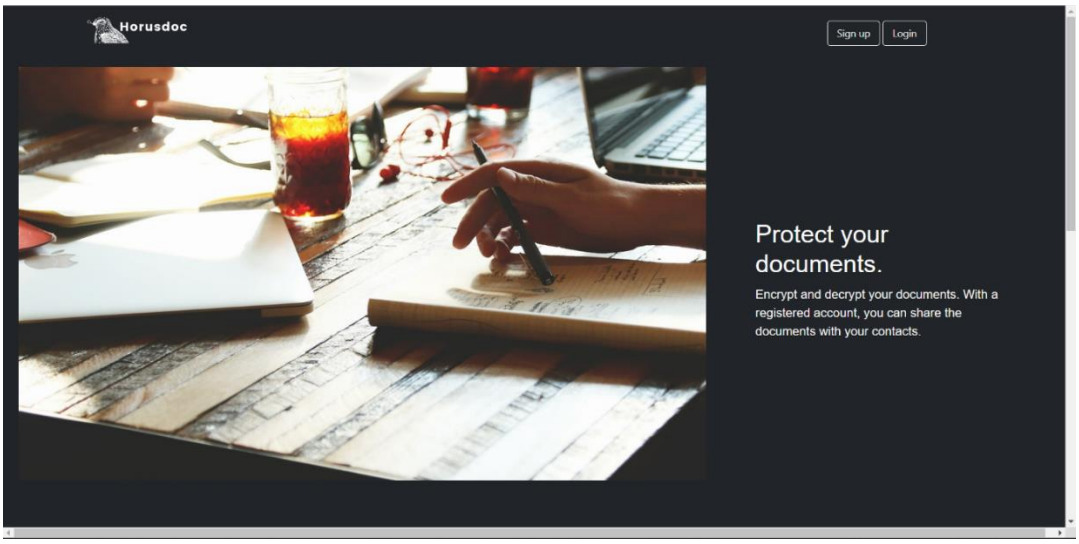


Figure 7.1: Landing page of the project

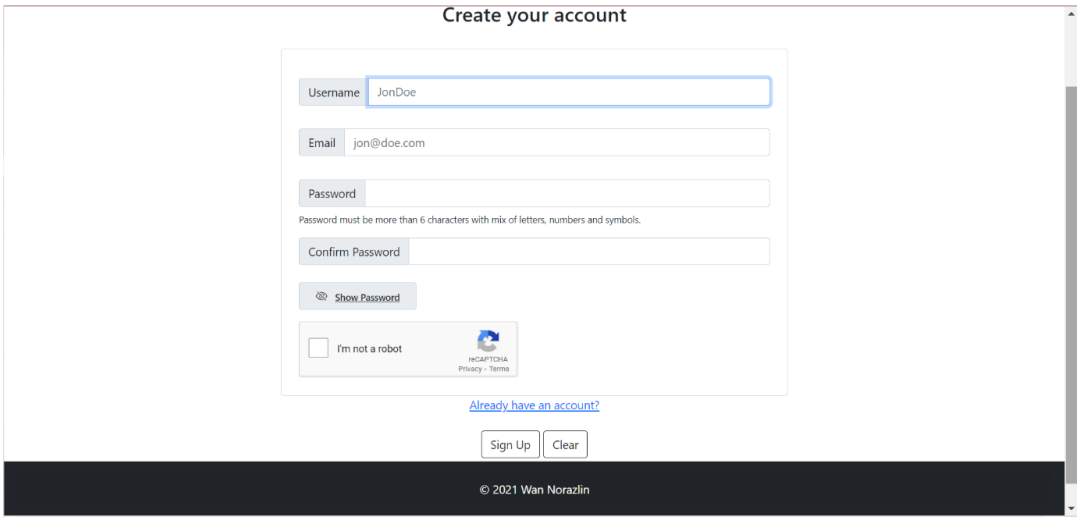


Figure 7.2: The Registration form on registration page

A valid registration requires users to submit valid registration credentials, which are a valid email address that has not yet been registered, a valid username and a password. If users leave any of the field empty, the field will be highlighted in red such as in Figure 7.3. Besides that, users also need to pass the reCAPTCHA located under the 'Show Password' button. The 'Show Password' button allows users to see their typed-in passwords in case of any mistypes, as the password visibility feature provided by HTML disappears when you click away from the password input field.

If users attempt to submit the form with empty fields, the submission is halted, and users will get this HTML alert on the required input fields. All the input fields are required to be filled in. Users will be informed if the typed username and email is taken, prior submission. Both inputs are being checked once users clicks away from the currently focused input field.

The reCAPTCHA has its own time limit, where if users already passed the test but left it for a given amount of time, the reCAPTCHA will become invalid and expired. Users will get notified when the reCAPTCHA expires. The reCAPTCHA being used in this project is the second version, where it consists of puzzles for users to solve when needed.

If users submit the registration form with valid credentials, the credentials will be stored in the database and users can access their online accounts by logging in.

An email verification link will be sent to the user's email for users to verify their email. After a successful registration, users will be greeted with a confirmation page.

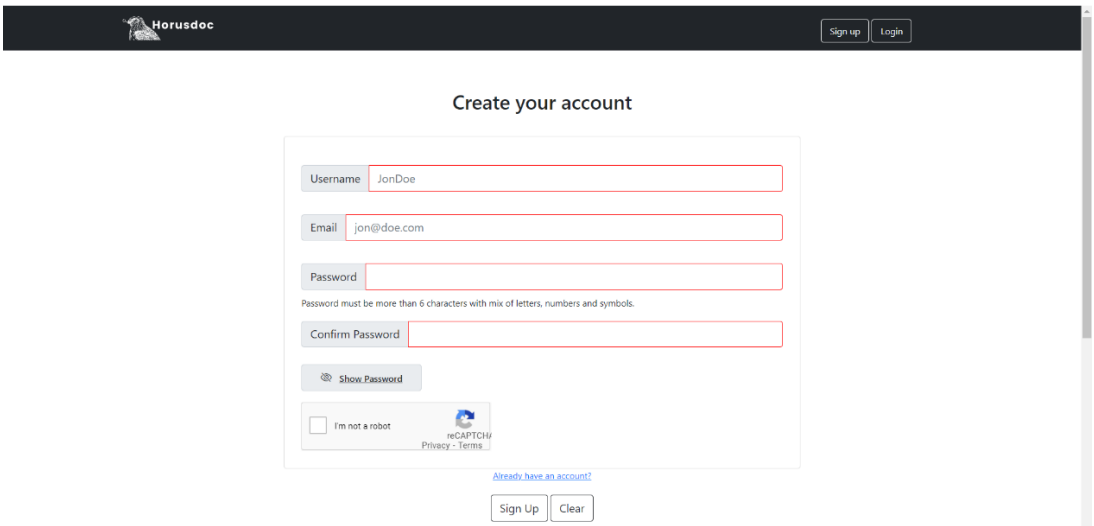


Figure 7.3: Demonstration of the input fields in the registration form when empty

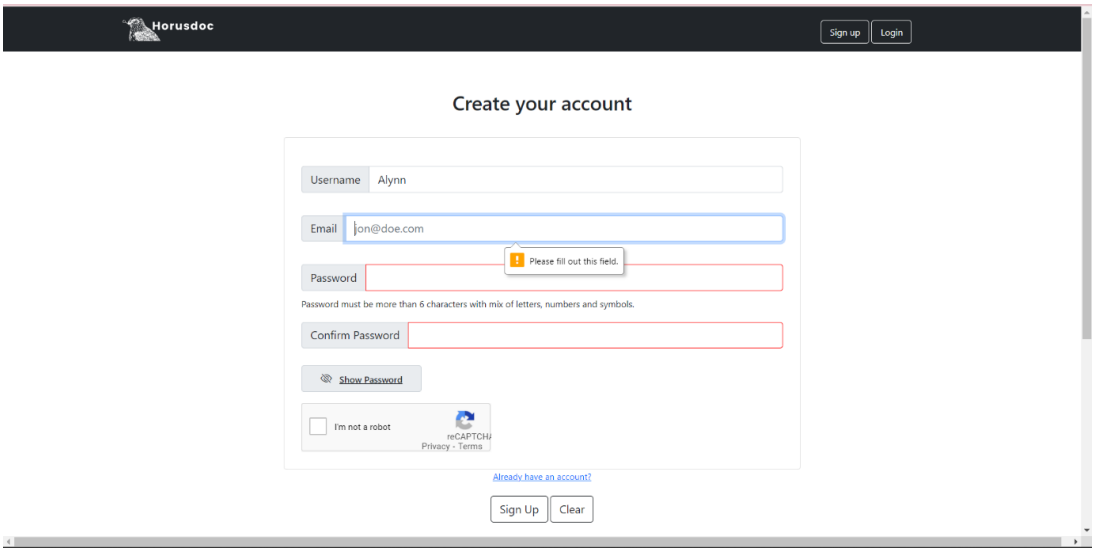
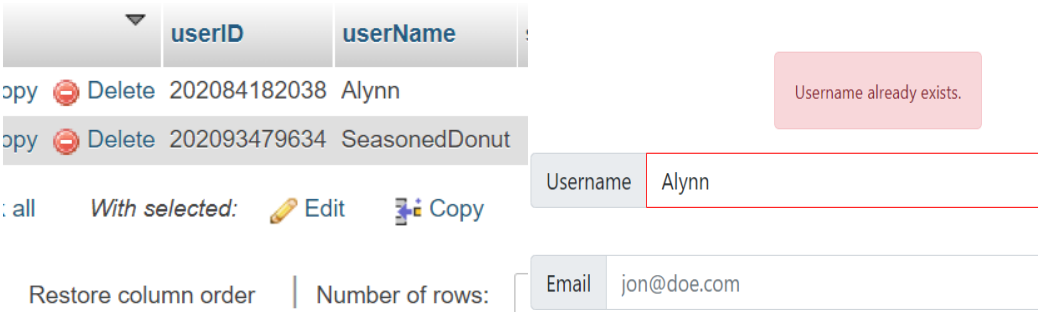
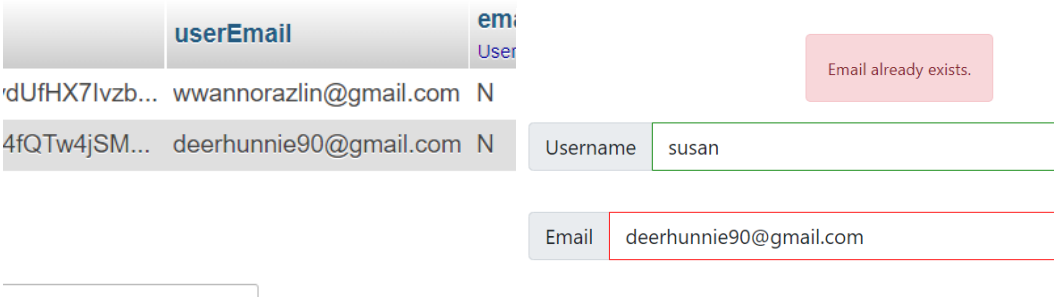


Figure 7.4: Demonstration of the empty input fields in the registration form when submitted



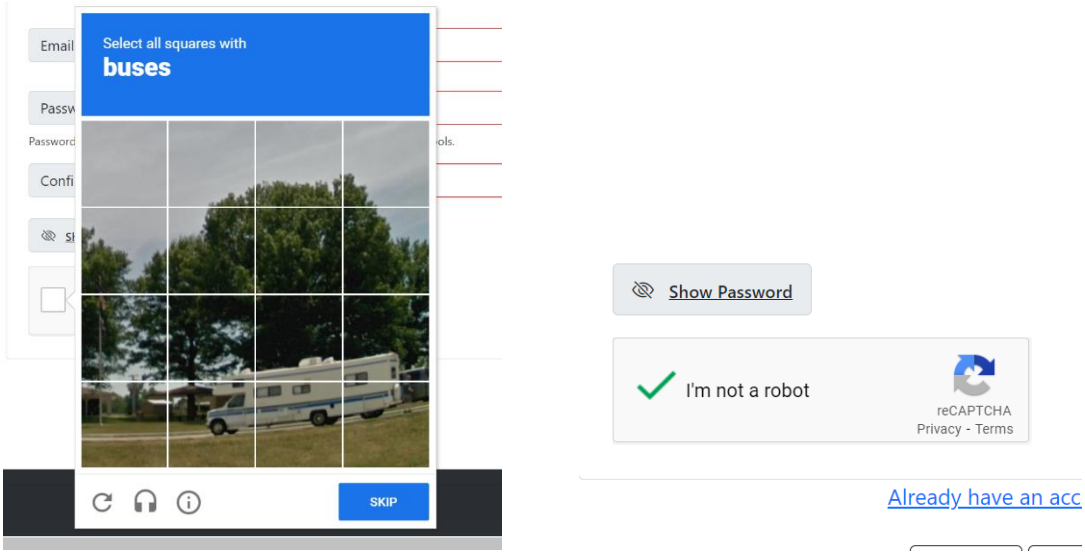
(a) (b)

Figure 7.5: Username validity check



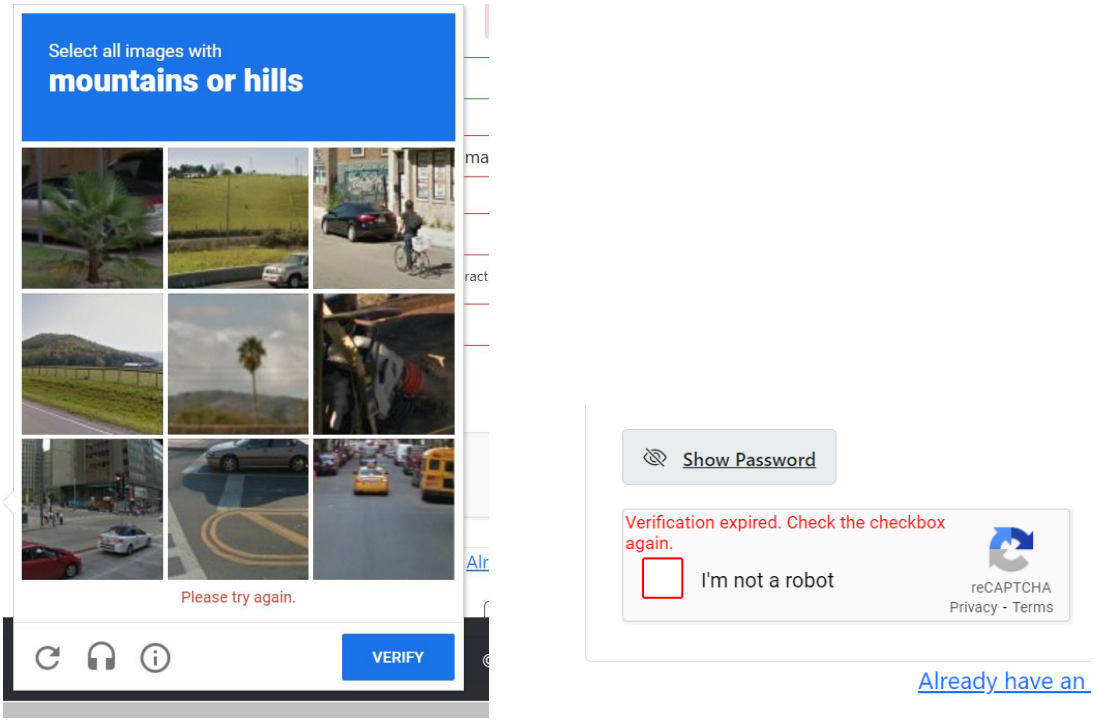
(a) (b)

Figure 7.6: Email validity check



(a) (b)

Figure 7.7: The process of reCAPTCHA puzzle



(a) (b)

