



GRAPHICAL SECRET KEY VERIFICATION SYSTEM

A PROJECT REPORT

Submitted by

DHARSHINI.S [211418104050]

JOSEPHINE BENITA.T [211418104103]

KAMALI.B [211418104108]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2022

BONAFIDE CERTIFICATE

Certified that this project report “**GRAPHICAL SECRET KEY VERIFICATION SYSTEM**” is the bonafide work of “DHARSHINI.S [211418104050], JOSEPHINE BENITA.T [211418104103], KAMALI.B [211418104108] “who carried out the project work under **MRS.A.KANCHANA.,M.E.,** supervision.

SIGNATURE

SIGNATURE

Dr.S.MURUGAVALLI,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

MRS.A.KANCHANA.,M.E.,
SUPERVISOR
ASSISTANT PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project
Viva-Voce Examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION BY THE STUDENT

We DHARSHINI.S [211418104050], JOSEPHINE BENITA.T [211418104103], KAMALI.B [211418104108] hereby declare that this project report titled “GRAPHICAL SECRET KEY VERIFICATION SYSTEM”, under the guidance Of MRS.A.KANCHANA.,M.E., is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

DHARSHINI S

JOSEPHINE BENITA T

KAMALI B

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,Ph.D.**, for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI, M.E.,Ph.D.**, for the support extended throughout the project.

We would like to thank my Project Guide **MRS.A.KANCHANA.,M.E.**, and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

DHARSHINI.S

JOSEPHINE BENITA.T

KAMALI.B

ABSTRACT

Computer security depends largely on passwords to authenticate human users from attackers. The most common computer authentication method is to use alphanumerical usernames and password. Users tend to pick password that can be easily guessed. So, here we are using three step authentication system, in order to keep our information highly secure. The three step verification systems are as follows: Alpha numerical user name and password, Picking the colors in the correct sequence and Drag and drop the image in grid(pattern). Textual passwords are the most often used authentication method for all websites and services. Textual passwords are made up of a series of letters and numbers that may or may not include special characters or numbers. In most circumstances, users can log into several accounts with just one username and password. However, they are not completely secure. Users may have difficulty remembering a password that is long and random appearing. Instead, they create short, simple, and insecure passwords. Graphical passwords have been designed to try to make passwords more memorable and easier for people to use and, therefore, more secure. Using a graphical password, users click on images and colors rather than typing alphanumeric characters. This graphical password authentication system can be applied to any sector.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	Ii
	LIST OF TABLES	Vii
	LIST OF FIGURES	Viii
	LIST OF SYMBOLS, ABBREVIATIONS	Ix
1.	INTRODUCTION	
	1.1 Over view	2
	1.2 Problem Definition	3
2.	LITERATURE SURVEY	5
3.	SYSTEM ANALYSIS	
	2.1 Existing System	9
	2.2 Proposed system	10
	2.3 Feasibility Study	10
	2.4 Hardware Environment	13
	2.5 Software Environment	13

CHAPTER NO.	TITLE	PAGE NO.
4.	SYSTEM DESIGN	
	4.1. ER diagram	15
	4.2 Data Flow Diagram	16
	4.3 UML Diagrams	17
	4.4 Data Dictionary	18
5.	SYSTEM ARCHITECTURE	
	5.1 System Architecture Diagram	24
	5.2 Module Design Specification	25
6.	SYSTEM IMPLEMENTATION	
	6.1 Client-side coding	28
	6.2 Server-side coding	39
7.	SYSTEM TESTING	
	7.1 Unit Testing	52
	7.2 Integration Testing	53
	7.3 Test Cases	53
8.	CONCLUSION	
	8.1 Results & Conclusion	56
	8.2 Future Enhancements	56

CHAPTER NO.

TITLE

PAGE NO.

APPENDICES

A.1 Sample Screens

57

REFERENCES

63

LIST OF TABLES:

TABLE.NO	NAME OF THE TABLE	PAGE NO
TABLE.3.1	Economic Feasibility	11
TABLE.4.1	Data dictionary	22
TABLE.7.1	Functional Test cases	53
TABLE.7.2	Security Test cases	54

LIST OF FIGURES:

FIG.NO	NAME OF THE FIGURE	PAGE NO
FIG.4.1	ER Diagram	15
FIG.4.2	Data flow diagram	16
FIG.4.3	Use case diagram	17
FIG.4.4	Class diagram	18
FIG.4.5	Sequence diagram	19
FIG.4.6	Activity diagram	20
FIG.4.7	State chart diagram	21
FIG.4.8	System architecture diagram	24
FIG.A1	Sample screen shots A1	57
FIG.A2	Sample screen shots A2	58
FIG.A3	Sample screen shots A3	58
FIG.A4	Sample screen shots A4	59
FIG.A5	Sample screen shots A5	59
FIG.A6	Sample screen shots A6	60
FIG.A7	Sample screen shots A7	60
FIG.A8	Sample screen shots A8	61
FIG.A9	Sample screen shots A9	61
FIG.A10	Sample screen shots A10	62
FIG.A11	Sample screen shots A11	62

LIST OF ABBREVIATIONS:

1. HTML - Hyper Text Markup Language.
2. CSS - Cascading Style Sheet.
3. XML - Extensible Markup Language.
4. OTP - One Time Password.
5. SMTP - Simple Mail Transfer Protocol.
6. XHTML - EXtensible Hyper Text Markup Language.
7. ESMTP - Extended Simple Mail Transfer Protocol.
8. NPM - Node Package Manager.
9. APT - Application Programming Interface.
10. SQL - Structured Query Language.
11. MySQL - My Structured Query Language.
12. UML - Unified Modelling Language.
13. ER - Entity Relationship.
14. CORS - Cross Origin Resource Sharing.
15. UI - User Interface.

CHAPTER 1

INTRODUCTION

1.INTRODUCTION

1.1 OVER VIEW:

Graphical password authentication system can be applied to any sector. Initially, We have the option to register. We will first click on the register button after that we have to enter our details .

- **Alpha numerical user name and password.**

The first level of authentication will be a normal authentication. After entering our details, click on the next button.

- **Picking the colours in the correct sequence.**

Next the level two of authentication will be to create a pattern of colors by selecting it so quickly.

If we want to select green ,red, blue then my color pattern will be green, red, blue for authentication .

- **Drag and drop the image in grid(pattern).**

Next the level three of authentication is, where we have to drag and drop the images to create a pattern in the grid like structure.

- ✓ After these 3 levels, there is an another level of authentication called email authentication where we can receive OTP during registration.
- ✓ After registration ,the user will be redirecting to the user interface called file system ,where we can upload our files in a secure manner. The file can be any format and also we can view the file whenever we want by logging into it.

1.2 PROBLEM DEFINITION

The risk of cybercrime and data breaches is increasing as more and more things move online. On both online and offline venues, passwords are critical for data security. The most prevalent method of gaining access to our accounts is through passwords. Today, providing security is seen as a key issue in a variety of domains, including internet banking and other areas where a high level of security is required to protect user data confidentiality. Static passwords make it easy for hackers to gain access to a user's account. As a result, using only single-level authentication elements to assure security is insufficient. An idea is to provide three tiers of protection for actual users authentication. When data is entered into a system, the authentication process begins. The system then compares the information entered to the information in the database. The user can access the system if it matches. Parents, friends, coworkers, social media, and government legislation all play a part in the social environment. When utilizing traditional username-password authentication, alphanumeric passwords are either easy to guess or difficult to remember. Because memorizing several passwords is difficult, users often use the same password for all of their accounts. Other authentication methods, such as color graphical passwords and drag-and-drop graphical passwords, are used to overcome the limitations associated with the traditional username-password authentication methodology.

CHAPTER 2

LITERATURE SURVEY

2.LITERATURE SURVEY

A lot of research has been done to password authentication. Some of them are,

Title : Efficient Password-Based Threshold Single-Sign-On Authentication for Mobile Users against Perpetual Leakage. Author: Yuan Zhang, Chunxiang Xu, Hongwei Li, Kan Yang, Nan Cheng, Xuemin Shen, Year: 2021, Journal name: IEEE, Methodology used: single-sign-on authentication scheme, Observations: Pros: Reduces the load of memorizing several passwords .Cons: Using a single password increases the chances of password vulnerability[1]. **Title: : Password-Authenticated Decentralized Identities.** Author: Pawel Szalachowski, Year: 2021, Journal name: IEEE, Methodology used: Decentralized Identity Management, Observations: Pros: Users control what data to share, Mobile phones act as digital identity devices ,Cons: Users may have trouble receiving all the credentials they need to store in their wallets ,Users may have trouble selecting a reputable company to manage their identity[2]. **Title : A Novel Hybrid Textual-Graphical Authentication Scheme With Better Security, Memorability, and Usability.** Author: Shah zaman nizamani , Syed raheel hassan , Riaz ahmed shaikh 2 , Ehab atif abozinadah, and Rashid mehmood , (senior member, ieee), Year: 2021, Journal name: IEEE, Methodology used: Password security, Textual Password, Graphical password, Observations: Pros: Graphical password schemes provide a way of making more human-friendly passwords. Cons: Easily tracked[3]. **Title : A Longitudinal Study on Web-Sites Password Management (in)Security: Evidence and Remedies.** Author: Simone Raponi, Roberto Di Pietro, Year: 2020, Journal name: IEEE, Methodology used: Secret sharing algorithm, Email based authentication. Observations: Pros: Can be universally used. User friendly experience. Cons: Less space weaker confidentiality. **Title : A Password-Based Authentication System Based on the CAPTCHA AI Problem.** Author: Masoud Alajmi, Ibrahim Elashry Hala ,S. El-Sayed ,Osama S. Faragallah, Year: 2020, Journal name: IEEE, Methodology used: The brute force attack and the replay attack ,

CAPTCHA AI Observations: Pros: protect against abusive bot traffic. Cons: no protection against machine learning and ai , irritates users. sometimes very difficult to read[5].**Title : Doodle-Based Authentication Technique Using Augmented Reality.** Author: Waqas wazir , Hasan ali khattak , (senior member, ieee), Ahmad almogren , (senior member, ieee), Mudassar ali khan and Ikram ud din , (senior member, ieee),Year: 2019,Journal name: IEEE, Methodology used: Augmented reality, gesture recognition, password, doodle-based authentication, usable security. Observations: Pros: It is based on simple rules and very famous among the users. Cons: Fails in some browsers. Time consuming. Sometimes it is difficult to read[6].**Title : A Secure Hybrid Authentication Scheme using PassPoints and Press Touch Code.** Author: Saiful azad1, Noor elya afiqah che nordin , Nur nadhirah ab rasul , Mufti mahmud, (senior member, ieee), Kamal z. zamli1 , (senior member, ieee),Year: 2019,Journal name: IEEE, Methodology used. Authentication schemes, Graphical authentication schemes, hybrid authentication schemes, Locimetric schemes, Passpoints, Press touch code. Observations: Pros: Users select the pre selected faces from a grid. Cons: Biases towards faces and not being secure against shoulder surffing, Brute force and dictionary attacks[7].**Title : Authentication using text and graphical password.** Author: Abhilash M Joshi, Balachandra Muniyal. Year: 2018,Journal name: IEEE , Methodology used: Token Based Authentication Knowledge based authentication Biometric based authentication. Observations: Pros: Images in this matrix will be shuffled within, to avoid eavesdropping and shoulder surfing. The shuffle feature of this graphical password will stand against various attacks. Cons: It is very expensive[8].**Title : A Secure and User-Friendly Graphical Password Scheme.** Author: Gi-Chul Yang, Year: 2017,Journal name: IEEE, Methodology used: A mixture of recognition and recall based method. Observations: Pros: It's a combination of recognition and recall based technique, hence provides flexibility. Cons: This system perform some complex actions like pre processing[9].**Title : A Cued-Recall and Emotion Classification Graphical Password Authentication Scheme .**Author:

Danilo E. Vieira¹ , Tonny L. Mesquita Abreu¹ , Max E. Vizcarra Melgar¹ , Luz A. M. Santander² ,Year: 2017,Journal name: IEEE, Methodology used: Captcha security method. Recognition-Based method. Recognition of emotional images. Cued-Recall security layer Observations: Pros: highly usable and could improve the security level on ATM machines. perception of security is enhanced using human emotions as main decision factor. Cons: recall graphical password schemes are vulnerable, Strong Passwords are difficult to memorize.

CHAPTER 3

SYSTEM ANALYSIS

3.SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

A lot of research has been done to password authentication. Text passwords have been used in authentication systems for many decades. Users must recall the textual strings selected during registration to pass authentication. However, there are some serious problems with text passwords---recollection and security. Hence, various graphical-password authentication systems have been proposed to solve the problems of text passwords. Previous studies indicate that humans are better at recognizing and recalling images than texts. The existing system includes text based password , biometric password, pattern password, Token authentication, Captcha's and others. The password which is typed by using keyboard or mouse can easily identified using key stroke, mouse movement and shoulder movement. Hand geometry and fingerprinting require a scanner equipment, and these approaches are ineffective in the treatment of arthritis, rheumatism, and injury dirtiness, roughness. Signatures are easily guessable and hackable.

DISADVANTAGES:

- In the system of facial recognition .The accuracy of the outcome is poor, and it also necessitates the use of a camera.
- We'll need a touch panel and an optical pen for signature recognition. Because signatures are frequently changing and easy to forge, signature recognition accuracy is low.
- Users using passwords are frequently unaware of their security. They need to use alphanumeric, lowercase, and uppercase to set a strong password which is difficult for the users to remember.

3.2 PROPOSED SYSTEM:

Passwords aren't user friendly Consumers are encouraged to make their passwords more complex by using numbers, uppercase letters, lowercase letters, and unusual characters. This makes them difficult to recall. And if people can't remember them, they write them down or use the same password on multiple websites...which makes them less secure. Passwords aren't safe because they can be exchanged, guessed, or stolen. Over half of young people admit to sharing their log-in information with friends, and 59 percent admit to reusing the same password across various websites. It's tough to strike a balance between security and usability: a memorable password is unsafe, but a secure password is difficult to remember. So, what's the alternative .For that in our proposed system we are using three level of authentication for improved security.

1.Alphanumeric password

2.Picking the colors

3.Drag and drop

In first level of authentication we are using alphanumeric password which ask us to use a combination of numbers and letters, which creates stronger passwords. Second level of authentication is picking the color here there will be few colors we have to Pick the correct color code.

Third level of authentication is drag and drop here there will be 4x4 grid box which contains 4 images at random place we have to drag and drop the images at correct place in the grid.

3.3 FEASIBILITY STUDY:

- ECONOMICAL
- TECHNICAL
- SOCIAL

3.3.1. ECONOMICAL FEASIBILITY:

This website is economically feasible since any device like laptop or mobile phone have microphone access through which the authentication is given. Open-source python framework called flask is used in the user interface.

MODULE	CATEGORY
Alpha numerical user name and password.	ORGANIC
Picking the colors	SEMI DETACHED
Drag and drop	SEMI DETACHED

TABLE 3.1

3.3.2. TECHNICAL FEASIBILITY:

1.Node.JS - Node.js is secure, third-party packages may need more security standards to safeguard your web app. Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications.

Node.js is primarily used for non-blocking, event-driven servers, due to its single-threaded nature. It's used for traditional web sites and back-end API services, but was designed with real-time, push-based architectures.

2.CSS - Cascading Style Sheets CSS is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

3.JavaScript -JavaScript is the programming language of web. JavaScript is the language that browsers use. It's easy to get started with and to understand. You can get going right away - unlike other languages, you don't have to install a bunch of programs before you can even begin. It is used to develop interactive web applications. JavaScript can power features like interactive images, carousels, and forms. The language can be used with back-end frameworks like Node.js to power the mechanics behind a web page.

4.Python - We send OTP to register the user email using SMTP (Simple Mail Transfer Protocol). The json body, which is a sequence of numbers, is transformed to an otp message and delivered to the appropriate email address (from server to client). These routines were developed in Python because the smtplib module defines an SMTP client session object that can be used to deliver email to any Internet system that has an SMTP or ESMTP listener daemon.

5.NPM - NPM is the package manager for the Node JavaScript platform. It puts modules in place so that node can find them, and manages dependency conflicts intelligently. Programs are executed in the node environment, which is a free, open-source, cross-platform runtime environment based on JavaScript. Designed to run on the server or client side of a computer without the requirement for an internet connection. User information is hashed using the "crypto" node.js module. The CSS code is compiled using node-sass NPM package.

6.Firebase API - APIs are a means to make services available to our own or third-party apps. Here passwords are stored and matched in three levels of authentication using the Firebase API, which stores and syncs data in real time. The service allows developers to create apps without needing to manage servers or write server-side code.

3.3.3 SOCIAL FEASIBILITY:

- The more security it provides for your account.
- It help your personal accounts stay private and secure
- Our project is user friendly and simple to use.
- The goal of this project is to create a secure storage system for files.
- It takes very little time to store files, and it is straightforward and quick to use while also being more secure.
- It is not difficult to remember the passwords.

3.4 HARDWARE ENVIRONMENT:

- Processor - I5
- Speed - 3 GHz
- RAM - 8 GB (min)
- Hard Disk - 100 GB
- Monitor - LCD

3.5 SOFTWARE ENVIRONMENT:

- Operating System: Windows 7/10/11
- Server: NPM, Mysql ,Firebase API.
- Front End: HTML, CSS, Node.JS
- Server-side Script: Python, JavaScript.