Figure 7.8: The error messages produced by reCAPTCHA

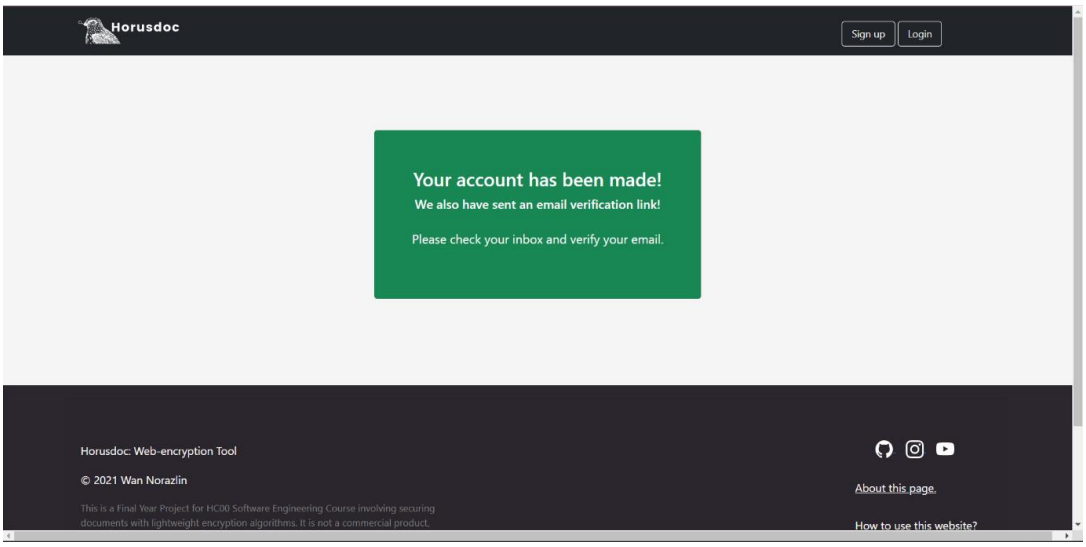


Figure 7.9: The confirmation message after successful registration

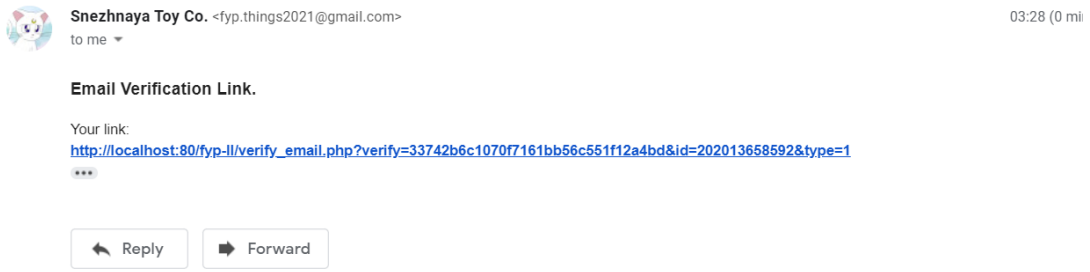


Figure 7.10: Email verification sent to user's inbox

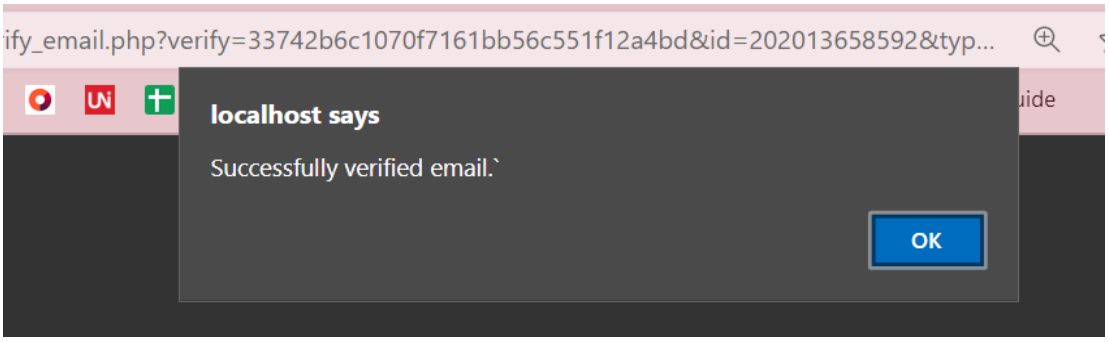


Figure 7.11: Alert message upon successful email verification

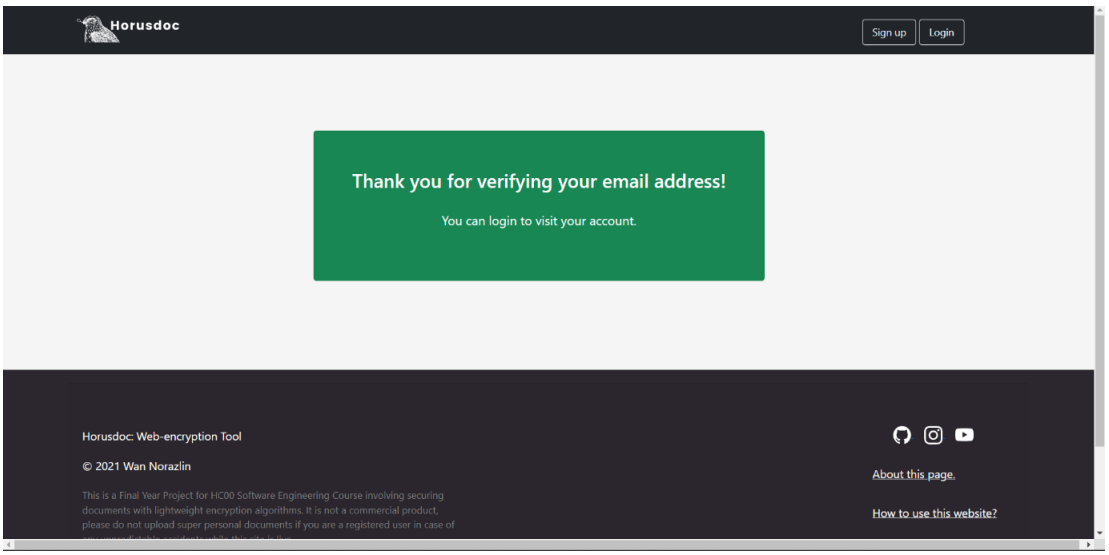


Figure 7.12: Confirmation message after successful alert message

	userEmail	email_verified Users verify email or not
E0wneUdtXPNZVyATUHigRzAS.9...	wwannorazlin@gmail.com	Y
sILMp.B63DhDPCWF0N4fQTw4jSM...	deerhunnie90@gmail.com	N

Figure 7.13: Status of user's email verification being updated

The second component is the login page. Users can navigate to the login page by clicking the 'Login' button on the landing page or clicking the 'Already have an account?' link on registration page. Users are required to submit valid login credentials, which are their registered email and account password. Users also need to pass the reCAPTCHA prior submitting the form. The error checking on login page is like registration page. There is a different link on login page, which is the 'Forgot password?' link. Users could click this link if they forgot their password and want to reset it. The reset password falls under the User Account Management module as users are allowed to change basic information.

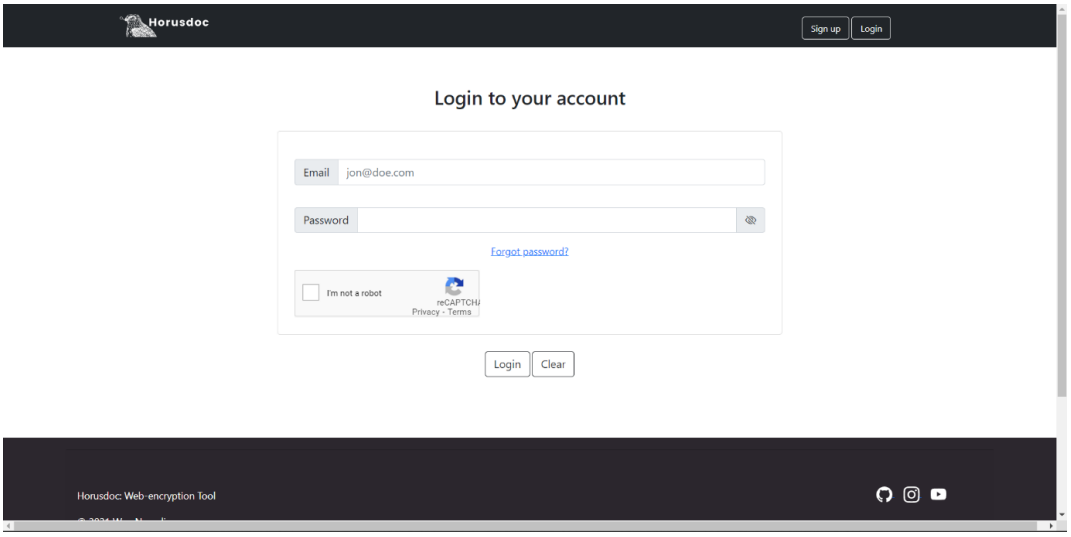


Figure 7.14: The login page

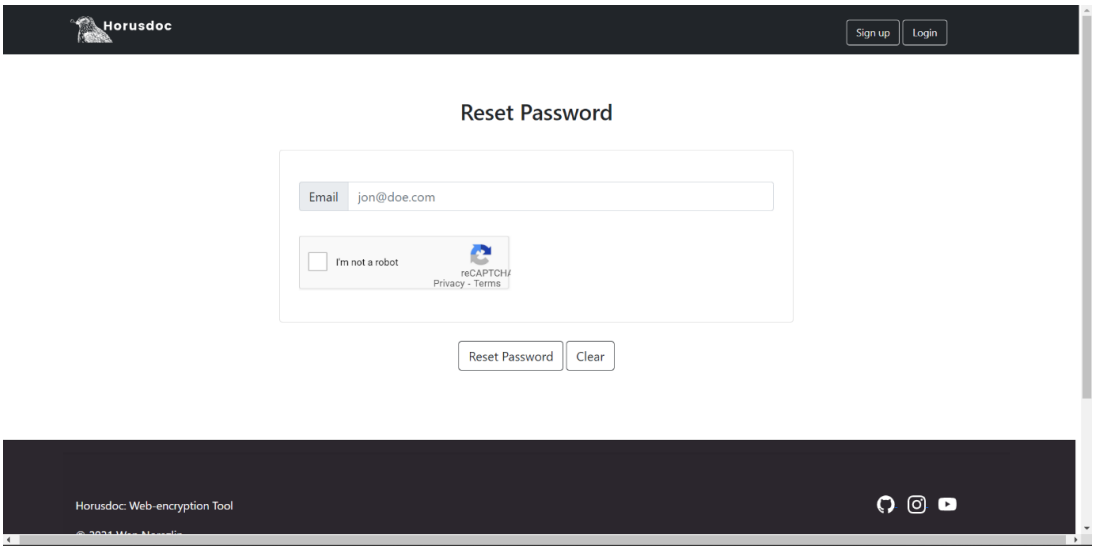


Figure 7.15: The page for password resetting

Figure 7.15 shows the page for users to reset their password. Users will need to submit their email so that a password reset link can be sent to their email. Once users click the link provided, an alert message will show if the verify token on the URL matches with the verify token in database.

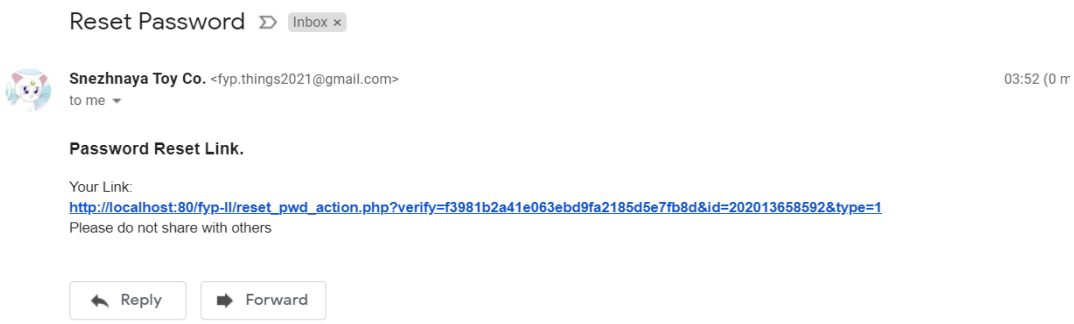


Figure 7.16: Password reset link sent to user's email inbox

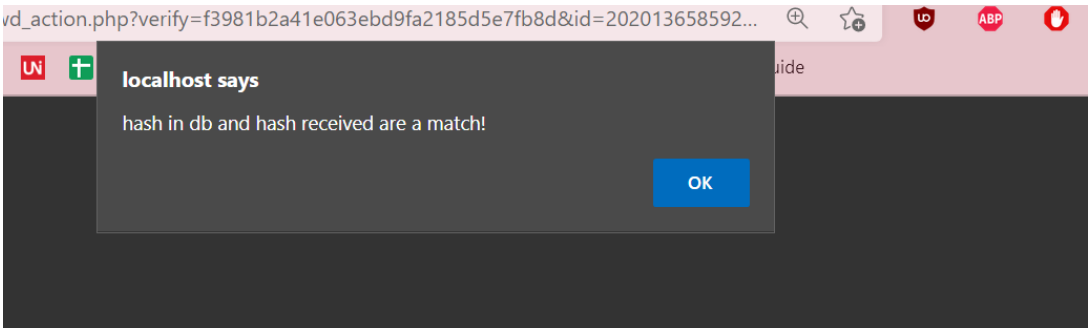


Figure 7.17: Alert message if the verify tokens match

This alert will only be available during development phase. If it is the same tokens, users will be navigated to the password reset page which is shown in Figure 7.18. Users will then need to submit their email and new password. The 'UID' shown on the password reset page is the user's unique ID. It is shown on the page to make it easier to be stored in session for querying process later the server-side.

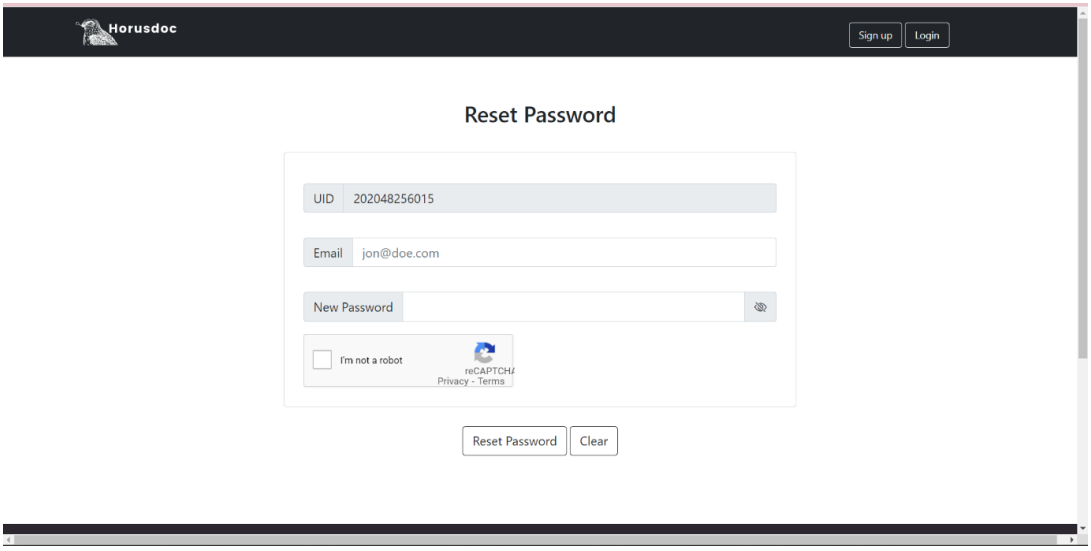


Figure 7.18: Password reset page

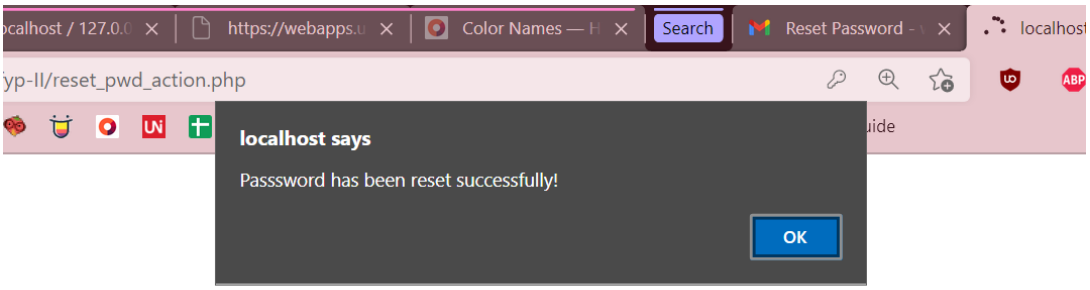


Figure 7.19: Alert message for successful password reset

If the password reset is successful, users will get an alert such as in Figure 7.19 and will be able to login with their new password. Users will be redirected to the login page after clicking 'OK' on the alert message.

Other components for User Account Management are in the 'Settings' page in user account, where users can update their basic information and deactivate their account, as shown in Figure 7.21.