CHAPTER 4

SYSTEM DESIGN

4.SYSTEM DESIGN

4.1 ER DIAGRAM:

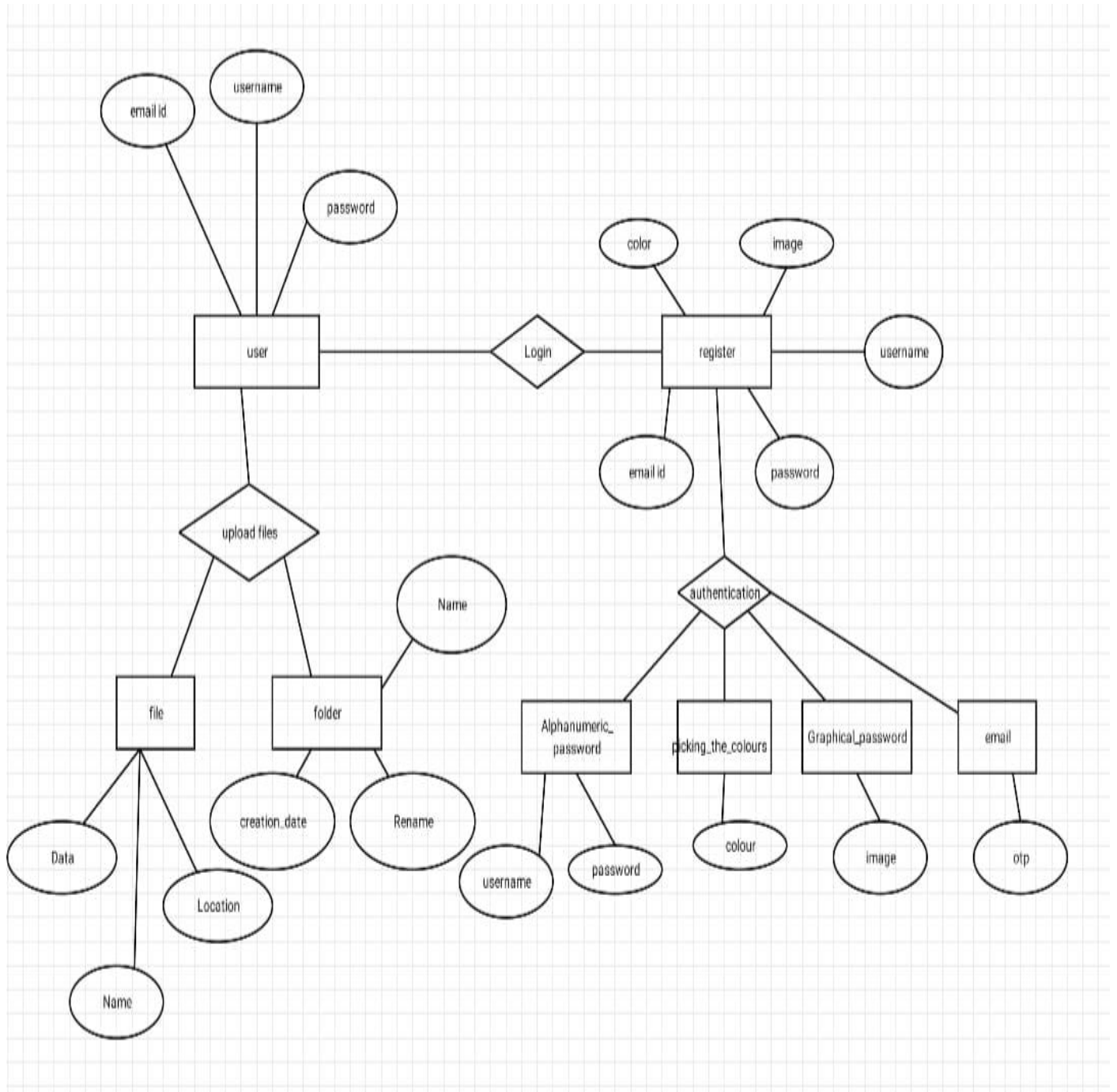


FIG.4.1

4.2 DATA FLOW DIAGRAM:

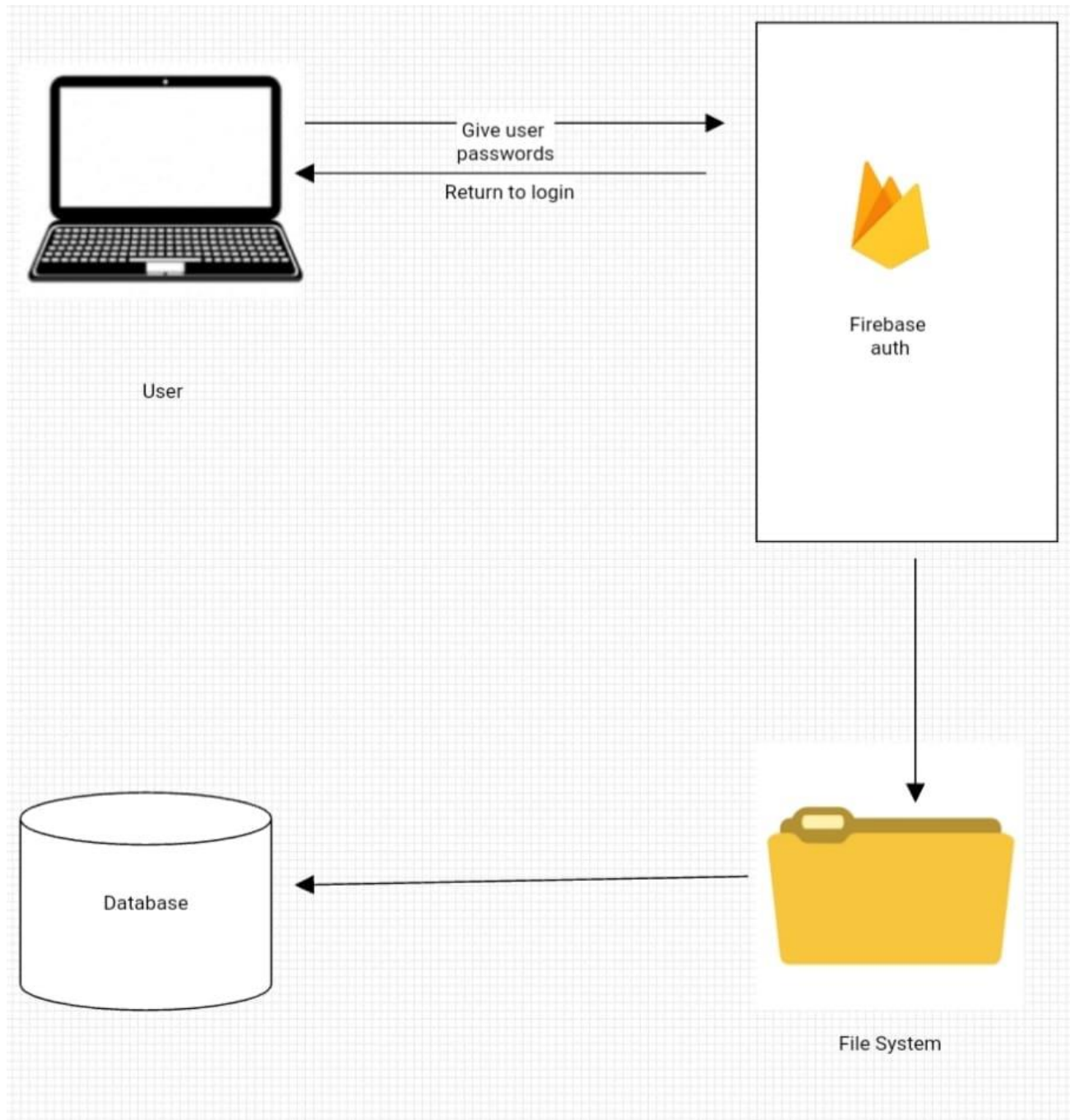


FIG 4.2

4.3 UML DIAGRAMS:

4.3.1.USE CASE:

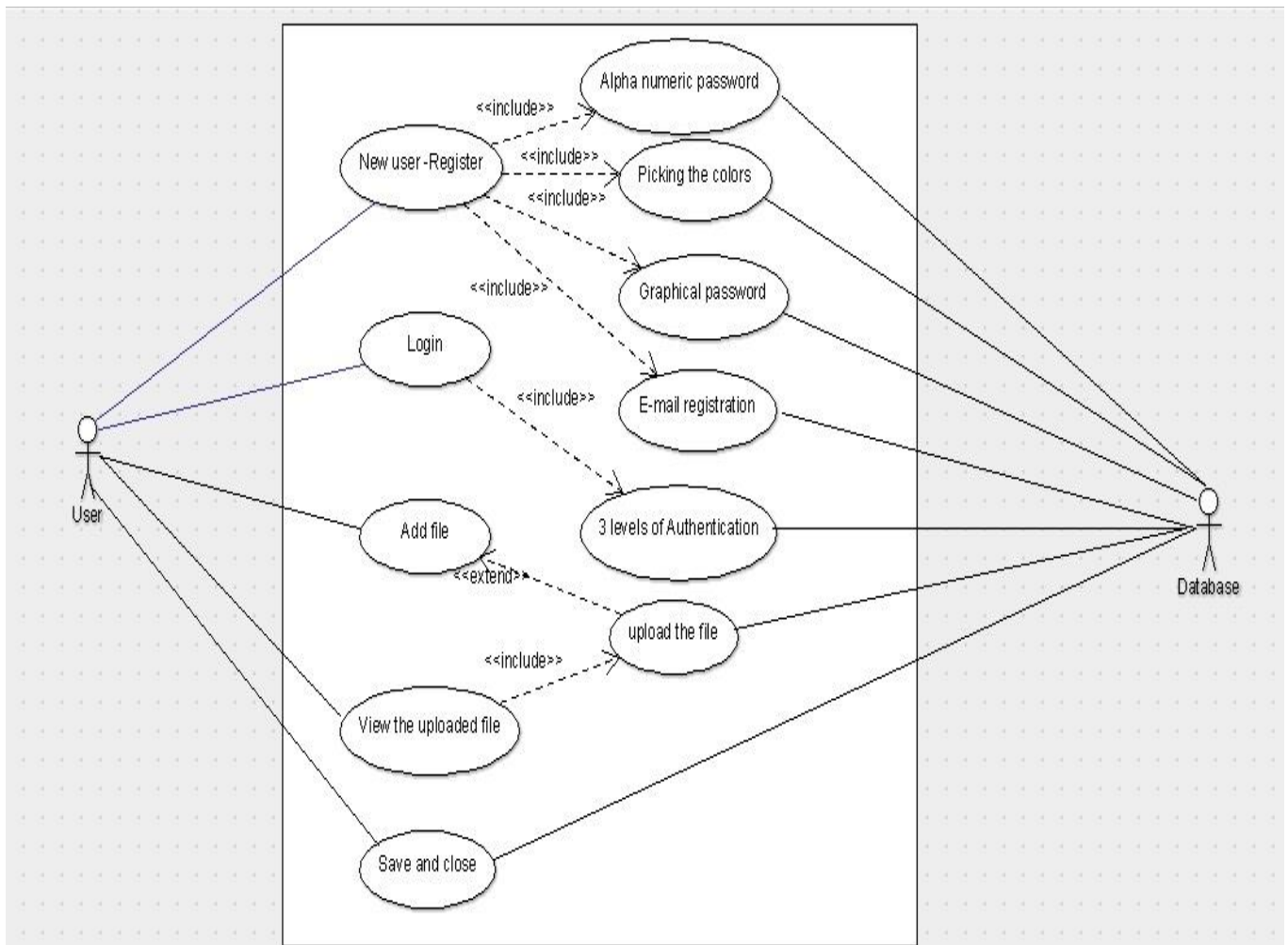


FIG 4.3

4.3.2.CLASSDIAGRAM:

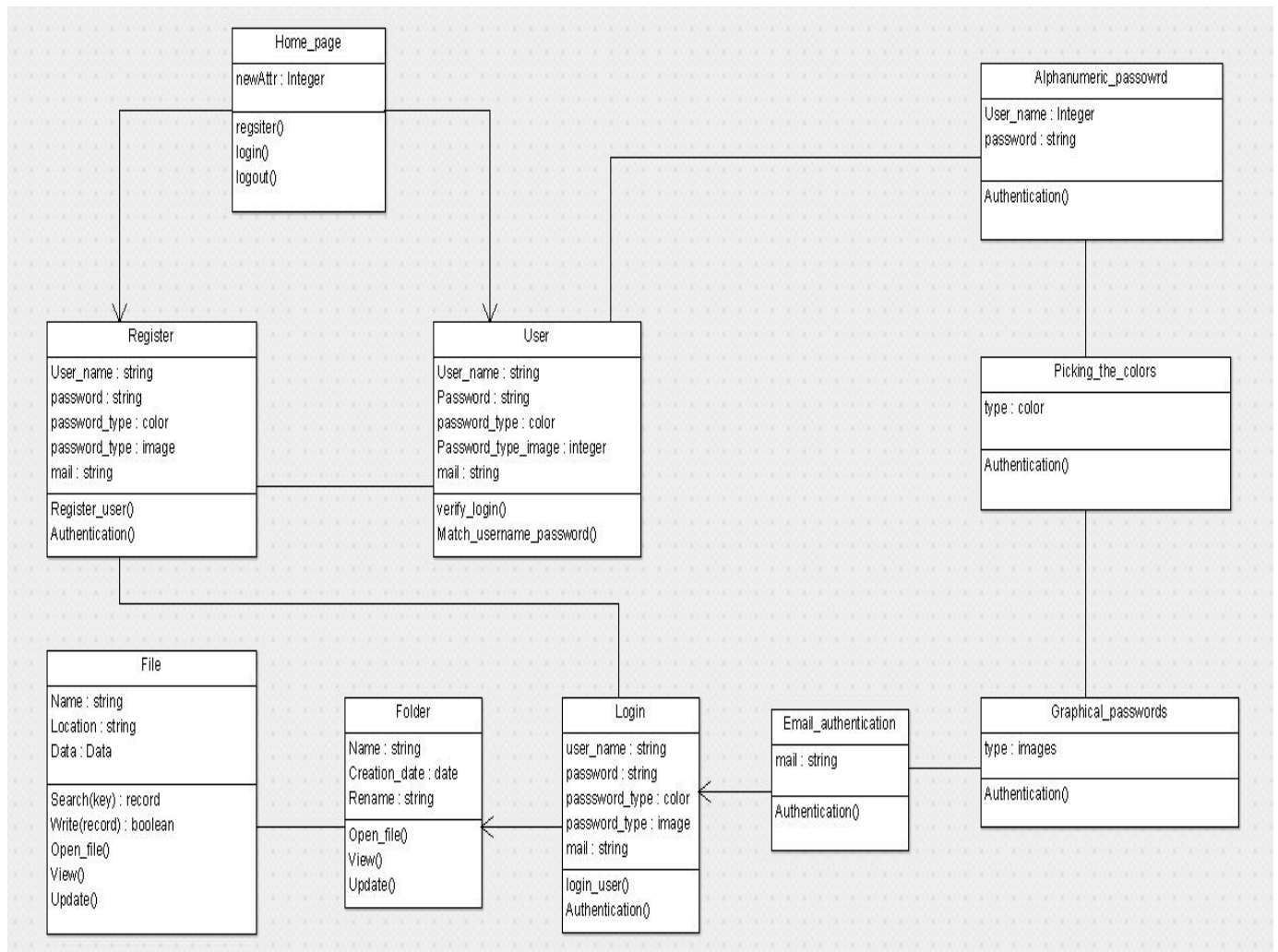


FIG 4.4

4.3.3. SEQUENCE DIAGRAM:

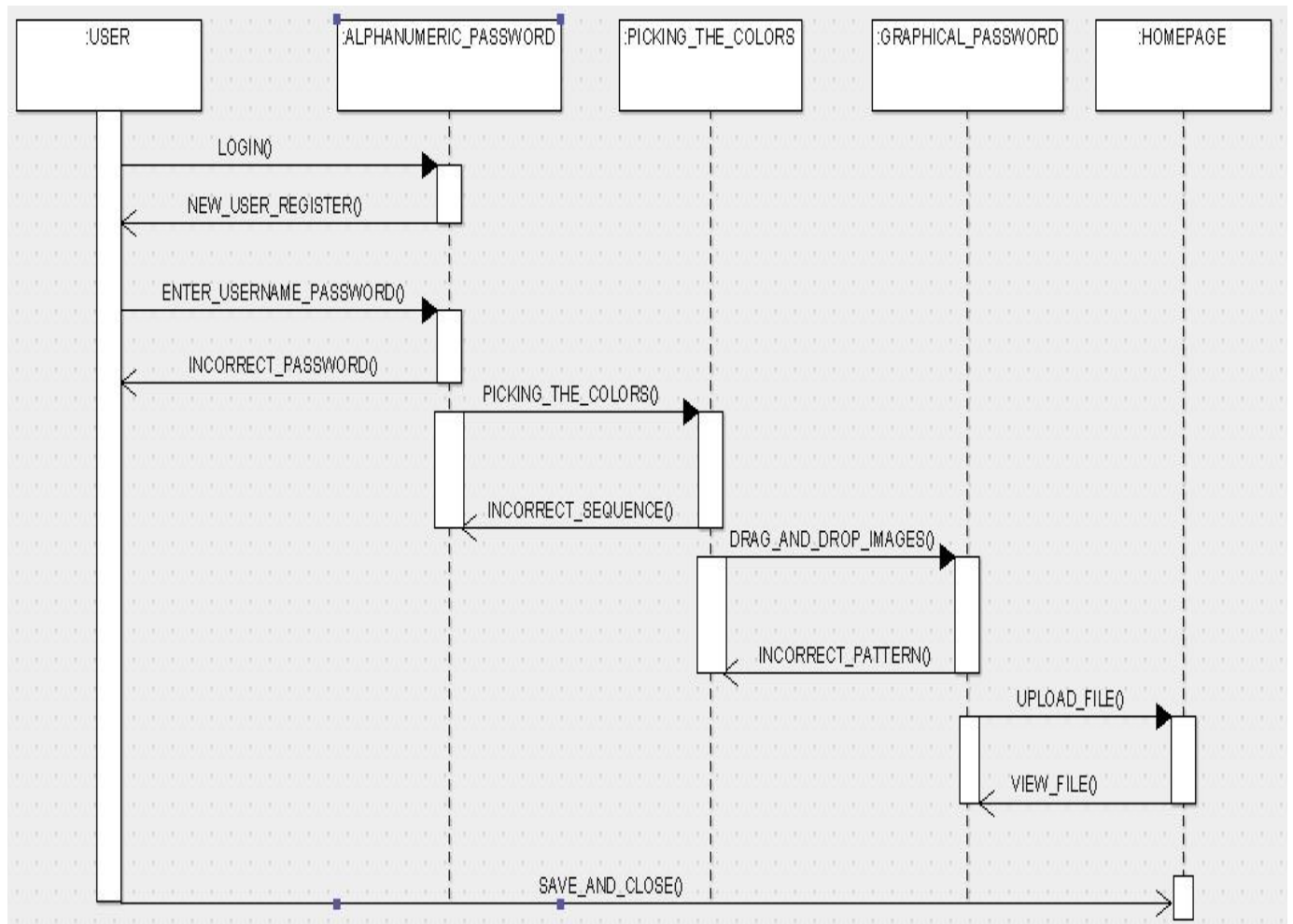


FIG.4.5

4.3.4. ACTIVITY DIAGRAM:

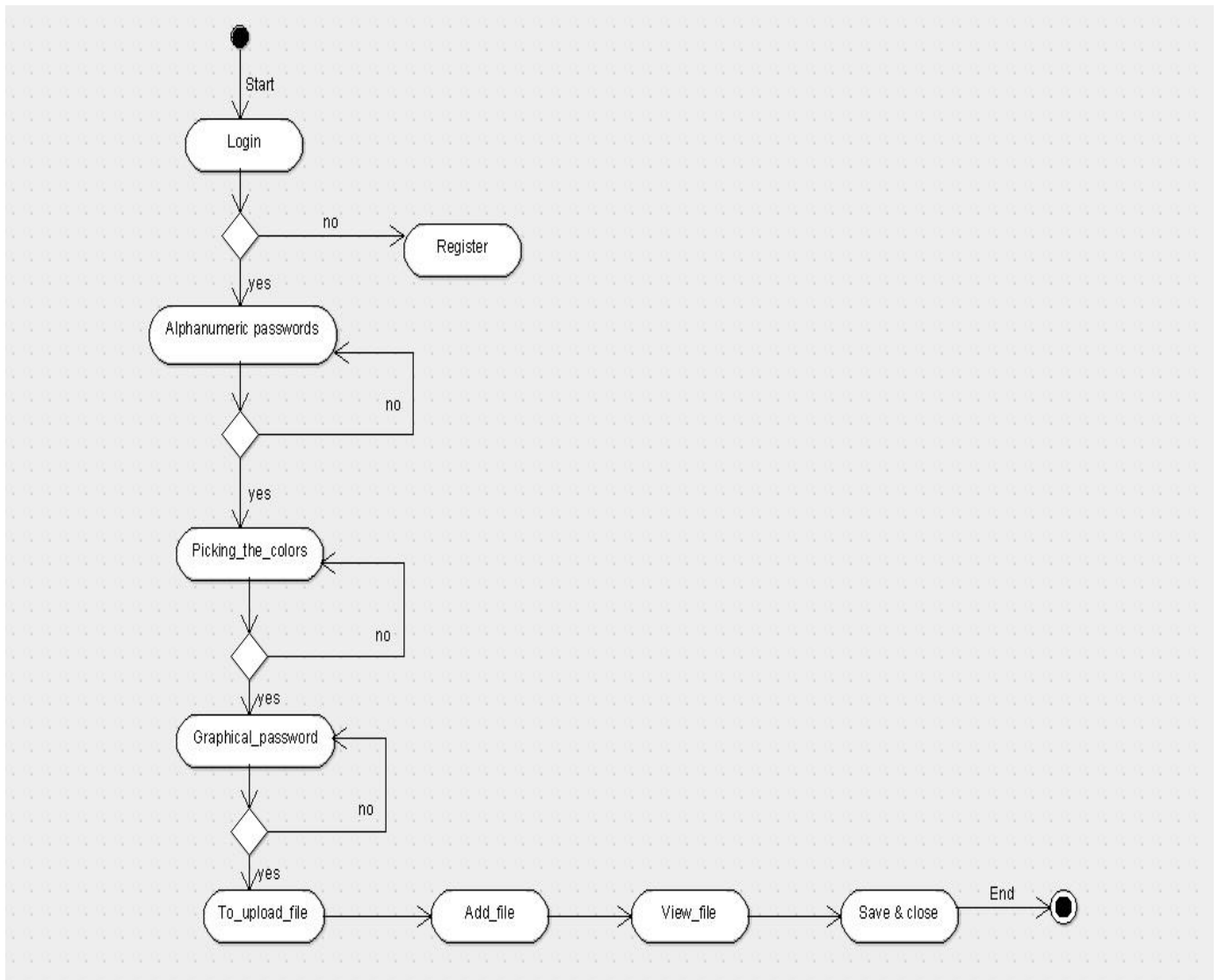


FIG 4.6

4.3.5. STATE CHART DIAGRAM:

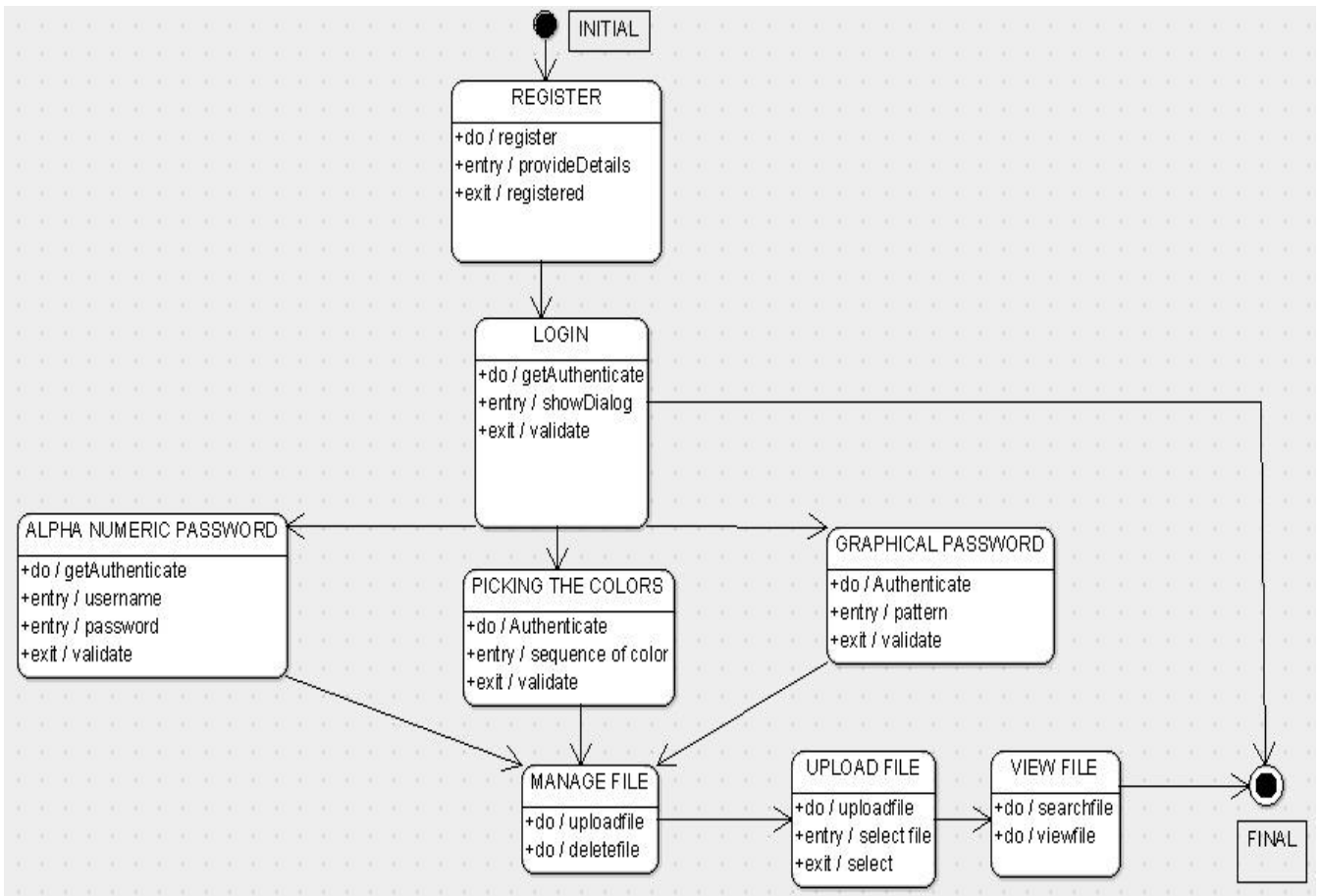


FIG.4.7

4.4 DATA DICTIONARY:

S.NO	FIELD NAME	DESCRIPTION	DATA TYPE	DATA SIZE
1.	User name	User id	string	100
2.	Text password	Password created by the user	string	100
3.	Color password	Pick the correct color code	string	100
4.	Grid password	Drag and drop the images at correct position	string	100

TABLE 4.1

CHAPTER 5

SYSTEM ARCHITECTURE

5.SYSTEM ARCHITECTURE

5.1 SYSTEM ARCHITECTURE DIAGRAM

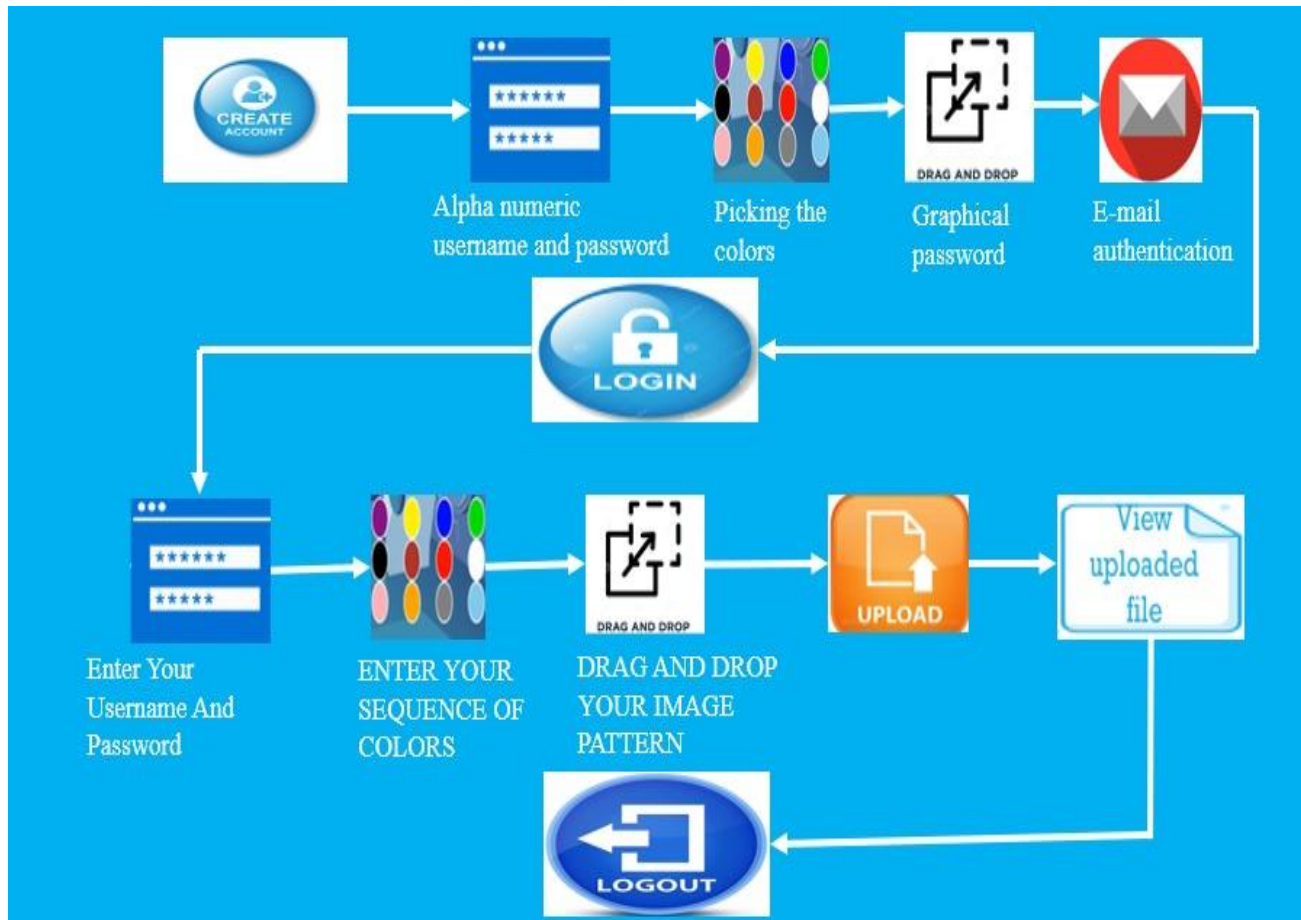


FIG.5.1

DESCRIPTION:

The user initially wishes to register. The user must complete three steps of registration, as well as email authentication by receiving an OTP for the email address provided. After registering, the user must login and go through all of the verification steps. After logging in, the user will be sent to the file system user interface, where we can upload our files and examine them whenever we like. We can finally logout.

5.2 MODULES, DESIGN AND SPECIFICATION:

5.2.1 REGISTRATION MODULE

- In this module, the user will get registered in a particular website. It will be formed like a structure where the user details will be filled.
- It will have the three fields of authentication.
 - ✓ Alpha numerical user name and password.
 - ✓ Picking the colors in the correct sequence.
 - ✓ Drag and drop the image in grid(pattern).
- As this module is present online, the user can register from anywhere they have access to the internet.

5.2.2 LOGIN MODULE

- A user reaches a login page on a website they have previously created an account with.
- Enables registered users to log in to a site using credentials
- The user provides their unique ID and key to verify their identity.
- users input their credentials on the website's login form. That information is then sent to the authentication server where the information is compared with all the user credentials on file.
- When a match is found, the system will authenticate users and grant them access to their accounts.

5.2.3 THREE LEVEL AUTHENTICATION MODULE

1.Alpha numerical user name and password.