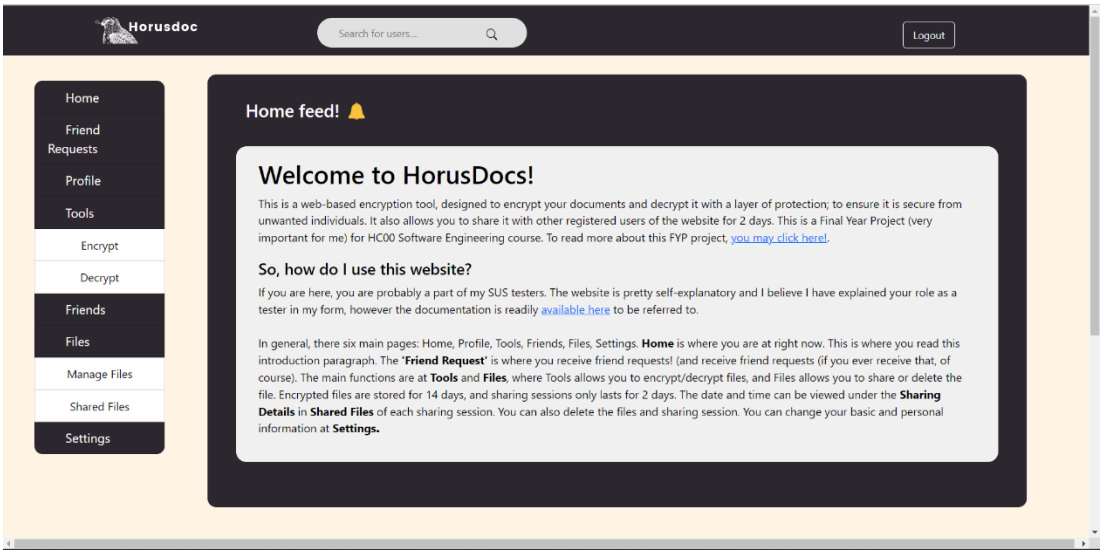


Figure 7.20: The landing page for user's account

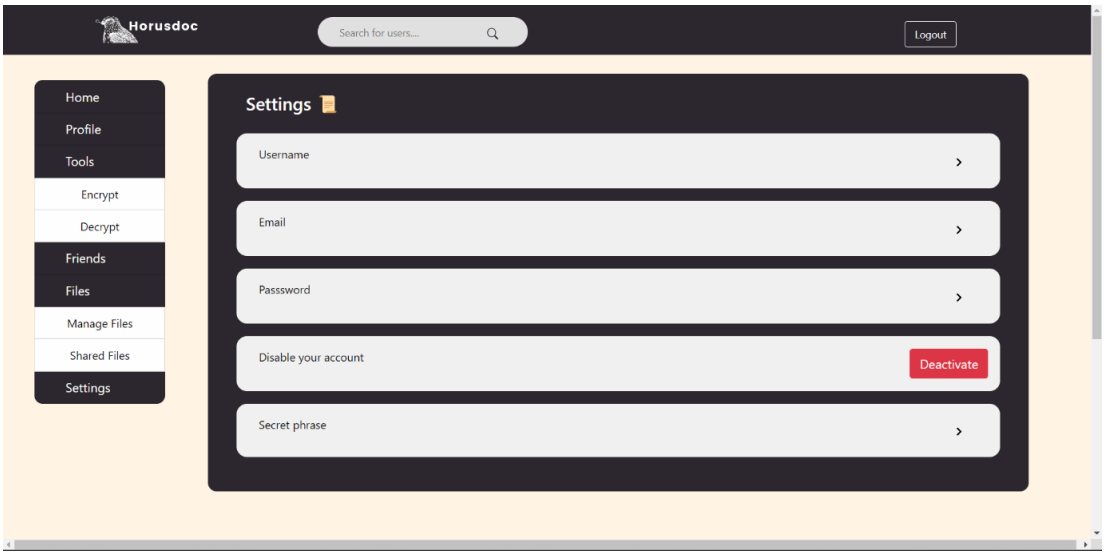


Figure 7.21: The 'Settings' page

The 'Settings' page allows five user's information to be modified or updated, which are username, email, password, account deactivation and secret phrase. The procedure for updating the user's information varies according to the sensitivity level of the data. Username and email have the same procedure, where users need to provide the new user detail and their current password for verification. The procedure for changing user's password involves OTP (One-Time-Password) through email. Users need to provide their new password, current password, and the OTP within a time limit. Users need to click the 'Request OTP' button to receive their OTP, which can be re-clicked after 5 minutes as seen in Figure 7.24(c). A timer countdown is shown under the OTP input field. After a successful update, an alert message will be shown to confirm users about the changes. A successful password update will force users to be logged out, so that users can re-login with their new passwords.

Deactivation button will direct user to a page that prompts user to provide their password and OTP prior deactivation for user authentication. Once user submit the deactivation form, the account will be deleted, including any files stored within the system.

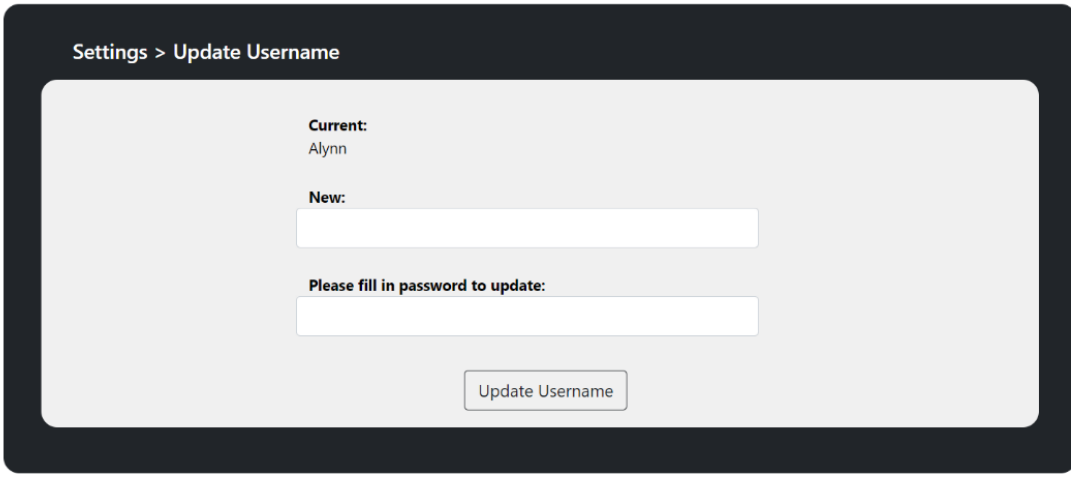


Figure 7.22: Page for updating username

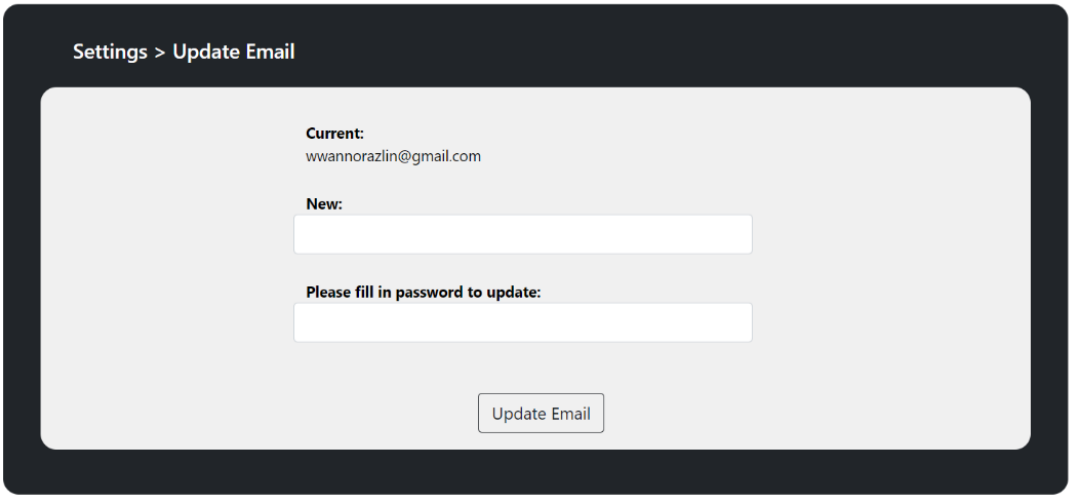


Figure 7.23: Page for updating email

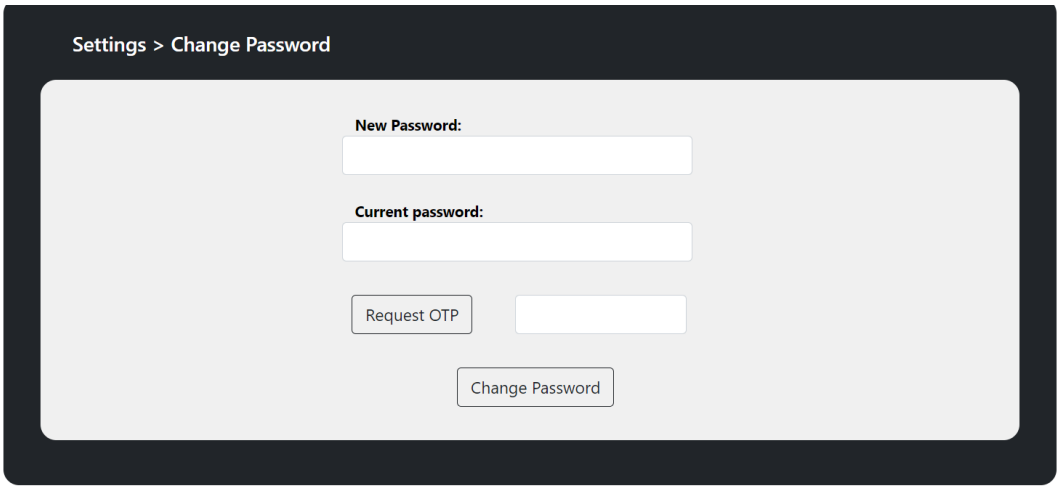


Figure 7.24: Page for changing password

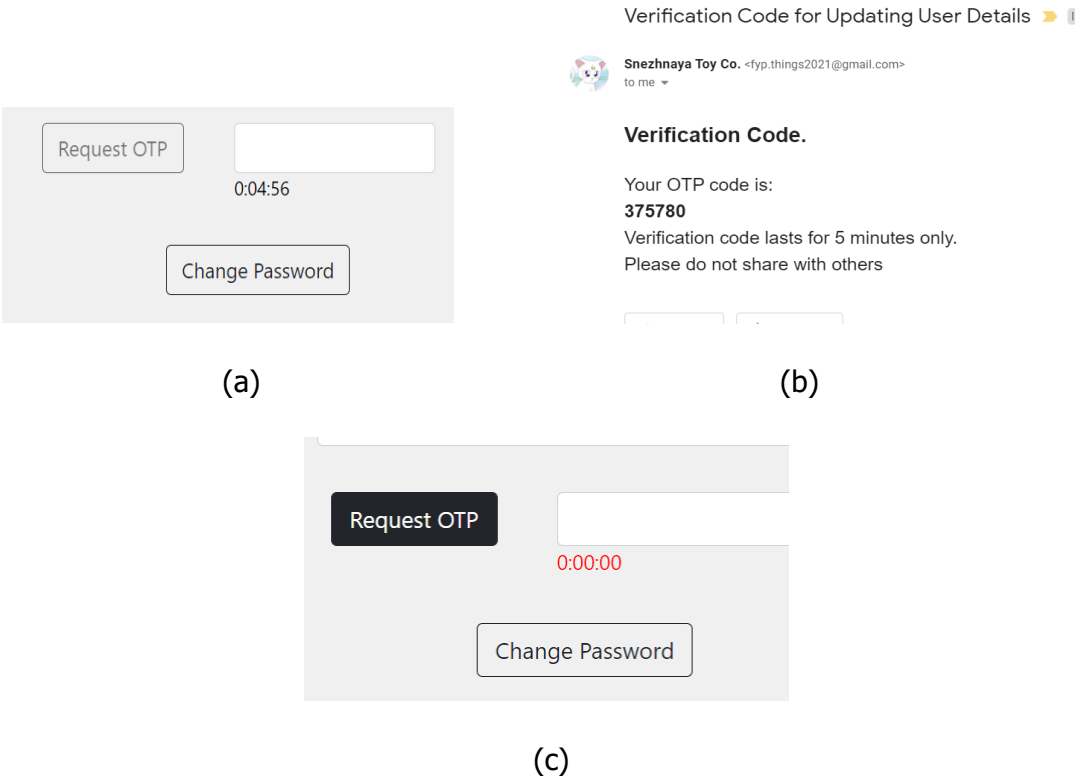


Figure 7.25: The OTP (One-Time-Password) process

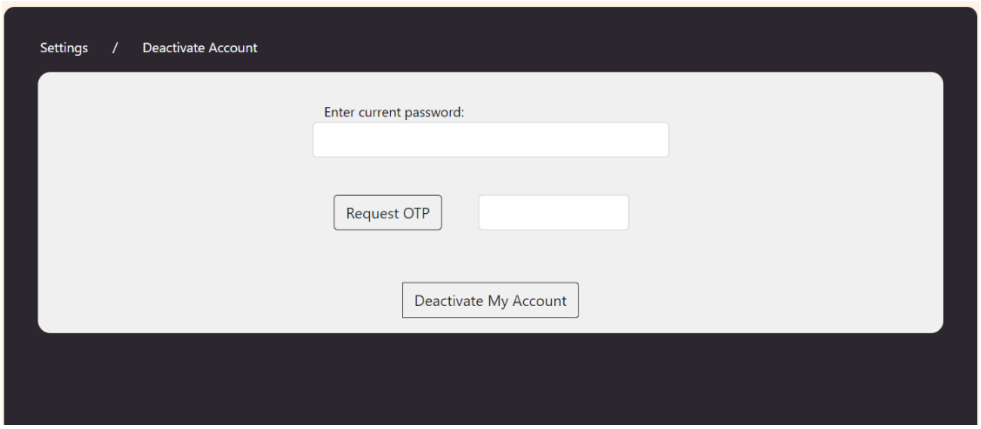


Figure 7.26: Deactivate account page

Next, components in Web-based Encryption Tool module will be explained. This module requires so that users are allowed to encrypt/decrypt document files. As of now, the completed component is within the 'Tools' page in user account. The encryption page requires users to submit a file and a passphrase/key. The

passphrase/key however can be auto generated by checking the checkbox prior submitting. The flow of the encryption process can be seen in Figure 7.27. It is worth to note that the encrypted file is stored in the filesystem as it is faster for file retrieval later during decryption process. However the uploaded raw fie is immediately deleted as soon as encryption process is completed using the PHP unlink() function. Decryption process goes through a similar process, which can be seen in Figure 7.28. The raw file must be saved temporarily in the filesystem so that the decrypted content can be written into a file and be assigned its extension. After sending the file to the browser for download, it is deleted using the same PHP function. Due to this, user only have one chance to download the file. User cannot re-download the file after opening it in browser or cancelling it since the file does not exist anymore in the directory.