The first level of authentication will be a normal authentication

After entering our details, click on the next button.

2.Picking the colors in the correct sequence.

Next the level two of authentication will be to create a pattern of colors by selecting it so quickly.

If we want to select green ,red ,blue then my color pattern will be green, red, blue for authentication .

3.Drag and drop the image in grid(pattern).

Next the level three of authentication is, where we have to drag and drop the images to create a pattern of colors.

5.2.4 FILE SYSTEM MODULE

- After login, the user can upload their files and also he/she can able to view it after uploading.
- The File-System module contains a number of interfaces that let you perform standard file management operations on files already resident on the filesystem.
- You can rename, copy, or delete any file, and you can set any available properties for the file.
- And also we can upload any format of files like audio ,video ,pdf ,image ,etc :-

CHAPTER 6

SYSTEM IMPLEMENTATION

6.SYSTEM IMPLEMENTATION

6.1 CLIENT-SIDE CODING:

```
import * as Grid from './gridView';
export const elements = {
  loginTop: document.querySelector('#login'),
  registerTop: document.querySelector('#register'),
  register: document.querySelector('.register'),
  login: document.querySelector('.login'),
  remove: document.querySelector('#remove'),
  mainBody: document.body,
  newfile_btn: document.getElementsByClassName("db_addfile"),
  newfolder_btn: document.getElementsByClassName("db_addfolder"),
  addFile_btn: document.getElementsByClassName("bn33"),
  file_delete_btns: document.getElementsByClassName("file_delete_btn"),
  preview_btns: document.getElementsByClassName("preview_btn"),
  privacy_btns: document.getElementsByClassName("privacy_btn"),
  preview_close_btn: document.getElementsByClassName("preview_close_btn"),
  preview: document.getElementsByClassName("preview"),
  logout_btn: document.getElementsByClassName("logout_btn"),
  notification_btn: document.getElementsByClassName("notification_btn"),
  open_notification_btn: document.getElementsByClassName("open_notification_btn"),
  close_notifiatiion: document.getElementsByClassName("notification_close_btn"),
  send_otp_btn: document.getElementsByClassName('send_otp'),
  verify_otp_btn: document.getElementsByClassName('verify_otp'),

};
```

```
export const messages = {  
  loginfail: 'Someone trying to login your account',  
}
```

```
export const elementStrings = {  
  formOne: '#form--1',  
  formTwo: '#form--2',  
  formFour: '#form--4',  
  group: '.form__group',  
  icon: '.circle',  
  username: '#username',  
  password: '#password',  
  opt: '#otp_field',  
  email: '#user_email',  
  retypepassword: '#retype_password',  
  pattern: '#pattern',  
  securityquest: '#securityquest',  
  reset: '#reset',  
  nextOR: '#register--one',  
  nextTR: '#register--two',  
  nextHR: '#register--three',  
  nextFR: '#register--four',  
  nextOL: '#login--one',  
  nextTL: '#login--two',  
  nextHL: '#login--three',  
  drag: '.graphic__row--drag',  
  drop: '.graphic__row--drop',
```

```
}
```

```
const markups = {
```

```
one:
```

```
`<div class="container">
```

```
<div class="container__form">
```

```
<div class="margin-bottom center-text">
```

```
<h2 class="sub-heading">Level One</h2>
```

```
<p class="paragraph">%_DESCRIPTION%</p>
```

```
</div>
```

```
<form onsubmit="event.preventDefault();" class="form" id="form--1 "
```

```
autocomplete="off">
```

```
<div class="form__group">
```

```
<input type="text" class="form__input" placeholder="username" id="username"
```

```
minlength="3" required />
```

```
<label for="username" class="form__label">username</label>
```

```
</div>
```

```
<div class="form__group">
```

```
<input type="password" class="form__input" placeholder="password" id="password"
```

```
minlength="6" required />
```

```
<label for="password" class="form__label">password</label>
```

```
</div>
```

```
%_retypepassword_%
```

```
<div class="form__group">
```

```
<button id="%_BTNLEVEL%" class="btn btn--primary">Next &rarr;</button>
```

```
</div>
```

```
</form>
```

```
</div>
```


</div>`,

two:

`<div class="container">

<div class="container__icons">

<div class="margin-bottom center-text">

<h2 class="sub-heading">Level Two</h2>

<p class="paragraph">%_DESCRIPTION%</p>

</div>

<div class="row">

<div class="col-1-of-5">

</div>

<div class="col-1-of-5">

</div>

<div class="col-1-of-5">

</div>

<div class="col-1-of-5">

</div>

<div class="col-1-of-5">

</div>

<div class="col-1-of-5">

</div>

```

<div class="col-1-of-5">
  <span id="red" class="circle circle--red">&nbsp;</span>
</div>
<div class="col-1-of-5">
  <span id="white" class="circle circle--white">&nbsp;</span>
</div>
<div class="col-1-of-5">
  <span id="pink" class="circle circle--pink">&nbsp;</span>
</div>
<div class="col-1-of-5">
  <span id="orange" class="circle circle--orange">&nbsp;</span>
</div>
<div class="col-1-of-5">
  <span id="grey" class="circle circle--grey">&nbsp;</span>
</div>
<div class="col-1-of-5">
  <span id="skyblue" class="circle circle--skyblue">&nbsp;</span>
</div>
<div class="col-1-of-5">&nbsp;</div>
</div>
</div>
<div class="container__form">
  <form class="form" id="form--2" autocomplete="off">
    <div class="form__group">
      <input type="password" class="form__input" placeholder="RGB pattern"
        id="pattern" required readonly />
      <label for="pattern" class="form__label">pattern</label>
    </div>
  </form>
</div>

```

```

<div class="form__group">
  <button type="button" id="reset" class="btn btn--primary">Reset</button>
  <button type="button" id="%_BTNLEVEL%" class="btn btn--primary btn--
right">Next &rarr;</button>
</div>
</form>
</div>
</div>
</div>`,

```

three: `

```

<div class="container">
  <div class="container__form">
    <div class="margin-bottom center-text">
      <h2 class="sub-heading">Level Three</h2>
      <p class="paragraph">%_DESCRIPTION%</p>
    </div>
    <div class="graphic margin-bottom">%_GRID%</div>
    <div class="form__group">
      <button type="button" id="%_BTNLEVEL%" class="btn btn--
primary">%_BTNDESC%</button>
    </div>
  </div>`,

```

four: `

```

<div class="container">
  <div class="container__icons">
    <div class="margin-bottom center-text">

```

```

<h2 class="sub-heading">Level Four</h2>
<p class="paragraph">%_DESCRIPTION%</p>
</div>
</div>
<p style="text-align:center;font-size:24px;">
What is Your pet name ?
</p>
<div class="container__form">
<form class="form" id="form--4" autocomplete="off">

<div class="form__group">
<input type="text" class="form__input" placeholder="Answer" id="securityquest"
required />
<label for="pattern" class="form__label">Question</label>
</div>
<div class="form__group">
<button type="button" id="reset" class="btn btn--primary">Reset</button>
<button type="button" id="%_BTNLEVEL%" class="btn btn--primary btn--
right">Next &rarr;</button>
</div>
</form>
</div>
</div>`,
five:`
<div class="container">
<div class="container__form">
<div class="margin-bottom center-text">
<h2 class="sub-heading">Level Five</h2>

```

```

<p class="paragraph">Enter Your Email address</p>
</div>
<form onsubmit="event.preventDefault();" class="form" id="form--1 "
autocomplete="off">
<div class="form__group">
<input class="form__input" placeholder="Email" type="email" id="user_email"
required />
<label for="username" class="form__label">Email</label>
</div>
<div class="form__group">
<button id="" class="send_otp btn btn--primary">Send OTP</button>
</div>
<div class="form__group">
<input type="number" class="form__input" placeholder="OTP" id="otp_field" />
<label for="password" class="form__label">OTP</label>
</div>
<div class="form__group">
<button id="" class="verify_otp btn btn--primary">Verify OTP</button>
</div>
</form>
</div>
</div>
、
};

const replacements = {
oneLD: 'Please enter your username and password.',

```

```

oneRD: 'Please choose username and password. Username must be at least 3
characters and password must be at least 6 characters.',
twoLD: 'Please click on the colors below in the order you picked during registration.',
twoRD: 'Please click on the colors below in any order to create a pattern.',
threeLD: 'Please drag and drop the images below to the locations you specified during
registration.',
threeRD: 'Please drag and drop the images below to your preferred locations to create
a graphical pattern.',
fourLD: 'Please enter your answer.',
fourRD: 'Please enter your answer.',
};

```

```

const placeholders = {
  description: '%_DESCRIPTION%',
  buttonLevel: '%_BTNLEVEL%',
  buttonDescription: '%_BTNDESC%',
  gridPlaceholder: '%_GRID%',
};

```

```

export const clear = () => {
  elements.register.innerHTML = "";
  elements.login.innerHTML = "";
};

```

```

export const clearFields = () => {
  const fields = document.querySelectorAll('input');
  fields.forEach(el => el.value = "");
}

```

```

export const updatePattern = (color) => {
  const cur = document.querySelector(elementStrings.pattern).value
  const update = cur.concat(`${color}`);
  document.querySelector(elementStrings.pattern).value = update;
}

```

```

export const renderOne = (type) => {
  let markup = markups.one.replace(placeholders.buttonLevel, `${type}--one`);
  if (type === 'login') {
    markup = markup.replace(placeholders.description, replacements.oneLD);
    markup = markup.replace("%_retypepassword_", ``)
    elements.login.innerHTML = markup;
  } else if (type === 'register') {
    markup = markup.replace(placeholders.description, replacements.oneRD);
    markup = markup.replace("%_retypepassword_", `<div class="form__group">
    <input type="password" class="form__input" placeholder="Retype password"
    id="retype_password" minlength="6" required/>
    <label for="password" class="form__label">Retype password</label>
    </div>`)
    elements.register.innerHTML = markup;
  }
};

```

```

export const renderTwo = (type) => {
  let markup = markups.two.replace(placeholders.buttonLevel, `${type}--two`);
  if (type === 'login') {
    markup = markup.replace(placeholders.description, replacements.twoLD);