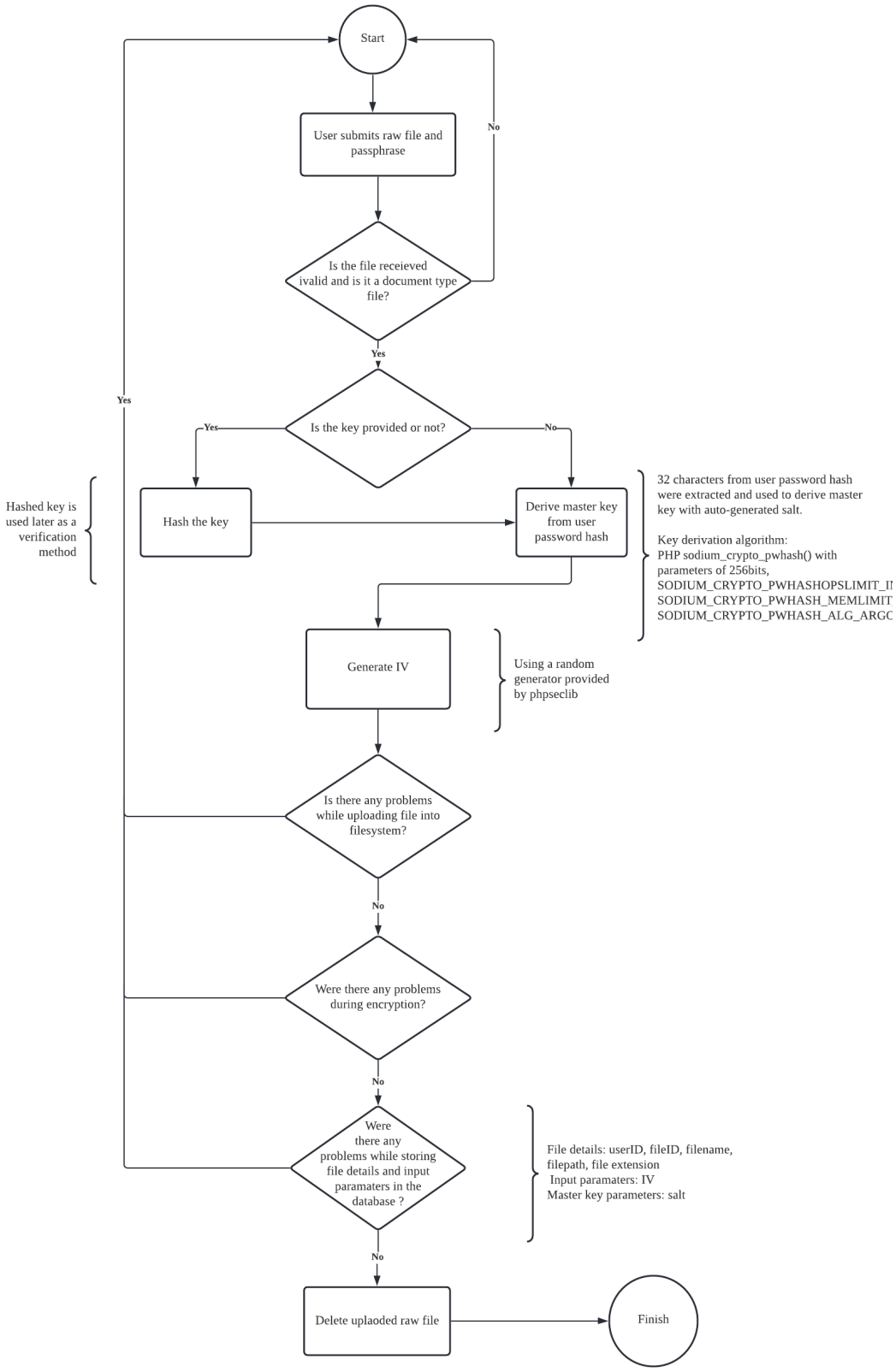


Figure 7.27: Flowchart of encryption process implemented in the system

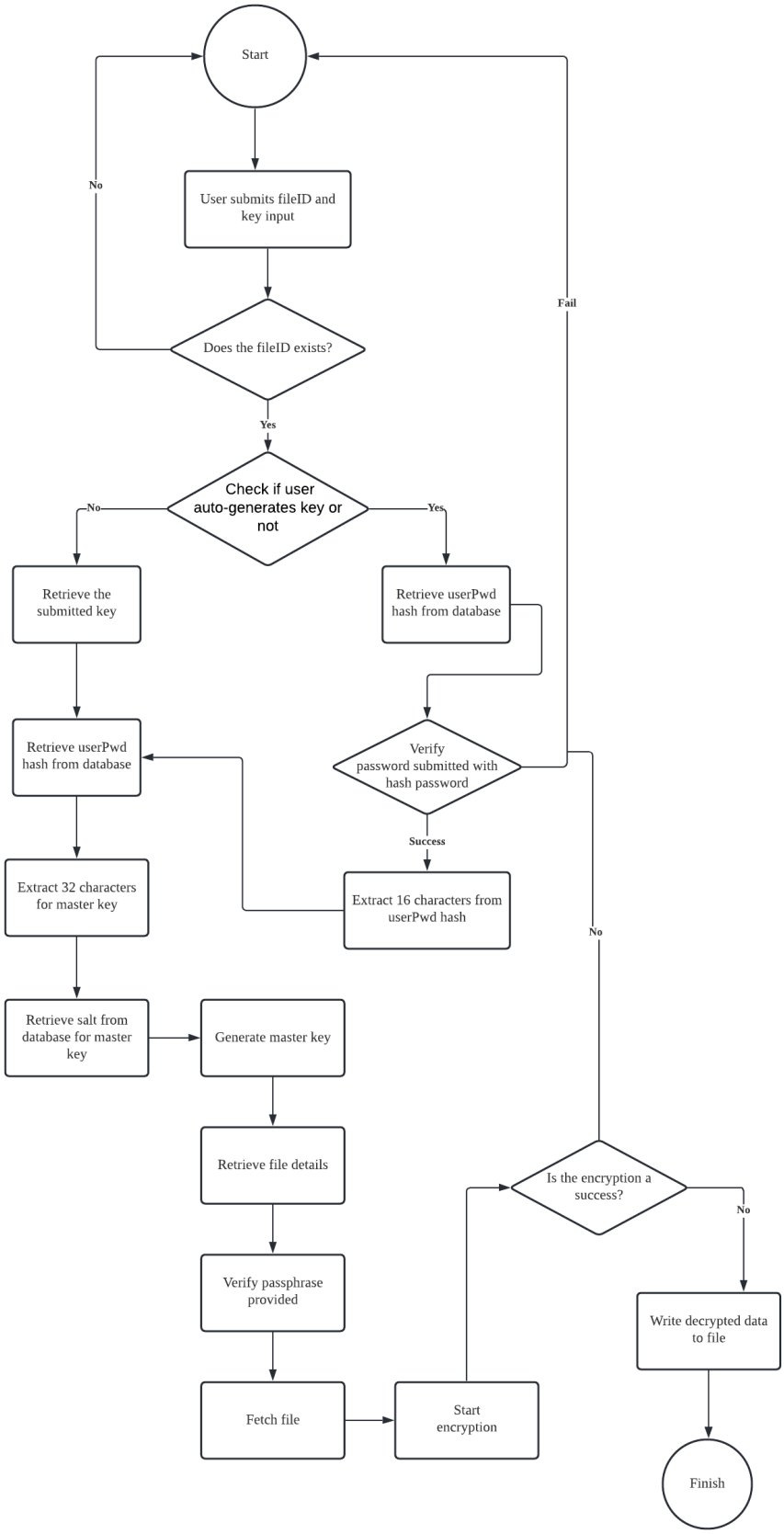


Figure 7.28: Flowchart of decryption process implemented in the system

Contact List and Access Policy Management module should allow users to add other users as friends, unfriend their contacts (friends) and block other user accounts. This module is also necessary for file sharing; thus, it should be implemented. As of now, users can send friend requests and accept them, however unable to unfriend or block them.

The component of this module consists of the search user bar, the 'Friend Request' page and the 'Friend' page. Users can search for other users by searching their name, email address or user ID. The string that is inputted will be used to query the database, specifically the useraccount table. Then, the user can send a friend request to the user they want to befriend or connect. Users receiving friend requests can accept the friend request, however the function to decline has not been added yet. Thus, this module is not complete however it allows File Sharing module to function. The table used for this module is the userfriends table, where it records the status of friend requests and connections between the users. The flowchart of this component can be seen in Figure 7.29.

File Sharing Management module allows users to share files among other users and contains several tasks to manage sharing sessions. The component for this module is implemented under 'Manage Files' and 'Shared Files'. 'Manage Files' allow users to share files and delete files, while 'Shared Files' allows users to view file, delete sharing sessions and see session details. User will need to add another user prior to be able to share a file. Once a user has 'friends', user can choose which friends to share a file with.

The interface to choose the users and submitting access phrase are done using bootstrap modals, where the values are then fetched using html *data-bs-* attribute and JQuery, then sent to the server-side using Ajax. This is so that the process does not involve frequent browser reloads as it can take a long time if the connection is bad. Once the server retrieves the data and process it, the responds will be echo-ed to Ajax, and the corresponding modal will be executed. Any errors or even successful transaction has messages that is stored in sessions. It will later be showed on the client side and be unset after user closes the alert containing the messaged. Every user that are selected will receive an email notification containing the sharing session link. After 2 days, the session link will be deleted automatically.

The last module is the E-mail Notification module where it sends email to users that are being selected for file sharing by another user. This module is also applied for verification and authentication purposes such as email verification, OTP and password reset. The main component of this module is the implementation of PHPMailer library. The library allows developers to send emails to users programmatically.

A google account was made specifically for this system as to send emails programmatically through PHPMailer to Gmail's SMTP Server, some changes need to be done to the Google Account Security Settings. A password needs to be generated to allow PHPMailer access the google account as if it is an application or device. Once that is settled, the password will need to be included in the SMTP connection parameters. The body of the emails are done in the simplest way as most notifications sent from this system are for user authentication and verification. Stylized body messages are more appropriate for email advertising, which is not implemented in this system.

Another important component to this system is task scheduling. This system is designed to delete encrypted files within 14 days and session links in 2 days. The most typical way is to run a Cron job, however since this system is developed using Windows 10, Windows Task Scheduler is utilized to run tasks automatically at a designated time and frequency. The tasks needed to be done are to query the database for any encrypted files and sharing sessions that has a time remaining of 0 minutes. The ID of those files will be selected and be used to run another query to delete the files. The file path will also be queried as it is important to delete stored encrypted files as well. Sharing sessions can just be deleted by retrieving the share ID. The execution lines of these tasks are written in a bat file and are scheduled to run once per day, at 9.00AM.

CHAPTER 8

PRELIMINARY TESTING

8.1 Introduction

This section elaborates on the testing phase done during the development of this project. The testing phase is done to identify any errors that exists in each component before being deployed for user to use.

8.2 Unit Testing

This section shows the unit testing done towards the functions or units currently functional in the system. There are three units that are going to be tested, which are registration, login and updating user details. Each testing for each unit will consists of positive and negative test cases.

8.2.1 User Registration and Authentication

Table 8.1: Positive Test Case for Registration Component

Test Scenario ID	Registration_01	**Test Case ID**	Registration-1A
Test Case Description	Registration-Positive Test Case	**Test Priority**	High
Pre-requisite	NA	**Post-requisite**	A valid user account

Test Execution Steps:

1. Launch the application
2. Navigate to the registration page
3. Register account
4. Check for email verification in email inbox

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to registration page	Click 'Sign up' button	The registration form	The registration form	Edge, Chrome	Pass	Launch Successful
3	Register account	i. Username:	Confirmation page on	Confirmation page on	Edge, Chrome	Pass	Registration complete

		CrispyDonut ii. Email: crispy.cookie0710@gmail.com iii. Password: crispyCookie_0710 iv. Confirm Password: crispyCookie_0710 v. Re-Captcha V2: Attempted and passed	successful registration	successful registration			
4	Check email inbox	Open Gmail of the email submitted for registration	Email verification sent to user's inbox	Email verification sent to user's inbox	Edge, Chrome	Pass	Email verification successfully sent to user

Table 8.2: Negative Test Case for Registration Component

Test Scenario ID	Registration_02	**Test Case ID**	Registration-1B
Test Case Description	Registration-Negative Test Case	**Test Priority**	High
Pre-requisite	NA	**Post-requisite**	NA

Test Execution Steps:

1. Launch the application
2. Navigate to the registration page
3. Register account with already registered user details

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to registration page	Click 'Sign up' button	The registration form	The registration form	Edge, Chrome	Pass	Launch Successful
3	Register account with existing username and email	i. Username: CrispyDonut ii. Email:	Error on registration form: "Username already exists"	Error on registration form: "Username already exists"	Edge, Chrome	Pass	Error is displayed and user details are not stored in database

		crispy.cookie0710@gmail.com iii. Password: crispyCookie_0710 iv. Confirm Password: crispyCookie_0710 v. Re-Captcha V2: Attempted and passed	"Email already exists" Upon form submission: "Your account already exists" alert	"Email already exists" Upon form submission: "Your account already exists" alert			
4	Register account with existing email	i. Username: Susan ii. Email: crispy.cookie0710@gmail.com iii. Password: crispyCookie_0710 iv. Confirm Password: crispyCookie_0710 v. Re-Captcha V2:	Error on registration form: "Email already exists." Upon form submission: "Your account already exists" alert	Error on registration form: "Email already exists." Upon form submission: "Your account already exists" alert	Edge, Chrome	Pass	Error is displayed and user details are not stored in database

		Attempted and passed					
5	Register account with no details	Username, Email, Password, Confirm Password, Re-Captcha: empty	A HTML tooltip with alert "Please fill out this field"	A HTML tooltip with alert "Please fill out this field"	Edge, Chrome	Pass	Error is displayed
6	Register account with no email	i. Username: Susan ii. Email: empty iii. Password: crispyCookie_0710 iv. Confirm Password: crispyCookie_0710 v. Re-Captcha V2: Attempted and passed	A HTML tooltip with alert "Please fill out this field"	A HTML tooltip with alert "Please fill out this field"	Edge, Chrome	Pass	Error is displayed
7	Register account with valid details	i. Username: Susan ii. Email:	Upon submission: "Recaptcha not	"Recaptcha not successful"	Edge, Chrome	Pass	Error is displayed and user cannot register

	but expired re-captcha	crispy.cookie0710@gmail.com iii. Password: crispyCookie_0710 iv. Confirm Password: crispyCookie_0710 v. Re-Captcha V2: Expired	successful ” and navigate user to sign up page	and navigate user to sign up page			

Table 8.3: Positive Test Case for Login Component

Test Scenario ID	Login_01	**Test Case ID**	Login-1A
Test Case Description	Registration-Positive Test Case	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	User Account Page

Test Execution Steps:

1. Launch the application
2. Navigate to the login page
3. Login into account

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to login page	Click 'Login up' button	The login form	The login form	Edge, Chrome	Pass	Launch Successful
3	Login into account	i. Email: crispy.cookie0710@gmail.com ii. Password: crispyCookie_0710	User Homepage	User Homepage	Edge, Chrome	Pass	Login successful

		iii. Re-Captcha V2: Attempted and Passed					

Table 8.4: Negative Test Case for Login Component

Test Scenario ID	Login_02	**Test Case ID**	Login-1B
Test Case Description	Registration-Negative Test Case	**Test Priority**	High
Pre-requisite	NA	**Post-requisite**	NA

Test Execution Steps:
1. Launch the application
2. Navigate to the login page
3. Login into account

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to login page	Click 'Login up' button	The login form	The login form	Edge, Chrome	Pass	Launch Successful
3	Login into account with valid login credentials but expired	i. Email: crispy.cookie0710@gmail.com ii. Password: crispyCookie_0710	"Recaptcha not successful. Redirecting you to login page. " alert and redirects to login page	"Recaptcha not successful. Redirecting you to login page. " alert and	Edge, Chrome	Pass	Login unsuccessful and redirects user back to login page

	re-captcha	iii. Re-Captcha V2: Expired		redirects to login page			
4	Login into account with empty inputs	Email, Password: empty, Re-captcha: Attempted and passed	"Please fill out this field" alert on email field	"Please fill out this field" alert on email field	Edge, Chrome	Pass	Login is not submitted, and error is displayed

8.2.2 User Account Management

Table 8.5: Positive Test Case for Updating User Details

Test Scenario ID	UpdateDetails_01	**Test Case ID**	UpdateDetails -1A
Test Case Description	Update Details - Positive Test Case	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	User details updated

Test Execution Steps:

1. Launch the application
2. Navigate to the login page
3. Login into account
4. Navigate to user settings page
5. Update user details

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to login page	Click 'Login up' button	The login form	The login form	Edge, Chrome	Pass	Launch Successful
3	Login into account	i. Email: crispy.cookie0710@gmail.com ii. Password:	User Homepage	User Homepage	Edge, Chrome	Pass	Login successful

		crispyCookie_0710 iii. Re-Captcha V2: Attempted and Passed					
4	Navigate to settings tab	Click 'Settings' link	Settings Page	Settings Page	Edge, Chrome	Pass	Link navigation successful
5	Navigate to update username form page	Click 'Username' link	'Update Username' form	'Update Username' form	Edge, Chrome	Pass	Link navigation successful
6	Change username	New: 'Seasoned Donut' Password: crispyCookie_0710	'Username update is successful!' alert and username is changed to new username	'Username update is successful' alert and username: 'Seasoned Donut'	Edge, Chrome	Pass	Username update feature is successful
7	Change email	New: deerhunnie90@gmail.com Password:	'Email update is successful! Please check your	Expected alert is fired. The email	Edge, Chrome	Pass	Email is updated successfully

		crispyCookie_0710	inbox to verify your new email.' alert and redirect to change email page. User's email verification status is reset to 'N'. A new email verification link is sent to user's new email.	verification status has been reset to "N" and an email verification landed in the new email's inbox.			
8	Change password	New Password: crispyDoughnuts_0713 Current Password: crispyCookie_0710 OTP: Correct OTP	'OTP passed! alert fired, and user is logged out	'OTP passed!'	Edge, Chrome	Success	User is logged out after changing password. Password is changed correctly.
11	Secret Phrase	Secret Phrase:	A success alert 'Your secret	A success alert	Edge, Chrome	Success	Phrase is successfully stored,

		Asam Pedas Password: crispyDoughnuts_0713 OTP: correct OTP	phrase exists' will show up with a form to update into new secret phrase	'Your secret phrase exists' will show up with a form to update into new secret phrase			and alert is shown as intended.
10	Deactivate account	Password: crispyDoughnots_0713 OTP: Correct OTP	'Your account has been deleted successfully. Thank you!' alert shows up	'Your account has been deleted successfully. Thank you!' alert shows up	Edge, Chrome	Success	User is logged out.

Table 8.6: Negative Test Case for Updating User Details

Test Scenario ID	UpdateDetails_02	**Test Case ID**	UpdateDetails_1B
Test Case Description	Update Details - Negative Test Case	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	User details updated

Test Execution Steps:

1. Launch the application
2. Navigate to the login page
3. Login into account
4. Navigate to user settings page
5. Update user details

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Launch application	localhost/fyp-II/index.php	The homepage of the application	The homepage of the application	Edge, Chrome	Pass	Launch successful
2	Navigate to login page	Click 'Login up' button	The login form	The login form	Edge, Chrome	Pass	Launch Successful
3	Login into account	i. Email: crispy.cookie0710@gmail.com ii. Password:	User Homepage	User Homepage	Edge, Chrome	Pass	Login successful

		crispyCookie_0710 iii. Re-Captcha V2: Attempted and Passed					
4	Navigate to settings tab	Click 'Settings' link	Settings Page	Settings Page	Edge, Chrome	Pass	Link navigation successful
5	Navigate to update username form page	Click 'Username' link	'Update Username' form	'Update Username' form	Edge, Chrome	Pass	Link navigation successful
6	Change username with empty fields	New, Password: Empty	'Please fill out this field' alert on input field for new username	'Please fill out this field' alert on input field for new username	Edge, Chrome	Pass	Successful at preventing submission of empty string
7	Change username with invalid password	New: Glazed Donut, Password: crispCooks_0710	'Username update is not successful' alert	'Username update is not successful' alert	Edge, Chrome	Pass	Successful at preventing submission with

							invalid password
8	Change username with empty password	New, Password: Empty	'Please fill out this field' alert on input field for new username	'Please fill out this field' alert on input field for new username	Edge, Chrome	Pass	Successful at preventing submission of empty string
9	Change email with empty field	New, Password: Empty	'Please fill out this field' alert on input field for new username	'Please fill out this field' alert on input field for new username	Edge, Chrome	Pass	Successful at preventing submission of empty string
10	Change email with invalid password	New: crispy.cookie0710@gmail.com, Password: crispCooks_0710	'Email update is not successful. Your password is wrong' alert	'Email update is not successful. Your password is wrong' alert	Edge, Chrome	Pass	Successful at preventing submission with invalid password

11	Change email with empty password	New: crispy.cookie0710@gmail.com, Password: Empty	'Please fill out this field' alert on input field for new email	'Please fill out this field' alert on input field for new email	Edge, Chrome	Pass	Successful at preventing submission of empty string
12	Change password with empty field	New, Password, OTP: Empty	'Please fill out this field' alert on input field for new password	'Please fill out this field' alert on input field for new password	Edge, Chrome	Pass	Successful at preventing submission of empty string
13	Change password with invalid password	New Password: crispyDoughnut_0713, Password: crispCooks_0710, OTP: Passed	'Current password is incorrect' alert	'Current password is incorrect' alert	Edge, Chrome	Pass	Successful at preventing submission with invalid password
14	Change email with empty	New: crispyDoughnut_0713, Password:	'Please fill out this field' alert on input field for	'Please fill out this field' alert on	Edge, Chrome	Pass	Successful at preventing submissio

	passwor d	Empty, OTP: Pass	new password	input field for new passwor d			n of empty string
15	Change email with invalid OTP	New: crispyDoug hnut_0713 , Password: crispyCooki e_0713, OTP: invalid	'OTP is wrong' alert	'OTP is wrong' alert	Edge, Chrom e	Pass	Successfu l at preventin g update with invalid OTP
16	Change email with empty OTP	New: crispyDoug hnut_0713 , Password: crispyCooki e_0713, OTP: empty	'Please fill out this field' alert on input field for new password	'Please fill out this field' alert on input field for new passwor d	Edge, Chrom e	Pass	Successfu l at preventin g update with empty OTP
17	Change passphr ase with empty passwor d	New: crispyDoug hnut_0713 , Password: Empty, OTP: Pass	'Please fill out this field' alert on input field for new password	'Please fill out this field' alert on input field for new passwor d	Edge, Chrom e	Pass	Successfu l at preventin g submissio n of empty string

18	Change passphrase with invalid password	New: barbeque, Password: boobaboo13, OTP: Pass	'Password does not match! Try again.'	'Password does not match! Try again.'	Edge, Chrome	Pass	Successful at preventing update
19	Change passphrase with invalid OTP	New: barbequw, Password: crispyCookie_0713, OTP: invalid	'OTP is wrong' alert	'OTP is wrong' alert	Edge, Chrome	Pass	Successful at preventing update with invalid OTP
20	Change passphrase with empty OTP	New: barbeque, Password: crispyCookie_0713, OTP: empty	'Please fill out this field' alert on input field for new password	'Please fill out this field' alert on input field for new password	Edge, Chrome	Pass	Successful at preventing update with empty OTP
21	Deactivate with invalid password	Password: booba13 OTP: match	'Password does not match! Try again.'	'Password does not match! Try again.'	Edge, Chrome	Pass	Successful at preventing update
22	Deactivate with empty OTP		'Please fill out this field' alert on input	'Please fill out this field'	Edge, Chrome	Pass	*Successful at preventing update*

			field for new password	alert on input field for new passwor d			*with empty OTP*

Table 8.7: Positive Test Case for Reset Password

Test Scenario ID	ResetPwd_01	**Test Case ID**	ResetPwd _1A
Test Case Description	ResetPwd - Positive Test Case	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	Successful login

Test Execution Steps:

1. Navigate to login page
2. Click reset password link at login page
3. Submit email and check email for reset password email
4. Click on the link provided in the email
5. Submit email and new password
6. Login with new password

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to login page	Click 'Login' button on landing page	Login page	Login page	Edge, Chrome	Pass	Navigation successful
2	Clink reset password link	Click 'Forgot password?' link	Reset Password page	Reset Password page	Edge, Chrome	Pass	Navigation Successful
3	Submit email	Email: Deerhunnie90@gmail.com, reCAPTCHA: attempted and passed	'Check email for password reset!' alert and email arrive in inbox	Check email for password reset!' alert present and email	Edge, Chrome	Pass	Email is sent successfully with a functional link

				arrived in inbox			
4	Click on the link provided in email	Click the link provided	'hash in db and hash received are a match!' alert and navigation to reset password form	'hash in db and hash received are a match!' alert pops up and is navigated to reset password form	Edge, Chrome	Pass	Link works as expected
5	Submit email and new password	Email: deerhunnie90@gmail.com, New password: chicken$chop!sedap987	'Password has been reset successfully' alert and navigation to login page	'Password has been reset successfully' alert shows up and is navigated to login page	Edge, Chrome	Pass	Form submission successful
6	Login with new password	Email: Deerhunnie90@gmail.com Password:	User homepage	User homepage shows up	Edge, Chrome	Pass	Password reset successful

		chicken$ch op!sedap98 7					

Table 8.8: Negative Test Case for Reset Password

Test Scenario ID	ResetPwd_02	**Test Case ID**	ResetPwd _1B
Test Case Description	ResetPwd - Negative Test Case	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	Unsuccessful login

Test Execution Steps:

1. Navigate to login page
2. Click reset password link at login page
3. Submit email and check email for reset password email
4. Click on the link provided in the email
5. Submit email and new password
6. Login with new password

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to login page	Click 'Login' button on landing page	Login page	Login page	Edge, Chrome	Pass	Navigation successful
2	Clink reset password link	Click 'Forgot password?' link	Reset Password page	Reset Password page	Edge, Chrome	Pass	Navigation Successful
3	Submit email	Email: Deerhunnie90@gmail.com, reCAPTCHA: attempted and passed	'Check email for password reset!' alert and email arrive in inbox	Check email for password reset!' alert present and email	Edge, Chrome	Pass	Email is sent successfully with a broken link

				arrived in inbox			
4	Click on the link provided in email	Click the link provided	A notice from mysql	**Notice**: Trying to access array offset on value of type null in **D:\xampp\htdocs\fyp-II\reset_pwd_action.php** on line **44**	Edge, Chrome	Pass	Link works as expected
6	Login with old password	Email: Deerhunnie90@gmail.com Password: chicken$chop!sedap987	User homepage	User homepage shows up	Edge, Chrome	Pass	No reset password is done

8.2.3 Email notification

Table 8.9: Positive Test Case for Email Verification

Test Scenario ID	EmailVerify_01	**Test Case ID**	EmailVerify _1A
Test Case Description	Email Verification - Positive Test Case	**Test Priority**	High
Pre-requisite	A valid email	**Post-requisite**	Email verification success

Test Execution Steps:

1. Register a new acc
2. Check for email verification in email inbox
3. Click the verification link

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Register new account	Username: Zelda6754 Email: crispy.cookie0710@gmail.com password: pedro$zelda_876	Email verification sent from the system is present	Email verification sent from the system is present	Edge, Chrome	Pass	Launch successful
2	Check email inbox for email verification	Open Google Mail	Email verification link is present	Email verification link is present	Edge, Chrome	Pass	Email sent to user successfully

3	Click the verification link	Click the link provided in the email	'Successfully verified email!' alert, with confirmation page shown and email status is updated on database (N -> Y)	'Successfully verified email!' alert, with confirmation page shown and email status is updated on database (N -> Y)	Edge, Chrome	Pass	Registration complete

Table 8.10: Negative Test Case for Email Verification

Test Scenario ID	EmailVerify_02	**Test Case ID**	EmailVerify _1B
Test Case Description	Email Verification - Negative Test Case	**Test Priority**	High
Pre-requisite	A valid email and a broken verification link	**Post-requisite**	NA

Test Execution Steps:

1. Register a new acc
2. Check for email verification in email inbox
3. Click the verification link

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Register new account	Username: Zelda6754 Email: crispy.cookie0710@gmail.com password: pedro$zelda_876	Email verification sent from the system is present	Email verification sent from the system is present	Edge, Chrome	Pass	Launch successful
2	Check email inbox for email verification	Open Google Mail	Email verification link is present	Email verification link is present	Edge, Chrome	Pass	Email sent to user successfully
3	Click the broken	Click the link	'ERR: Your email	'ERR: Your	Edge, Chrome	Pass	Email is not

	verificatio n link	provided in the email	verification is either completed or something is wrong...?' echo from verification module and email is still unverified	email verificatio n is either complete d or somethin g is wrong...? ' echo from verificatio n module and email is still unverifie d			verified, and verification process is halted

8.2.4 Web-based Encryption Tool

Table 8.11: Positive Test Case for Web Encryption

Test Scenario ID	EncryptTool_01	**Test Case ID**	EncryptTool _1A
Test Case Description	EncryptTool - Positive Test Case	**Test Priority**	High
Pre-requisite	Successful Login	**Post-requisite**	Encrypted file recorded in database

Test Execution Steps:

1. Log in into account
2. Navigate to 'Encrypt' page
3. Submit a document file and passphrase
4. Navigate to 'Manage Files' page

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Login into account	wwannorazlin@gmail.com nyaonyao_13	User homepage	User homepage	Edge, Chrome	Pass	Login successful
2	Navigate to 'Encrypt' page	Click on 'Encrypt' button	Encryption page	Encryption Page	Edge, Chrome	Pass	Navigation Successful
3	Submit a document file and	Document file: SUS.pdf Passphrase:	Success alert 'Check the 'Manage Files' section to	Success alert 'Check the 'Manage Files'	Edge, Chrome	Pass	Encryption is successful

	passphrase	Albedo_nim07 Click 'Encrypt File' button	see your encrypted file! 😊 `	section to see your encrypted file! 😊 `			
4	Navigate to 'Manage Files'	Click 'Manage Files' button	The SUS document present with 14 days left	The SUS document present with 14 days left	Edge, Chrome	Pass	Query successful

Table 8.12: Positive Test Case for Web Encryption

Test Scenario ID	EncryptTool_02	**Test Case ID**	EncryptTool _1B
Test Case Description	EncryptTool - Positive Test Case	**Test Priority**	High
Pre-requisite	Successful Login	**Post-requisite**	Encrypted file recorded in database

Test Execution Steps:

1. Log in into account
2. Navigate to 'Encrypt' page
3. Submit a document file and no passphrase
4. Navigate to 'Manage Files'

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Login into account	wwannorazlin@gmail.com nyaonyao_13	User homepage	User homepage	Edge, Chrome	Pass	Login successful
2	Navigate to 'Encrypt' page	Click on 'Encrypt' button	Encryption page	Encryption Page	Edge, Chrome	Pass	Navigation Successful
3	Submit a document file and passphrase	Document file: BMC.pptx Passphrase: Auto-generate	Success alert 'Check the 'Manage Files' section to see your	Success alert 'Check the 'Manage Files' section to	Edge, Chrome	Pass	Encryption is successful

		Click 'Encrypt File' button	encrypted file! 😊 '	see your encrypted file! 😊 '			
4	Navigate to 'Manage Files'	Click 'Manage Files' button	The BMC.pptx document present with 14 days left	The BMC.pptx document present with 14 days left	Edge, Chrome	Pass	Query successful

Table 8.13: Negative Test Case for Web Encryption

Test Scenario ID	EncryptTool_03	**Test Case ID**	EncryptTool _2A
Test Case Description	EncryptTool - Negative Test Case	**Test Priority**	High
Pre-requisite	Successful Login	**Post-requisite**	Error message

Test Execution Steps:

1. Log in into account
2. Navigate to 'Encrypt' page
3. Submit a non-document file and no passphrase
4. Navigate to 'Manage Files'

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Login into account	wwannorazlin@gmail.com nyaonyao_13	User homepage	User homepage	Edge, Chrome	Pass	Login successful
2	Navigate to 'Encrypt' page	Click on 'Encrypt' button	Encryption page	Encryption Page	Edge, Chrome	Pass	Navigation Successful
3	Submit a non-document file and passphrase	Document file: Hand Sanitizer 2.png Passphrase: Auto-generate	Danger alert: "Invalid file type submitted! NOT a document file >:'("	Danger alert: "Invalid file type submitted! NOT a document file >:'("	Edge, Chrome	Pass	Encryption is unsuccessful

		Click 'Encrypt File' button					
4	Navigate to 'Manage Files'	Click 'Manage Files' button	File does not exists	File does not exists	Edge, Chrome	Pass	Query was was not executed as planned

Table 8.14: Negative Test Case for Web Encryption

Test Scenario ID	EncryptTool_04	**Test Case ID**	EncryptTool _2B
Test Case Description	EncryptTool - Negative Test Case	**Test Priority**	High
Pre-requisite	Successful Login	**Post-requisite**	HTML input validation message

Test Execution Steps:

1. Log in into account
2. Navigate to 'Encrypt' page
3. Submit no file and auto-generate passphrase
4. Navigate to 'Manage Files'

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Login into account	wwannorazlin@gmail.com nyaonyao_13	User homepage	User homepage	Edge, Chrome	Pass	Login successful
2	Navigate to 'Encrypt' page	Click on 'Encrypt' button	Encryption page	Encryption Page	Edge, Chrome	Pass	Navigation Successful
3	Submit no file and auto-generate passphrase	Document file: - Passphrase: Auto-generate	HTML input validation message shows up	HTML input validation message shows up	Edge, Chrome	Pass	Encryption is unsuccessful

		Click 'Encrypt File' button					
4	Navigate to 'Manage Files'	Click 'Manage Files' button	File does not exist	File does not exist	Edge, Chrome	Pass	Query was not executed as planned

Table 8.15: Negative Test Case for Web Decryption

Test Scenario ID	DecryptTool_05	**Test Case ID**	DecryptTool _2C
Test Case Description	DecryptTool - Negative Test Case	**Test Priority**	High
Pre-requisite	Encryption file exists	**Post-requisite**	Download Prompt of decrypted file

Test Execution Steps:

1. Navigate to 'Decrypt page'
2. Decrypt file with valid key

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to 'Decrypt page'	Click on 'Decrypt' button	Decrypt page	Decrypt page	Edge, Chrome	Pass	Navigation successful
2	Decrypt file	Click on 'DecryptFile' button under ENC-14087084 Key: nyaonyao_13	Download prompt of decrypted file	Download prompt of decrypted file	Edge, Chrome	Pass	Decryption Successful

Table 8.16: Negative Test Case for Web Decryption

Test Scenario ID	DecryptTool_06	**Test Case ID**	DecryptTool _2D
Test Case Description	DecryptTool - Negative Test Case	**Test Priority**	High
Pre-requisite	Encryption file exists	**Post-requisite**	Download Prompt of decrypted file

Test Execution Steps:

1. Navigate to 'Decrypt page'
2. Decrypt file with invalid key

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to 'Decrypt page'	Click on 'Decrypt' button	Decrypt page	Decrypt page	Edge, Chrome	Pass	Navigation successful
2	Decrypt file	Click on 'DecryptFile' button under ENC-14087084 Key: aaaaaa23	A danger alert: An error occurred! : Invalid user password!	A danger alert: An error occurred! : Invalid user password!	Edge, Chrome	Pass	Decryption unsuccessfu

8.2.5 Contact List and Access Policy

Table 8.17: Positive Test Case for Add Friend

Test Scenario ID	AddFriend_01	**Test Case ID**	AddFriend _1A
Test Case Description	DecryptTool - Positive Test Case	**Test Priority**	High
Pre-requisite	Valid account	**Post-requisite**	Show friend request button

Test Execution Steps:

1. Search DeerDear in search bar
2. Add friend

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Search DeerDear in search bar	DeerDear	DeerDear mini-profile page	DeerDear mini-profile page	Edge, Chrome	Pass	Search successful
2	Add friend	Click on 'Add Friend'	Friend request sent	Friend request sent	Edge, Chrome	Pass	Friend request successful

Table 8.18: Negative Test Case for Add Friend

Test Scenario ID	AddFriend_01	**Test Case ID**	AddFriend _1A
Test Case Description	DecryptTool - Positive Test Case	**Test Priority**	High
Pre-requisite	Valid account	**Post-requisite**	No result