```

```

elements.login.innerHTML = markup;
} else if (type === 'register') {
markup = markup.replace(placeholders.description, replacements.twoRD);
elements.register.innerHTML = markup;
}
}

```

```

export const renderThree = (type) => {
let markup = markups.three.replace(placeholders.buttonLevel, `${type}--three`);
markup = markup.replace(placeholders.gridPlaceholder, Grid.create());
markup = markup.replace(placeholders.buttonDescription, type);
if (type === 'login') {
markup = markup.replace(placeholders.description, replacements.threeLD);
elements.login.innerHTML = markup;
} else if (type === 'register') {
markup = markup.replace(placeholders.description, replacements.threeRD);
elements.register.innerHTML = markup;
}
Grid.addImages();
}

```

```

export const renderFour = (type) => {
let markup = markups.four.replace(placeholders.buttonLevel, `${type}--four`);
if (type === 'login') {
markup = markup.replace(placeholders.description, replacements.fourLD);
elements.login.innerHTML = markup;
} else if (type === 'register') {

```



```

markup = markup.replace(placeholders.description, replacements.fourRD);
elements.register.innerHTML = markup;
}
}
export const renderFive = ()=>{
elements.login.innerHTML = markups.five
}

```

6.2 SERVER-SIDE CODING:

```

from turtle import title

from flask import Flask,request,url_for

from werkzeug.utils import secure_filename

from flask_cors import CORS

from flask_sqlalchemy import SQLAlchemy

import json,os,random

import smtplib, ssl

app = Flask(__name__)

CORS(app)

app.config["SQLALCHEMY_DATABASE_URI"] =
"mysql+pymysql://root:kapil@localhost/passwordauthapp"

app.config["UPLOAD_FOLDER"] = "static"

db = SQLAlchemy(app)

```

```

class User(db.Model):

    __tablename__ = "user"

    id = db.Column(db.Integer,primary_key = True)

    username = db.Column(db.String(100),nullable = False)

    textpassword = db.Column(db.String(100),nullable = False)

    colorpassword = db.Column(db.String(100),nullable = False)

    gridpassword = db.Column(db.String(100),nullable = False)

    answer = db.Column(db.String(100),nullable = False)


    @property

    def serialize(self):

        """Return object data in easily serializable format"""

        return {

            'id'      : self.id,

            'username'  : self.username,

            'textpassword' : self.textpassword,

            'colorpassword' : self.colorpassword,

            'gridpassword' : self.gridpassword,

            'answer'    : self.answer

        }

```

```

class Files(db.Model):

    __tablename__ = "files"

    id = db.Column(db.Integer,primary_key = True)

    userid = db.Column(db.Integer,nullable = False)

    title = db.Column(db.String(200),nullable = False)

    filename = db.Column(db.String(200),nullable = False)

    folder = db.Column(db.Integer,nullable = False,default = 0)

    private = db.Column(db.Boolean,nullable = False,default=0)


    @property

    def serialize(self):

        """Return object data in easily serializable format"""

        return {

            'id'      : self.id,

            'userid'   : self.userid,

            'title'    : self.title,

            'filename' : self.filename,

            'folder'   : self.folder,

            'private'  : self.private

        }

class Folders(db.Model):

```

```
__tablename__ = "folders"

id = db.Column(db.Integer,primary_key = True)

userid = db.Column(db.Integer,nullable = False)

folder_name = db.Column(db.String(50),primary_key = True)

folders = db.Column(db.Text,nullable = False,default = "{}")
```

```
@property
```

```
def serialize(self):
```

```
    """Return object data in easily serializable format"""
```

```
    return {
```

```
        'id'      : self.id,
```

```
        'userid'   : self.userid,
```

```
        'folder_name' : self.folder_name,
```

```
        'folders'   : self.folders,
```

```
    }
```

```
class Notification(db.Model):
```

```
    __tablename__ = "notifications"
```

```
    id = db.Column(db.Integer,primary_key = True)
```

```
    userid = db.Column(db.Integer,nullable = False)
```

```
    title = db.Column(db.String(100))
```

```
    message = db.Column(db.Text,nullable = False);
```

```

@property

def serialize(self):
    """Return object data in easily serializable format"""
    return {
        'id'      : self.id,
        'userid'   : self.userid,
        'title'    : self.title,
        'message'  : self.message,
    }

@app.route("/")
def index():
    return "Hello World"


@app.route("/api/adduser",methods=['POST'])
def adduser():
    try:
        data = request.data
        data2=json.loads(data)

        new_user = User(

```

```

username = data2["username"],
textpassword = data2["password"],
colorpassword = data2["pattern"],
gridpassword = data2["grid"],
answer = data2["answer"]
)

db.session.add(new_user)

db.session.commit()

return {"message": "success"}

except:

return {"message": "error"}


@app.route("/api/getalluser", methods = ["POST"])

def getAllUSers():

all_users = User.query

datalist = [i.serialize for i in all_users.all()]

return json.dumps(datalist)


@app.route("/api/addfile", methods = ["POST"])

def Addfile():

data = request.form

```

```
f = request.files.get("file")

file = Files(

userid = data.get("user_id"),

title = data.get("file_name"),

filename = secure_filename(f.filename)

)

db.session.add(file)

f.save(os.path.join(app.config['UPLOAD_FOLDER'], secure_filename(f.filename)))

db.session.commit()

return "200"
```

```
@app.route("/api/allfiles",methods = ["POST"])

def GetAllFiles():

data = json.loads(request.data)

files = Files.query.filter_by(userid = data["userid"])

file_list = [i.serialize for i in files.all()]

return json.dumps(file_list)
```

```
@app.route("/api/deletefile",methods = ["POST"])
```

```
def removeFile():  
  
    data = json.loads(request.data)  
  
    file = Files.query.filter_by(id = data["key"]).first()  
  
    os.remove(os.path.join(app.config['UPLOAD_FOLDER'], file.filename))  
  
    db.session.delete(file)  
  
    db.session.commit()  
  
    return "200"
```

```
@app.route("/api/addfolder",methods = ["POST"])  
  
def AddFolder():  
  
    for f in request.files.getlist("folder"):  
  
        print(f.filename)  
  
        file = Files(  
  
            userid = request.form.get("user_id"),  
  
            title = f.filename.split("/")[-1],  
  
            filename = secure_filename(f.filename.split("/")[-1])  
  
        )  
  
        f.save(os.path.join(app.config['UPLOAD_FOLDER'],  
            secure_filename(f.filename.split("/")[-1])))  
  
        db.session.add(file)  
  
        db.session.commit()
```



```
return "200"
```

```
@app.route("/api/privacy",methods = ["POST"])
```

```
def ChangePrivacy():
```

```
data = json.loads(request.data)
```

```
file = Files.query.filter_by(id = data["file_id"]).first()
```

```
file.private = not file.private
```

```
db.session.commit()
```

```
return "200"
```

```
@app.route("/api/preview",methods = ["POST"])
```

```
def preloadFile():
```

```
data = json.loads(request.data)
```

```
file = Files.query.filter_by(id = data["fileid"]).first()
```

```
return ("http://127.0.0.1:5000"+url_for('static', filename=file.filename))
```

```
@app.route("/api/resetuser",methods = ["POST"])
```

```
def ResetUser():
```

```
users = User.query.all()
```

```
for u in users:
```

```
db.session.delete(u)
```

```

db.session.commit()

return "200"


@app.route("/api/addnotification",methods = ["POST"])
def AddNotification():
    try:
        data = request.data
        data = json.loads(data)
        print(data)
        notification = Notification(
            userid = data["userid"],
            title = data['title'],
            message = data['message']
        )
        db.session.add(notification)
        db.session.commit()
        return "200"
    except:
        return "500"


@app.route("/api/getnotification",methods = ["POST"])
def GetNotification():

```

```

data = request.data

data = json.loads(data)

allnotification = Notification.query.filter_by(userid = data['userid'])

datalist = [i.serialize for i in allnotification.all()]

return json.dumps(datalist)

```

```

@app.route("/api/getonenotification",methods = ["POST"])

def getOneNotification():

data = request.data

data = json.loads(data)

notification = Notification.query.filter_by(id = data['id']).first()

data = notification.serialize

return json.dumps(data)

```

```

@app.route("/api/sendotp",methods = ["POST"])

def SendOTP():

data = json.loads(request.data)

reciever = data["email"]

seq = ["1","2","3","4","5","6","7","8","9","0"]

otp = "".join(random.choices(seq,k=5))

print("OTP=>",otp)

emaildata = json.load(open("./config/mail.config.json"))

```

```

sender,passw = [emaildata["email"],emaildata['password']]

print(sender,passw)

print(reciever)

port=587 # For starttls

smtp_server = "smtp.gmail.com"

sender_email = sender

receiver_email = reciever

password = passw

message = f"""\
Subject: OTP for Authentcation.

Your OTP is {otp}

"""

context = ssl.create_default_context()

with smtplib.SMTP(smtp_server, port) as server:

    server.ehlo() # Can be omitted

    server.starttls(context=context)

    server.ehlo() # Can be omitted

    server.login(sender_email, password)

    server.sendmail(sender_email, receiver_email, message)

    return otp

app.run(debug=True)

```

CHAPTER 7

SYSTEM TESTING

7.SYSTEM TESTING

The goal of testing is to find mistakes. Testing is the practise of attempting to find all possible flaws or weaknesses in a work product. It allows you to test the functionality of individual components, sub-assemblies, assemblies, and/or a whole product. It is the process of testing software to ensure that it works properly.

The software system meets the requirements and expectations of the users and does not fail in an unacceptable way. There are many different types of tests. Each test type is designed to fulfil a distinct testing need.

7.1 UNIT TESTING:

Unit testing entails creating test cases to ensure that the program's internal logic is working properly and that programme inputs result in valid outputs. Validation should be performed on all decision branches and internal code flow. It is the testing of separate software units of an application after they have been completed and before they are integrated. This is a type of structural testing. Unit tests ensure that each unique path of a business process follows the documented specifications and has clearly defined inputs and outputs.