Test Execution Steps: 1. Search non-registered user in search bar							
No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Search Zul in search bar	Zul	Nothing	Nothing	Edge, Chrome	Pass	Search successful

Table 8.19: Positive Test Case for Accept Friend

Test Scenario ID	AcceptFriend_02	**Test Case ID**	AcceptFriend _1B
Test Case Description	DecryptTool - Positive Test Case	**Test Priority**	High
Pre-requisite	Valid account	**Post-requisite**	Friend status updated

Test Execution Steps: 1. Navigate to 'Friend Requests' page 2. Accept friend request							
No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to 'Friend Requests' page	Click 'Friend Requests' button	Friend Request	Friend Request	Edge, Chrome	Pass	Navigation successful
2	Accept friend request	Click 'Accept Friend Request'	Button becomes 'Friends'	Button becomes 'Friends'	Edge, Chrome	Pass	Navigation successful

8.2.6 File Sharing Management

Table 8.20: Positive Test Case for Share File

Test Scenario ID	ShareFile_01	**Test Case ID**	ShareFile _1A
Test Case Description	ShareFile - Positive Test Case	**Test Priority**	High
Pre-requisite	An encrypted file and a friend	**Post-requisite**	Session link exists

Test Execution Steps:

1. Navigate to 'Manage Files' page
2. Click 'Share File' button under any encrypted file
3. Selecta a friend and click 'Confirm'
4. Set Access Password and click 'Submit'
5. Navigate to 'Shared Files' and check if session is made

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to 'Manage Files' page	Click 'Manage Files' button	'Manage Files' page	'Manage Files' page	Edge, Chrome	Pass	Navigation successful
2	Click 'Share File' button	Click 'Share File' button	A modal for selecting contacts appears	A modal for selecting contacts appears	Edge, Chrome	Pass	Modal is responsive
3	Select a friend and click 'Confirm'	Tick the checkbox under DeerDear	Checkbox is ticked with a flower, and	Checkbox is ticked with a flower,	Edge, Chrome	Pass	Jquery is responsive and works

			'Confirm' button is available	and 'Confirm' button is available			as intended
4	Set Access Password and click 'Submit'	Password: Milktea_43	Modal showing confirmation message shows up	Modal showing confirmation message shows up	Edge, Chrome	Pass	Jquery is responsive and Ajax works as intended
5	Navigate to 'Shared Files' and check if session is made	Click 'Shared Files' button	Share session available	Share session available	Edge, Chrome	Pass	Query successful

Table 8.21: Negative Test Case for Share File

Test Scenario ID	ShareFile_02	**Test Case ID**	ShareFile _1B
Test Case Description	ShareFile - Negative Test Case	**Test Priority**	High
Pre-requisite	An encrypted file but no passphrase	**Post-requisite**	HTML input validation message

Test Execution Steps:

1. Navigate to 'Manage Files' page
2. Click 'Share File' button under any encrypted file
3. Selecta a friend and click 'Confirm'
4. Click 'Submit' and leave password empty

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to 'Manage Files' page	Click 'Manage Files' button	'Manage Files' page	'Manage Files' page	Edge, Chrome	Pass	Navigation successful
2	Click 'Share File' button	Click 'Share File' button	A modal for selecting contacts appears	A modal for selecting contacts appears	Edge, Chrome	Pass	Modal is responsive
3	Select a friend and click 'Confirm'	Tick the checkbox under DeerDear	Checkbox is ticked with a flower, and 'Confirm'	Checkbox is ticked with a flower, and 'Confirm'	Edge, Chrome	Pass	Jquery is responsive and works as intended

			button is available	button is available			
4	Set Access Password and click 'Submit'	Passwor d: -	HTML input validation shows up	HTML input validatio n shows up	Edge, Chrome	Pass	Input validation worked as intended
5	Navigate to 'Shared Files' and check if session is made	Click 'Shared Files' button	No session created	No session created	Edge, Chrome	Pass	Query halted successfully

8.3 Integration Testing

This section shows the integration testing done towards the modules that are implemented in this system. This is to ensure that each module executes the expected tasks and communicate with each other smoothly with appropriate outputs and error declarations.

8.3.1 User Account Registration and Verification

Table 8.22: Integration testing for user account creation

Test Scenario ID	UserAccount_01	**Test Case ID**	UserAccount _1A
Test Case Description	User account registration, verification, and login test	**Test Priority**	High
Pre-requisite	A valid email address	**Post-requisite**	A valid account

Test Execution Steps: 1. Navigate to index page 2. Navigate to Sign Up Page 3. Fill-in and submit registration form 4. Verify email 5. Login into account 6. Check email address status in settings							
No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**

1	Navigate to index page	http://localhost/fyp-II/index.php	The index page	The index page	Edge, Chrome	Pass	Navigation successful
2	Navigate to sign up page	Click 'Sing up' button	Registration form	Registration form	Edge, Chrome	Pass	Navigation successful
3	Fill-in and submit registration form	Username: BubbleTea Email: crispy.cookie0710@gmail.com Password: pedro$zelda_876 ReCAPTCHA: Passed	The input elements have green outlines and submission is accepted. An email was received.	The input elements have green outlines and submission is accepted. An email was received.	Edge, Chrome	Pass	Registration successful
4	Verify Email	Click the link provided: http://localhost/fyp-II/verify_email.php?verify=7d864df24a0b0d25f4bbb6e4466ac9a6&id=202034542493&type=1	'Successfully verified email' alert and directs user to confirmation page	'Successfully verified email' alert and directs user to confirmation page	Edge, Chrome	Pass	Email verification works as intended

5	Login into account	Email: crispy.cookie0710@gmail.com Password: pedro$zelda_876	Login successful and user homepage appears	Login successful and user homepage appears	Edge, Chrome	Pass	Login successful
6	Check email address status in settings	Go to 'Settings' page and click 'Email'	'You have verified your email'	'You have verified your email'	Edge, Chrome	Pass	Email verification successful and is displayed

8.3.2 User Account Management

Table 8.23: Integration testing for user update process

Test Scenario ID	UserManage_01	**Test Case ID**	UserManage _1A
Test Case Description	The integration testing for user account updates and deactivation	**Test Priority**	High
Pre-requisite	A valid user account	**Post-requisite**	Account is deleted

Test Execution Steps:

1. Navigate to Profile Page
2. Change profile picture and biography
3. Navigate to Settings page
4. Change username
5. Change email
6. Change password
7. Update secret phrase
8. Deactivate account

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Navigate to Profile Page	Click 'Profile' button	The Profile page	The Profile page	Edge, Chrome	Pass	Navigation successful
2	Change profile picture and biography	Upload image file: Venti_RPG.png Biography: 'Nak main genshin'	'Successful update' And 'Successful update of dp'	'Successful update' And 'Successful update of dp'	Edge, Chrome	Pass	Update successful

3	Navigate to Settings page	Click 'Settings' page	Settings page	Settings page	Edge, Chrome	Pass	Navigation successful
4	Change username	New: HazelnutSweety Password: pedro$zelda_876	'Username is updated!' alert	'Username is updated!' alert	Edge, Chrome	Pass	Update successful
5	Change email	Email: bi18110278@student.ums.edu.my Password: pedro$zelda_876	Email verification link is received. Email is tagged as important. Email status has been updated to not verified.	Email verification link is received. Email is tagged as important. Email status has been updated to not verified.	Edge, Chrome	Pass	Update successful
6	Check password	New: Chickencho p!34 Current: pedro$zelda_876 OTP: Valid	OTP is received. Password updated successfully and user is logged out. Successful login using	OTP is received. Password updated successfully and user is logged out.	Edge, Chrome	Pass	Password reset successful

			new password.	Successful login using new password.			
7	Update secret phrase	Secret phrase: Bubble tea Password: Chickencho p!34 OTP: Valid	OTP is received. 'Secret Phrase updated successfully.' Alert shows up. Status has been updated.	OTP is received. 'Secret Phrase updated successfully.' Alert shows up. Status has been updated.	Edge, Chrome	Pass	Update successful
8	Deactivate account	Password: Chickencho p!34 OTP: Valid	'Your account has been deleted successfully!' alert shows up and redirected to the index page	'Your account has been deleted successfully!' alert shows up and redirected to the index page	Edge, Chrome	Pass	Deactivation successful

8.3.3 File Management

Table 8.24: Integration testing for file management process

Test Scenario ID	FileManage_01	**Test Case ID**	FileManage _1A
Test Case Description	Integration testing for file management process	**Test Priority**	High
Pre-requisite	A valid account	**Post-requisite**	Successful encryption and decryption process

Test Execution Steps:

1. Encrypt a file
2. Decrypt the encrypted file
3. Share File
4. Check sharing sessions if exists
5. View the file
6. Delete file

No.	**Action**	**Inputs**	**Expected Output**	**Actual Output**	**Test Browser**	**Test Result**	**Test Comments**
1	Encrypt a file	File: pdrm surat banjir.pdf Passphrase: auto-generate	Page reload, an alert 'Check the 'Manage Files' section to see your encrypted	Page reload, an alert 'Check the 'Manage Files' section to see your	Edge, Chrome	Pass	Encryption successful

			fil! 😊' shows up	encrypted fil! 😊' shows up			
2	Decrypt the encrypted file	Navigate to decrypt page Click 'decrypt file' in the options in the file's row. Key: nyaonyao_13	Encrypted file shows up. A modal prompting key input shows up. Submission successful. A download prompt shows up in browser. File can be viewed.	Encrypted file shows up. A modal prompting key input shows up. Submission successful. A download prompt shows up in browser. File can be viewed.	Edge, Chrome	Pass	Decryption successful
3	Share File	Navigate to 'Manage Files'. Click 'Share File' button.	Modal confirming the file sharing and email notification will show up.	Modal confirming the file sharing and email notification shows	Edge, Chrome	Pass	Sharing file successful

		Select contact to share file: 'DeerDear' Click 'Confirm' button. Set Access Password: Marshmallow_16 Click 'Submit' button	Link is clickable and works.	up. Link is clickable and works.			
4	Check sharing sessions if exists	Navigate to 'Shared Files' page	Sharing session can be seen on page	Sharing session can be seen on page	Edge, Chrome	Pass	Interface works as intended
5	View the file	Access phrase: Marshmallow_16 Click 'Here's my access phrase! >.< ' button	Modal confirming the access phrase matched will show up. File will be prompted for download.	Modal confirming the access phrase matched will show up. Download prompt	Edge, Chrome	Pass	File viewing successful

		Click 'Proceed' button		shows up.			
6	Delete file	Navigate to 'Manage Files'. Click 'Delete files' button. Navigate to the 'Shared Files'.	A modal asking confirmation from user for deletion shows up. Modal shows up to confirm deletion. Page reloads and file is gone. Shared session is gone.	A modal asking confirmation from user for deletion shows up. Modal shows up to confirm deletion. Page reloads and file is gone. Shared session is gone.	Edge, Chrome	Pass	File deletion successful

CHAPTER 9

SYSTEM USABILITY SCALE (SUS) TESTING

9.1 Introduction

This chapter discusses about the analysis of the performance of the system, where System Usability Scale (SUS) testing was done to evaluate the usability of the system and interface designs. User Acceptance Testing was also done to ensure that the key functions work as intended. The results will be presented statistically using graph and pie chart.

9.2 Data Collections

The form was done using Google Form and distributed through a personal messaging platform. The questionnaire consists of the 10 SUS questions, and participants were asked to follow the instructions written in a Google Document sheet that was included in the form prior to answering the questions. This is so that the User Acceptance Testing (UAT) can also be done simultaneously, to ensure that the key functionalities work as intended.

9.3 Questionnaire Findings

Questionnaires were given out to a total of 15 target users, and only 10 users were involved in the SUS testing. The questionnaire was held out for four days, thus it may have been too short of a duration to collect data. Target users were possibly very occupied, thus did not have time to participate.

9.3.1 SUS Score

According to John Brooke (1995), the creator of SUS, each of the assigned response on the Likert scale has its own score, which can be summarized as such in Table 9.1.

Table 9.1: Scoring SUS

Response assigned	Points	Score Deduction/Addition
Strongly Disagree	1	Odd numbered questions need to be deducted 1 point per question.
Disagree	2	
Neutral	3	
Agree	4	5 points must be deducted from each of the even questions points.
Strongly Agree	5	

The easiest way to calculate is to compile all the scores from odd numbered questions and subtract 50 points (5 points multiply 10). Then, collect all scores from even numbered questions, and subtract 250 points (25 points multiply 10) with the scores. The results from both calculations are then added and multiplied with 2.5. Thus, the calculations are as such in Table 9.2.

Table 9.2: SUS score calculations

Odd questions	Q1 3(3)+4(4)+5(3) = 40 - 10	Q3 3+4(5)+5(4) = 43-10	Q5 3(4)+4(2)+5(4) =40-10	Q7 4(6)+4(5) =44 -10	Q9 2(3)+6(4)+2(5) = 40-10
Even questions	Q2 1(5)+4(4) = 50 - 21	Q4 1(6)+2(3)+3(1) = 50 - 15	Q6 1(4)+2(5)+1(3) =50 - 17	Q8 1(6)+2(4) = 50 - 30	Q10 3(1)+5(2)+2(3) = 50 - 19
SUS score	[(30+33 + 30 + 34 + 30) + (29 +35 +33 + 20 + 31)] x 2.5 = (157 + 148) x2.5 = 762.5 762.5 / 10 = 76.25				

The average SUS score is 76.25, which indicates that the website is acceptable and can be further improved by doing minor changes to the design. In addition to the 10 SUS questions, users were also asked to provide any feedback if it is necessary, as this is to see if there are any features experiencing problems. Certain respondents were confused of the auto-generated key feature. They expressed that the feature has problems because the auto-generated key cannot be viewed or cannot be saved. Besides that, a respondent expressed that the sharing file feature is buggy, and this could either be caused by slow internet connection or slow processing from server side. The overall feedback indicates that the website is acceptable, and it is fairly easy to use.

Thank you for participating! Is there is any comments you would like to add?

5 responses

I think the website is really simple and easy to use it.

Cannot view the generated passphrase and mouse pointer is missing when hovering on the buttons that have black background

The view shared file page was buggy for me hhh but overall all is great !

great application :D

Expecting for us to be able to save the passphrase/key

Figure 9.1: The feedback left by respondents

CHAPTER 10

CONCLUSION

10.1 Summary of Project

Project I involved the discussion of six chapters: Introduction, Literature Review, Methodology, System Analysis and Design, Experiment Implementation and Discussion. The first chapter mainly helps in identifying the project structure prior developing the system. The second chapter helped in understanding lightweight encryption algorithms and current existing web encryption tools. Three lightweight encryption algorithms are then experimented to identify the appropriate lightweight encryption algorithm to be implemented into the project, which is discussed in chapter five and six. The third chapter elaborates on the methodology being implemented in the project. The fourth chapter is mainly on the design of the system, which includes the user interface and experience. It also demonstrated how the modules would be implemented.

Project II involves mainly the implementation and testing phase, where the system is developed, then tested during development and post-development. It consists of four chapters which are: Implementation, Preliminary Testing, System Usability Scale (SUS) Testing and Conclusion. Implementation chapter discusses on the development environment that is currently used and the components or units that are functional and working. Preliminary testing chapter shows all the test cases done towards the functions or units within the system. Each of the test cases for each unit consists of two types, which are positive cases and negative cases. Next, SUS testing alongside with UAT was done to measure the usability and performance of the system and its design. The results were analysed statistically and presented using graphs and pie chart.

Overall, the objectives in this project are achieved. Chapter two and five complied with the first objective as it includes reviewing five lightweight encryption algorithms and experimenting three of the selected algorithms to identify the most appropriate lightweight algorithm to be implemented into the system. Chapter four and seven complied with the second objective which involves designing and developing the system to be able to protect document's confidentiality and integrity. Meanwhile the third objective is achieved as chapter eight and nine evaluated the system during development (using unit testing and integration testing) and post-development (using SUS testing and UAT).

10.2 Constraints

There are constraints in several features within the system, which can be discussed according to the module it is being integrated in. The main constraint involves the inability for users to decrypt their stored encrypted files upon password reset. Since the master key used to encrypt user's files are tied to the user's password, it will require for the encrypted files to be decrypted and re-encrypted with an updated master key. This would be a problem if multiple users reset their passwords and there are hundreds or thousands of encrypted files to be decrypted and re-encrypted. Besides that, there are no alternative method for users to decrypt their files if they forgot their own provided key or password. This is related to another constraint where the 'secret phrase' is not yet integrated effectively into the user account management module. There is also a constraint in the decryption tool where there is no option for users to upload their downloaded encrypted files. The system allows users to download their encrypted files, thus there should be included a way for users to upload the encrypted file if needed for decryption, within the 14 days. Another constraint is the slow file viewing feature in file sharing management module.

10.3 Future Works

The system can be further improved in terms of the UI/UX, as the current layout can seem inconsistent, shows unpredictable behavior and can be hard to maintain in the future. The interface design itself can be improved to be more aesthetic and follow the current trends in web design. The features can be improved to provide a much more secure and robust file sharing web-based application. For example, the current process for resetting passwords does not benefit users in terms of decrypting older encrypted documents (as in documents that were encrypted prior password reset) since the master key is tied to user's password. One of the ways to combat this issue is to generate the master key upon registration with a uniquely generated ID and being fed into a key derivation algorithm with its own unique salt. After that, the master key can be encrypted with a key tied to user's password and stored in a separate server directory, secluded from the main database. This way, the password reset would not affect the master key, and thus users can decrypt the files with ease. Besides that, it would be more accessible for users if a mobile version of this system can be developed as users nowadays access to almost everything via their smartphones. By having a mobile version of the system, users can securely encrypt/decrypt and share their files on-the-go, without needing to access their personal computers. File sharing feature can also be redesigned in the mobile version, where it could be done via a private 'chatting room' rather than viewing it on the browser.

REFERENCES

Amazon. 2021. "*What is Cloud File Storage?* " [Online]. Available: https://aws.amazon.com/what-is-cloud-file-storage/. [Accessed: 21-Feb-2021]

Athow, D. 2021. Best cloud storage services of 2021. *Techradar.Pro*. 13 February: 1. https://www.techradar.com/news/the-best-cloud-storage

Appel, M., Pauer, C., & Wiesmaier, A. (2016). *Security aspects and comparison of block ciphers LED and TEA*.

D. Hong, Lee, J., Kim, D., Kwon, D., Ryu, K. H., & Lee, D. .2014. *LEA : A 128-Bit Block Cipher for Fast Encryption on Common Processors*. *1*, 3–27. https://doi.org/10.1007/978-3-319-05149-9

Foo, F. 2011. "*What specifically are wall-clock-time, user-cpu-time, and system-cpu-time in UNIX?"* [Online]. Available: https://stackoverflow.com/questions/7335920/what-specifically-are-wall-clock-time-user-cpu-time-and-system-cpu-time-in-uni

Bottarni, J. 2016. *List of Countries Banned & Restricted from Obtaining SSL Certificates*. Wiyre. 2 March: 1. https://www.wiyre.com/list-of-countries-banned-restricted-from-obtaining-ssl-certificates/

Brooke, John. 1995. SUS: A quick and dirty usability scale. Usability Eval. Ind.. 189. https://www.researchgate.net/publication/228593520_SUS_A_quick_and_dirty_usability_scale

Buchanan, W. J., Li, S., & Asif, R. .2017. Lightweight cryptography methods. *Journal of Cyber Security Technology*, *1*(3–4), 187–201. https://doi.org/10.1080/23742917.2017.1384917

Choi, B., Kim, B., & Park, J. .2018. Low-Cost Hardware Implementation of LEA-128 Encryption using Bit-Serial Technique. *2018 International SoC Design Conference (ISOCC)*, 46–47. https://doi.org/10.1109/ISOCC.2018.8649937

Coppock, M. 2020. The best cloud storage services for 2020. *Yahoo!News*. 17 October: 1. https://malaysia.news.yahoo.com/best-cloud-storage-services-191500344.html

Devglan. 2019. "*Online Tool for AES Encryption and Decryption*." [Online]. Available: https://www.devglan.com/online-tools/aes-encryption-decryption [Accessed: 17-4-2021]

Dropbox. 2021. "*What types of files can I preview in Dropbox?"* [Online]. Available: https://help.dropbox.com/files-folders/sort-preview/file-types-that-preview [Accessed: 21-Feb-2021]

Edgar, T. W., & Manz, D. O. 2017. Science and Cyber Security. In *Research Methods for Cyber Security* (pp. 33–62). Elsevier. https://doi.org/10.1016/b978-0-12-805349-2.00002-9

EncDec. 2020. "*EncDec - Online Encryption Tool*." [Online] Available: https://www.encdec.net/ [Accessed: 17-4-2021]

ENCRYPT.One. 2016. "*Encrypt & Decrypt Text or File Online (AES encryption)*." [Online]. Available: https://encrypt.one/ [Accessed 17-4-2021]

Ertaul, L., & Rajegowda, S. K. .2017. *Performance Analysis of CLEFIA, PICCOLO, TWINE Lightweight Block Ciphers in IoT Environment*.

Fearn, N., & Turner, B. 2020. Best encryption software of 2021: Free, paid and business tools and services. *Techradar*. 30 November: 1. https://www.techradar.com/sg/best/best-encryption-software#4-certainsafe

FreeCrypt.Org. 2019. "*Free Encrypt : The free online file encryption & decryption service*." [Online] Available: https://freecrypt.org/ [Accessed: 17-4-2021]

Google. 2021. "*Files you can store in Google Drive*." [Online]. Available: https://support.google.com/drive/answer/37603 [Accessed: 21-Feb-2021]

Hayes, M., & LLP, C. 2020. Privacy and Security: The Risks of Working from Home. *Lexology*. 29 June: 1. https://www.lexology.com/library/detail.aspx?g=cd509113-26f9-4f4a-855f-3b75a0d3d984

Joe. 2021. *The 4 Best Cloud Storage for Malaysia – Compared and Ranked!* https://www.goodcloudstorage.net/guide/best-cloud-storage-malaysia/

Jun, S. W. 2021. Putrajaya: Cloud service providers to invest up to RM15b under the Digital Economy Blueprint. *Malaymail*. 19 February: 1. https://www.malaymail.com/news/malaysia/2021/02/19/putrajaya-cloud-service-providers-to-invest-up-to-rm15b-under-the-digital-e/1951011

"Listings of WHO's response to COVID-19." 2020. [Online]. Available: https://www.who.int/news/item/29-06-2020-covidtimeline [Accessed: 12-Jan-2021]

Majlis Keselamatan Negara (MKN). 2021. *SOP PKP Pemulihan*. [Online]. Available: https://www.mkn.gov.my/web/ms/sop-pkp-pemulihan/ [Accessed: 12-Jan-2021]

TheStar. 2020. Malaysia to see massive spike in cloud services. 9 Jun: 1. https://www.thestar.com.my/business/business-news/2020/06/09/malaysia-to-see-massive-spike-in-cloud-services#close

Ma, C., Chandy, J., & Shi, Z. 2017. *Algebraic Side-Channel Attack on Twofish*. *2* (May), 32–43.

Maqsood, F., Ahmed, M., Mumtaz, M., & Ali, M. 2017. Cryptography: A Comparative Analysis for Modern Techniques. *International Journal of Advanced Computer Science and Applications*. https://doi.org/10.14569/ijacsa.2017.080659

McAfee. 2019. *How does cloud encryption work?* https://www.mcafee.com/enterprise/en-us/security-awareness/cloud/how-does-cloud-encryption-work.html

Meskauskas, T. 2020. "*CryLocker ransomware removal instructions*." [Online]. Available: https://www.pcrisk.com/removal-guides/10441-cry-ransomware#:~:text=Cry (CryLocker) is a new,%22 is renamed to %22sample. [Accessed: 18-April-2021]

Mell, P., & Grance, T. 2011. The NIST-National Institute of Standards and Technology- Definition of Cloud Computing. *NIST Special Publication 800-145*, 7. Retrieved from https://faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf

Microsoft. 2021. *File types supported for previewing files in OneDrive, SharePoint, and Teams*. https://support.microsoft.com/en-us/office/file-types-supported-for-previewing-files-in-onedrive-sharepoint-and-teams-e054cd0f-8ef2-4ccb-937e-26e37419c5e4

Mishra, S., Mishra, Z., & Acharya, B. (2019). A High Throughput and Speed Architecture of Lightweight Cipher LEA. *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 458–462. https://doi.org/10.1109/ICCCIS48478.2019.8974468

Moriai, S., & Yin, Y. L. .2000. *Cryptanalysis of Twofish*.

News 4, & FOX 11 Digital Team. 2021. WCSD issues message to parents about improper online document storage system incident. 5 February: 1. https://mynews4.com/news/local/wcsd-issues-message-to-parents-about-online-document-storage-system-incident

Nisha, S., & Farik, M. 2017. RSA Public Key Cryptography Algorithm-A Review. *International Journal of Scientific & Technology Research*. Retrieved from https://www.ijstr.org/final-print/july2017/Rsa-Public-Key-Cryptography-Algorithm-A-Review.pdf

Raphael, J. 2017. Welcome to the era of the Google ecosystem. *Computerworld*. 17 October:1. https://www.computerworld.com/article/3233300/google-ecosystem.html

Statista. 2021. *Public cloud services market size 2009-2022*. [Infographic]. Available: https://www.statista.com/statistics/273818/global-revenue-generated-with-cloud-computing-since-2009/ [Accessed: 22-Feb-2021]

Suzaki, T., & Minematsu, K. 2010. *Improving the Generalized Feistel*. 19–39.

Tariq, Q. 2019. Universiti Malaya, JPDP looking into report of massive data breach affecting the university. *TheStar*. 19 October: 1. https://www.thestar.com.my/news/nation/2019/10/19/university-malaya-pdpd-looking-into-report-of-massive-data-breach-affecting-the-university

The Economic Times. 2021. Singapore's Singtel says personal information of 1,29,000 users stolen in data breach. 18 February: 1. https://economictimes.indiatimes.com/news/international/business/singapores-singtel-says-personal-information-of-129000-users-stolen-in-data-breach/articleshow/81085100.cms

TheStar. 2020. Malaysia to see massive spike in cloud services. 9 Jun: 1. https://www.thestar.com.my/business/business-news/2020/06/09/malaysia-to-see-massive-spike-in-cloud-services#close

TheStar. 2021. Enterprises keen to adopt hybrid cloud services. 4 February: 1. https://www.thestar.com.my/business/business-news/2021/02/04/enterprises-keen-to-adopt-hybrid-cloud-services

Tolba, M., & Youssef, A. M. 2016. Generalized MitM attacks on full TWINE. *Information Processing Letters*, *116*(2), 128–135. https://doi.org/10.1016/j.ipl.2015.09.011

Goundar, S. 2012. Chapter 4 - Understanding Cloud Computing. *Cloud Computing*, *May*.

Guez, Y. 2018. "*6 encryption mistakes that lead to data breaches"*. [Online] Available: https://www.crypteron.com/blog/the-real-problem-with-encryption/

Guo, J., Peyrin, T., Poschmann, A., & Robshaw, M. .2011. *The LED Block Cipher*. 326–341.

Hatzivasilis, G., Fysarakis, K., Papaefstathiou, I., & Manifavas, C. (2018). A review of lightweight block ciphers. *Journal of Cryptographic Engineering*, *8*(2), 141–184. https://doi.org/10.1007/s13389-017-0160-y

Hercigonja, Z. 2016. Comparative Analysis of Cryptographic Algorithms. *International Journal of DIGITAL TECHNOLOGY & ECONOMY*, *1*(2), 127–134.

Hosseinzadeh, J., & Bafghi, A. G. 2017. Software Implementation And Evaluation Of Lightweight Symmetric Block Ciphers Of The Energy Perspectives And Memory. *International Journal of Engineering Education /(IJEE)*, *9*(2), 1–6.

Jeevalatha, E., & Mr.S.SenthilMurugan. 2018. *Evolution of AES, Blowfish and Two fish Encryption Algorithm 1*. *9*(4), 115–118.

Johnson, E. 2017. Lost in the cloud: Cloud storage, privacy, and suggestions for protecting users' data. *Stanford Law Review*, *69*(3), 867–909. Retrieved from https://review.law.stanford.edu/wp-content/uploads/sites/3/2017/03/69-Stan-L-Rev-867.pdf

Ortiz, J. J. G., & Compton, K. J. 2016. *A Simple Power Analysis Attack on the Twofish Key Schedule*. 1–23.

O'Sullivan, F. .2020. What Exactly is Zero-Knowledge in The Cloud and How Does it Work? *Cloudwards*. https://www.cloudwards.net/what-exactly-is-zero-knowledge-in-the-cloud-and-how-does-it-work/

Satapathy, A., & Livingston, J. 2016. A Comprehensive Survey on SSL/ TLS and their Vulnerabilities. *International Journal of Computer Applications*, *153*(5), 31–38. https://doi.org/10.5120/ijca2016912063

Schneier, B. 2017. Twofish Power Analysis Attack. https://www.schneier.com/blog/archives/2017/01/twofish_power_a.html

Schneier, B., Kelsey, J., Whiting, D., Wagner, D., & Hall, C. .1998. *Twofish : A 128-Bit Block Cipher*.

Suzaki, T., Minematsu, K., Morioka, S., & Kobayashi, E. 2013. A lightweight block cipher for multiple platforms. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7707 LNCS*, 339–354. https://doi.org/10.1007/978-3-642-35999-6_22

Tiahrt, K. J., & Hamburg, M. .1975. Basic Statistics: A Modern Approach. *Journal of the American Statistical Association*, *70*(351), 732. https://doi.org/10.2307/2285972

Uma, K., Karthik, G., & Prasath, R. V. 2017. A comparative analysis of Symmetric and Asymmetric key cryptography. *Journal of Chemical and Pharmaceutical Sciences*, *10*(1), 324–326. https://doi.org/10.1109/ICECCE.2014.7086640

Valea, E., Da Silva, M., Flottes, M. L., Di Natale, G., & Rouzeyre, B. 2019. Stream vs block ciphers for scan encryption. *Microelectronics Journal*, *86*, 65–76. https://doi.org/10.1016/j.mejo.2019.02.019

VMWare. 2019. "*What is XAMPP?*" [Online]. Available: https://www.apachefriends.org/index.html [Accessed: 19-11-2021]

wellknowngeek. 2019. "*Online Encryption Tools - AES, DES, BlowFish, RSA*." [Online]. Available: http://www.txtwizard.net/crypto [Accessed: 17-4-2021]

Wheeler, D. J., & Needham, R. M. .1997. *The Tiny Encryption Algorithm (TEA) TEA, a Tiny Encryption Algorithm*.

Witoolkollachit, P. 2016. The avalanche effect of various hash functions between encrypted raw images versus non-encrypted images: A comparison study. *Journal of the Thai Medical Informatics Association*, *1*, 69–82.

Yoshikawa, M., Nozaki, Y., & Asahi, K. .2016. *Electromagnetic Analysis Attack for a Lightweight Block Cipher TWINE*. *IEEE/ACES*. https://doi.org/doi:10.1109/ropaces.2016.7465354

APPENDIX A

Feedback to Proposal Review

REWIEW 1

Comments	Reply to Comments
What you mean "Cloud Documents"?	Title has been revised. Thank you for the feedback.
1.3 i. – what is the actual problem here? It is not clear.	I apologise for the lack in proper explanation. Further explanation is provided under Chapter 1 Section 1.3 under the first problem statement.
1.3 ii. – Any proof that these encryption algorithms cannot support large files?	Further explanation is provided under Chapter 1 Section 1.3 Problem Statements.
1.3 iii. – What is the problem?	The problem has been re-phrased for better understanding under Chapter 1 Section 1.3 Problem Statements.
1.4 i. What you mean "storage size" in objective 1?	Storage size refers to circuit size or RAM/ROM sizes. Basically, how much RAM size does the encryption algorithm use. I apologise for the vague wording.
1.4 iii. In which aspect of the performance will be evaluated in objective 3?	The computation speed and storage size will be evaluated.

The function of your web-based application is limited. More modules should be added such as student assignment submission, assignment evaluation module, etc. File uploading/exporting cannot consider a module in the project. It is just a function. For example, in a registration module, upload a photo is just a function in the module. The encryption/decryption module is just one of the modules in your project. You should propose a complete system/web application in this case.	I apologise for the incomplete modules under this section. It was incorrect. This section has been altered and improved in Chapter 1 Introduction Section 1.5 Project Scope in Table 1.1.
What is/are the differences between your web applications with existing encryption applications/systems especially in the cloud environment?	Thank you for the question. This will be further explained in Chapter 4 System Analysis and Design where the features of the project are detailed there.
Is the file be encrypted before "exporting" to the cloud storage? If yes, any different that I encrypt a file using word and upload it to the cloud after this?	Yes, the file is being encrypted before exporting. The web-tool is for people that wants to encrypt any files online using a web browser. Encrypting using Word serves the same purpose as what this project is aimed for (which is uploading an encrypted file to cloud rather than raw file) but using Word means it is limited to only Word documents. This web tool aims to support various types of files and provide an alternative option for users to encrypt their files.
Why you choose these three symmetric algorithms as your options for the web application?	Apologies for not explaining it further. Further explanation is provided in

	Chapter 1 Section 1.5 Project Scope in second paragraph.
The testing phase should include the way you prove that your proposed web application can protect the "confidentiality and integrity of cloud storage" as stated in the objective.	The method for proving confidentiality and integrity is added in Chapter 3 Methodology Section 3.1 Introduction on Phase 3
Target users are UMS lecturers and students. But what is/are the problems that they are facing now and motivate you to propose this project?	The explanation on this matter has been included in Chapter 1 Section 1.5 Project Scope in fourth paragraph.

REVIEW 2

Comments	Reply to Comments
The introduction section is needs to be address Working from home (WFH) culture in professional way.	Thank you for the input. Explanation regarding WFH is further improved under Chapter 1 Section 1.1 Introduction in second paragraph.
Student should specify the inferences from the related work and should specify the significance of this proposal.	Thank you for the observation. The inferences from related work are being explained in Chapter 2 Literature Review. The significance of the proposal is clarified in Section 1.5 Project Scope in fourth paragraph.
The limitation of previous studies should be clearly clarified.	Thank you for the feedback. The limitations of previous studies are clarified under Chapter 2 Section 2.1.6.
English need to improve, lots of grammatical errors should be removed.	Grammatical errors have been checked and corrected. I will try my best to improve my English.
I would like to suggest industry collaborations	Thank you for the suggestion on industry collaborations. Further explanation is provided in Chapter 1 Section 1.5 Project Scope in fourth paragraph.

APPENDIX B

QUESTIONNAIRE ABOUT WEB-BASED ENCRYPTION TOOL

SECTION A: USER EXPERIENCE (PROBLEM/LIMITATIONS)

NO.	QUESTIONS	DETAILS
1	Have you encrypted files before using web-based encryption tools?	*Two choices: Choose one:* *Yes or No. If "Yes", participant is prompted to include the encryption tool.*
2	What type of files do you often encrypt?	*Multiple choices. Select which applies:* *Document files (.doc, .docx, .txt, etc),* *Video files (.mp4, .3gp, .mov, etc),* *Audio files (.mp3, .aac, etc)* *Others*
3	How long do it usually take for you to encrypt your files?	*An Interval Scale will be used.*
4	Do the current web-based encryption tool(s) you are using, allows selection of multiple files to be encrypted?	*Two choices: Choose one:* *Yes or No.*
6	What cloud storage services do you often use?	*Multiple choices. Select which applies:* *Google Drive, OneDrive, Others*
7	Do you find your current process of encrypting your documents to be uploaded into your cloud storage, convenient?	*Two choices: Choose one:* *Yes or No.*
8	What limitations do you think should be addressed most in web-based encryption tools?	*Multiple choices. Select which applies:* *Lack in online file sharing (e.g upload to Google Drive),*

		Often only includes either text encryption or file encryption only and not both at the same website, Slow encryption for multimedia files, No indicator on the strength of passphrases/key provided by user, Others

SECTION B: EXPECTATION/SUGGESTIONS (DETERMINE FUNCTIONALITY)

NO.	**QUESTIONS**	**DETAILS**
1	What features do you expect in a web-based encryption tool?	*Multiple choices. Select which applies: Text encryption/decryption, File encryption/decryption, Key generator, File retrieval feature, File Sharing, Others*
2	What do you expect in terms of encryption/decryption performance?	*Multiple choices. Select which applies: Fast encryption process, Secure encryption, Able to support large files, Others*
3	Do you prefer your encryption key to be auto generated, or self-written?	*Two choices: Choose one: Auto-generated, Self-written*
4	Do you prefer for both options in Question 3 to be included in an encryption tool and allows you to choose which to apply?	*Two choices: Choose one: Yes or No.*
5	If multiple files can be selected to be encrypted, would you prefer: A. To download/receive the encrypted files in a zip file (all files	*Two choices: Choose one: A or B*

	are compiled in an encrypted zip file) B. Encrypt every file and download it separately.	
6	Do you think a retrieval feature for encrypted files is necessary?	*Two choices: Choose one:* *Yes or No.*
7	Do you think there should be a limited time for users to retrieve their encrypted files? (e.g encrypted files are deleted after 14 days to ensure confidentiality)	*Two choices: Choose one:* *Yes or No.*
8	Which one do you prefer for the file retrieval process: A. Making an account to store selected encrypted files and retrieve it later B. Receive file ID via email to query/search and retrieve the file later	*Two choices: Choose one:* *A or B*
9	Do you think there should be a feature to help you retrieve the keys of your encrypted files within a time limit? (e.g: keys are encrypted and stored for 14 days. Users can retrieve the encrypted keys and decrypt using their password account to see the key)	*Two choices: Choose one:* *Yes or No.*
10	Any additional feedbacks/comments on the current existing web-based encryption tools? Or maybe any	*An open answer submission*

	additional features you expect to be included in a web-based encryption tool? Thank you for your feedback!	

APPENDIX C

FEEDBACK TO FYP REPORT PROGRESS

EXAMINER 1

COMMENTS	REPLY TO COMMENTS
1.3 i. – What is the real issue here? It is unclear.	This problem statement is redone under Chapter 1 Section 1.3 Problem Statements
1.3 ii. – Is there any evidence that these encryption algorithms cannot handle large files?	An article has been provided in this problem statement.
1.3 iii. – What is the issue?	Thank you for the reply. It is explained under Chapter 1 Section 1.3 Problem Statement.
In Objective 1, what do you mean by "storage size"?	"Storage size" means ciphertext size.
Your web-based application's functionality is limited. More modules, such as student assignment submission and assignment evaluation, should be added.	Thank you for the feedback. This is explained under Chapter 1 Section 1.5 Project Scope
Your project's encryption/decryption module is just one of many. In this case, you should propose a complete system/web application.	This is explained under Chapter 1 Section 1.5 Project Scope
What are the differences between your web applications and existing file encryption applications/systems, particularly in the cloud?	This is explained under Chapter 1 Section 1.5 Project Scope
Is the file encrypted before it is "exported" to the cloud storage? If so, is there any	This is explained under Chapter 1 Section 1.5 Project Scope

difference between encrypting a file in Word and uploading it to the cloud after that?	

EXAMINER 2

COMMENTS	REPLY TO COMMENTS
Typo Error in the report.	The typo error in the title of the report has been fixed. Thank you for your feedback.
Failed to provide the supporting evidence (facts, research articles) in problem statements.	Supporting evidence is added into the problem statements under Chapter 1 Section 1.3 Problem Statements.
Missing citation in Phase 2 formula.	Phase 2 formula under Chapter 3 Methodology has been cited.
Highlight the methods in Methodology Figure.	The methods in Methodology figure under Chapter 3 has been highlighted.

APPENDIX D

RESULTS FROM EXPERIMENT IMPLEMENTATION

RESULTS FROM 100 TRIALS OF ENCRYPTION

Filetype	LEA					
	128-bit		192-bit		256-bit	
Data	Wall Time (seconds)	CPU Time (seconds)	Wall Time (seconds)	CPU Time (seconds)	Wall Time (seconds)	CPU Time (seconds)
XML	0.07	0.06999	0.066	0.066	0.071	0.0709
TXT	0.0957	0.0958	0.077	0.0766	0.086	0.0857
ODS	0.1123	0.1123	0.112	0.1126	0.137	0.1367
XLSX	0.2467	0.2472	0.257	0.2566	0.268	0.2669
CSV	0.5327	0.5329	0.554	0.5546	0.609	0.6093
PPTX	1.4483	1.448	1.458	1.4587	1.545	1.5451
DOCX	1.863	1.8632	1.974	1.9737	2.117	2.1164
PPT	5.7017	5.7016	5.872	5.872	5.904	5.904
PDF	6.572	6.5714	6.304	6.3045	6.704	6.7041

Filetype	TWOFISH					
	128-bit		192-bit		256-bit	
Data	Wall Time (seconds)	CPU Time (seconds)	Wall Time (seconds)	CPU Time (seconds)	Wall Time (seconds)	CPU Time (seconds)
XML	0.068	0.0679	0.073	0.0737	0.066	0.0661
TXT	0.076	0.0756	0.076	0.0763	0.094	0.0936
ODS	0.105	0.1053	0.094	0.0925	0.153	0.1526
XLSX	0.273	0.2737	0.255	0.2248	0.216	0.2153
CSV	0.675	0.6744	0.584	0.5839	0.492	0.492
PPTX	1.241	1.241	1.243	1.2434	1.913	1.9124
DOCX	1.69	1.6892	1.699	1.6989	1.667	1.667
PPT	4.704	4.7045	4.55	4.5506	4.581	4.5817
PDF	6.143	6.1429	5.213	5.2136	6.926	6.9257

Filetype	XTEA	
	128-bit	
Data	Wall Time (seconds)	CPU Time (seconds)
XML	0.06	0.0602
TXT	0.097	0.0971
ODS	0.12	0.1195
XLSX	0.408	0.4083

CSV	0.831	0.8303
PPTX	2.174	2.1747
DOCX	3.086	3.0869
PPT	7.794	7.7946
PDF	8.818	8.818

RESULTS FROM 100 TRIALS OF DECRYPTION

Filetype	**LEA**					
	128-bit		**192-bit**		**256-bit**	
Data	**Wall Time (seconds)**	**CPU Time (seconds)**	**Wall Time (seconds)**	**CPU Time (seconds)**	**Wall Time (seconds)**	**CPU Time (seconds)**
XML	0.066	0.0658	0.063	0.0633	0.065	0.0653
TXT	0.081	0.0812	0.088	0.0885	0.139	0.1388
ODS	0.107	0.1073	0.095	0.0951	0.093	0.0928
XLSX	0.184	0.1845	0.204	0.2045	0.33	0.3299
CSV	0.484	0.4843	0.401	0.4006	0.433	0.4328
PPTX	0.992	0.9919	0.989	0.9897	1.681	1.6809
DOCX	1.395	1.3958	1.432	1.4322	1.688	1.6882
PPT	3.456	3.4564	3.904	3.904	4.63	4.6301
PDF	4.153	4.1523	3.934	3.934	5.836	5.8365

Filetype	**TWOFISH**					
	128-bit		**192-bit**		**256-bit**	
Data	**Wall Time (seconds)**	**CPU Time (seconds)**	**Wall Time (seconds)**	**CPU Time (seconds)**	**Wall Time (seconds)**	**CPU Time (seconds)**
XML	0.068	0.0681	0.066	0.0656	0.218	0.2186
TXT	0.083	0.0833	0.09	0.0908	0.218	0.2186
ODS	0.1	0.0994	0.093	0.0931	0.089	0.0894
XLSX	0.23	0.2301	0.213	0.2135	0.218	0.21862
CSV	0.445	0.444	0.456	0.4561	0.544	0.5445
PPTX	1.219	1.2194	1.197	1.1971	1.195	1.1935
DOCX	1.647	1.6471	1.61	1.6101	1.64	1.6398
PPT	4.525	4.5254	4.62	4.6194	4.657	4.6565
PDF	5.89	5.8904	5.172	5.1725	5.191	5.19098

Filetype	**XTEA**	
	128-bit	
Data	**Wall Time (seconds)**	**CPU Time (seconds)**

XML	0.086	0.0857
TXT	0.185	0.1855
ODS	0.131	0.1293
XLSX	0.349	0.3491
CSV	0.773	0.7736
PPTX	1.917	1.9168
DOCX	3.018	3.0182
PPT	7.759	7.7589
PDF	8.923	8.9234

CIPHERTEXT SIZE AFTER ENCRYPTION

Filetype	**LEA**					
	128-bit		**128-bit**		**256-bit**	
Data	**Raw File Size (KB)**	**Encrypted File Size (KB)**	**Raw File Size (KB)**	**Encrypted File Size (KB)**	**Raw File Size (KB)**	**Encrypted File Size (KB)**
XML	4.548	4.560	4.548	4.560	4.548	4.560
TXT	17.900	17.904	17.900	17.904	17.900	17.904
ODS	27.655	27.664	27.655	27.664	27.655	27.664
XLSX	130.050	130.064	130.050	130.064	130.050	130.064
CSV	316.971	316.976	316.971	316.976	316.971	316.976
PPTX	898.827	898.832	898.827	898.832	898.827	898.832
DOCX	1285.356	1285.360	1285.356	1285.360	1285.356	1285.360
PPT	3985.920	3895.936	3985.920	3895.936	3985.920	3895.936
PDF	4556.290	4456.304	4556.290	4456.304	4556.290	4456.304

Filetype	**TWOFISH**					
	128-bit		**192-bit**		**256-bit**	
Data	**Raw File Size (KB)**	**Encrypted File Size (KB)**	**Raw File Size (KB)**	**Encrypted File Size (KB)**	**Raw File Size (KB)**	**Encrypted File Size (KB)**
XML	4.548	4.560	4.548	4.560	4.548	4.560
TXT	17.900	17.904	17.900	17.904	17.900	17.904
ODS	27.655	27.664	27.655	27.664	27.655	27.664
XLSX	130.050	130.064	130.050	130.064	130.050	130.064
CSV	316.971	316.976	316.971	316.976	316.971	316.976
PPTX	898.827	898.832	898.827	898.832	898.827	898.832
DOCX	1285.356	1285.360	1285.356	1285.360	1285.356	1285.360
PPT	3985.920	3895.936	3985.920	3895.936	3985.920	3895.936

PDF	4556.290	4456.304	4556.290	4456.304	4556.290	4456.304

Filetype	**XTEA**	
	128-bit	
Data	**Raw File Size (KB)**	**Encrypted File Size (KB)**
XML	4.548	4.552
TXT	17.900	17.904
ODS	27.655	27.656
XLSX	130.050	130.056
CSV	316.971	316.976
PPTX	898.827	898.832
DOCX	1285.356	1285.360
PPT	3985.920	3895.928
PDF	4556.290	4456.296

APPENDIX E

FEEDBACK TO FYP 1

EXAMINER 1

COMMENTS	REPLY TO COMMENTS
Gather more respondents	Thank you, Dr for your comment. This is still an ongoing process. I will try to gather as much as respondents as possible.
Fix the ERD and DFD, in terms of the relationships, and the connections, including how to handle key storage process.	ERD and DFD are still currently being fixed and it will progressively be updated during the development phase. The completed ERD and DFD will be placed in Chapter 4 Section 4.4. For key storage, that module is being removed from the system due to the complexity of the logic. Thank you, Dr, for your insight.

EXAMINER 2

COMMENTS	REPLY TO COMMENTS
Gather more respondents	Thank you, Dr, for your comment. This is still an ongoing process. I will try to gather as much as respondents as possible.

APPENDIX F

SUS TESTING QUESTIONS

1. I think that I would like to use this website frequently.
2. I found this website unnecessarily complex.
3. I thought this website was easy to use.
4. I think that I would need assistance to be able to use this website.
5. I found the various function in this website were well integrated.
6. I thought there was too much inconsistency in this website
7. I would imagine that most people would learn to use this website very quickly.
8. I found this website very cumbersome/awkward to use.
9. I felt very confident using this website.
10. I needed to learn a lot of things before I could get going with this website.

Likert Scaling:

1 - Strongly Disagree

3 - Neutral

5 - Strongly Agree

APPENDIX G

INSTRUCTION SHEET FOR UAT

- **User Registration & Authentication**

1. Make an account by going to the registration page by clicking the **'Sign Up'** button on the top right corner.
2. Fill-in the required fields and submit. Provide a valid email address as an email verification link will be sent to your email.
3. Do not verify the email yet.

- **Contact List and Access Policy Management**

1. Navigate to the login page by pressing the '**Login**' button on the top left corner of the homepage.
2. Log in using this account:

crispy.cookie0710@gmail.com
pedro$zelda_876

3. Once logged in, you will see the homepage. Search for your account using the username/email you just submitted during registration.
4. Your profile should appear. Send a friend request by clicking the '**Friend Request'** button.
5. Now log out

- **User Account Management**

1. Navigate to the login page
2. Login into your account.
3. Once logged in, navigate to 'Profile' and start changing your profile picture and biography. PSA: please do not use personal images for security purposes!
4. Next, go to 'Settings'. Go to 'Email' and check if your email is verified or not.
5. If not, go verify it by checking your gmail and clicking the link provided.
6. Refresh the email settings page. Are there any changes?
7. Now go back to 'Settings'. You can choose to change whatever you want. Just don't deactivate your account yet. Please note that some of it has OTP via email so open your Gmail on a new tab.

- **Web-Based Encryption Tool**

1. Click '**Encrypt**'.
2. Start encrypting a document file. **DOCUMENT FILE (pdf, docs, docs, xlsx, xls, pptx, ppt, txt).** There should be a success banner on top of the page.
3. Now, click '**Decrypt**'. Your encrypted file should be present there. There is a button with three dots on the left.
4. You must test the decrypt function so click '**Decrypt File**' and follow the steps. The other one is optional, but you can do it if you want.

- **File Sharing Management**

1. Go to '**Friend Request**' and accept the friend request.
2. Go to '**Manage Files**'.
3. There should be your submitted file there. Like the decrypt page, there is a button. You must test the '**Share File**'.
4. Next, go to '**Shared Files**'. Here you can see your shared files. Like the previous page, there is a button with three dots. You must test the '**View File**'. I hope you remember your access phrase.

- **Account Deactivation**

1. Navigate to 'Settings' page and click 'Deactivate' button.
2. Follow the instructions