7.2 INTEGRATION TESTING:

Integration tests are used to see if two or more software components can work together as a single application. Testing is mainly concerned with the basic output of screens or fields and is event driven. Integration tests show that, while the components were individually satisfying, the combination of components is proper and consistent, as demonstrated by successful unit testing. The component mix is accurate and consistent. Integration testing is a type of testing that focuses on uncovering issues that occur from combining components.

7.3 TEST CASES:

7.3.1 FUNCTIONAL TEST CASES:

S.NO	FUNCTIONAL TEST CASES	TYPE-POSITIVE/NEGATIVE TEST CASES
1.	Verify if a user will be able to login with a valid user name and valid password	positive
2.	Verify if a user cannot login with a valid username and invalid password.	Negative
3.	Verify the message for invalid login.	positive
4.	Verify if the “Enter key” of the keyboard is working correctly on the login page.	positive
5.	Verify if the font ,text color and color coding of the login page is as per the standard.	UI testing & positive

TABLE 7.1

7.3.2 SECURITY TEST CASES:

S.NO	SECURITY TEST CASES	TYPE- POSITIVE/NEGATIVE
1.	Verify if the users cannot enter the characters more than / less than the specified range(user name & password)	Positive
2.	Verify the login page by pressing “Back button” of the browser. It should not allow you to enter into the system once u logout.	positive
3.	Verify if the user should be able to login with the same credentials in different browsers at the same time	Positive

TABLE 7.2

7.3.3 UI TEST CASES:

Verify that all the labels and controls including text-boxes, buttons, and links are present on the Login page. Check that the font type and size of the labels and the text written on the different elements should be clearly visible. Verify that the size, color, and UI of the different elements are as per the specifications. Verify that the application’s UI is responsive i.e. it should adjust to different screen resolutions and devices.

CHAPTER 8

CONCLUSION

8.CONCLUSION

8.1 RESULT AND CONCLUSION:

Here, we are trying to make our authentication system more user friendly and also we have tried to implement mature & fast Shoulder Surfing Resistant Mechanism. We have considered both methods: text based and graphical based systems and tried to reduce the efforts required by end-user to remember passwords. A look at the advancement in technology over the past few years tells us that the next era will have system security at its core. Thus ,Graphical Password may be adapted in future as a major authentication system. Our system is a combination of recognition and recall based approach. It is more usable and secure as compare to previous graphical password authentication systems. As password space is very large it provides the security against brute force attack. It is easy to use. Passwords can be created and memorized easily Randomization in both the authentication steps provides strong security against shoulder surfing. Over all our system is resistant to all other possible attacks also. This system can be used for highly secure systems.

8.2 FUTURE ENHANCEMENT:

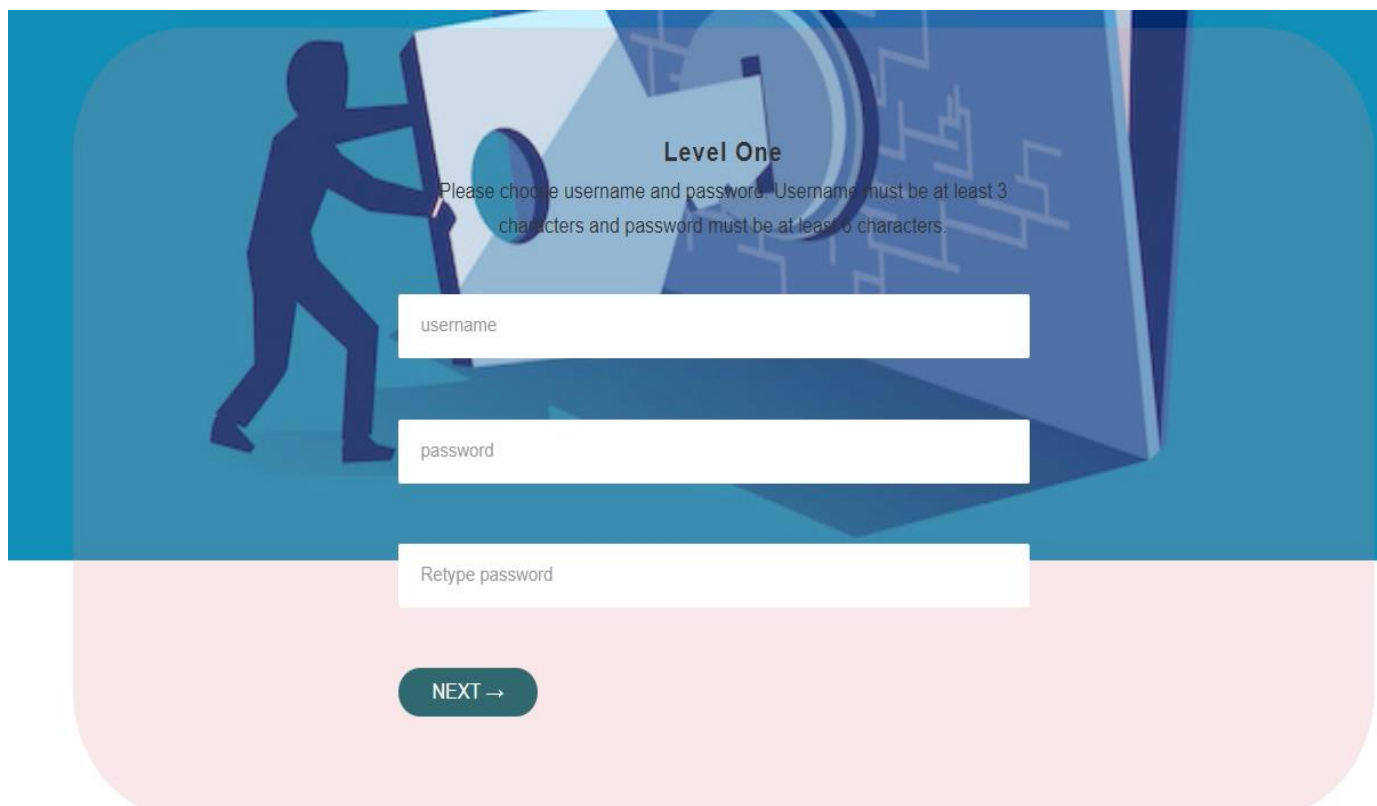
In future, one more addition possible to our system is, if the user forgets any password that password is mailed to user's registered mail id and such a message will be sent to user's registered mobile number also. So user can get the system updates although he is offline. Thus, in future, our system can be made more secure and easy to access.

APPENDICES:

A1.SAMPLE SCREENSHOTS:



FIG A.1



Level One

Please choose username and password. Username must be at least 3 characters and password must be at least 6 characters.

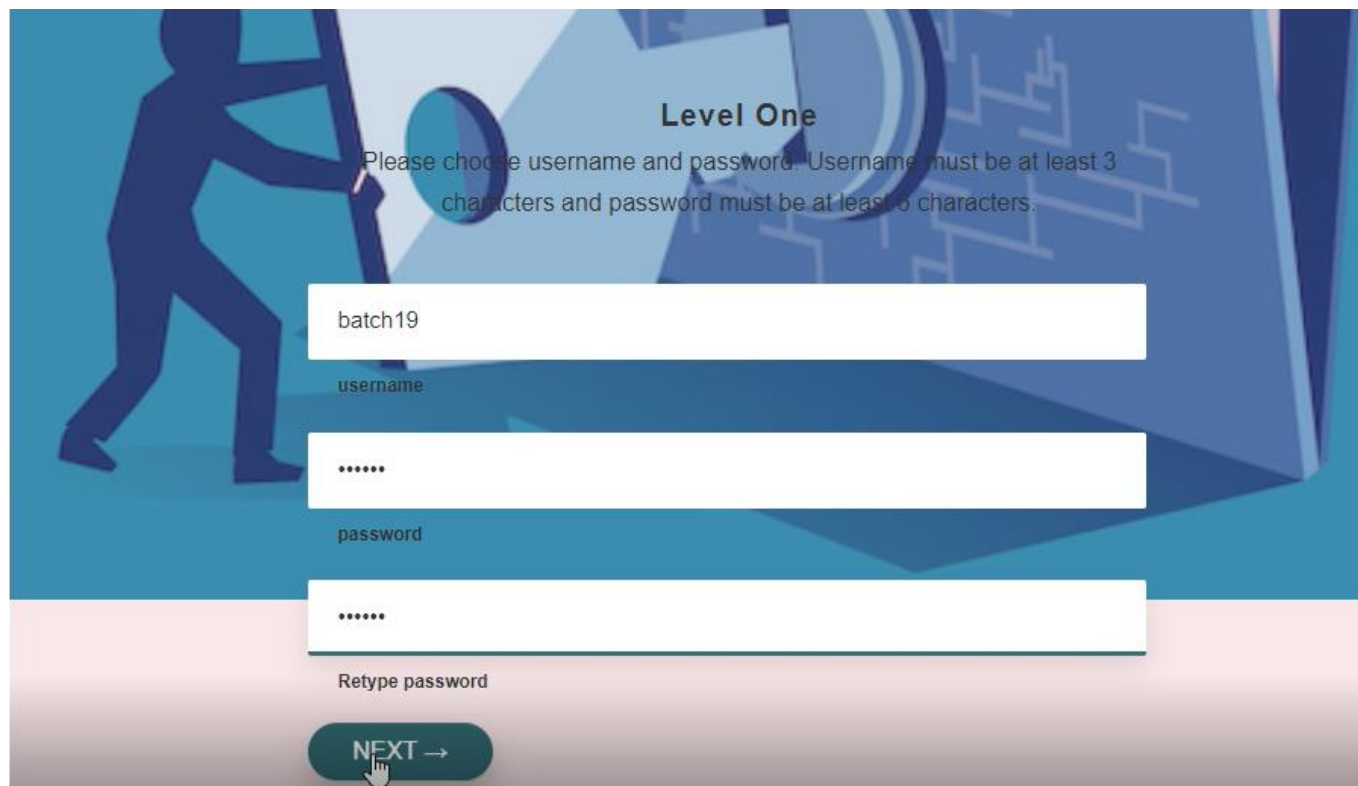
username

password

Retype password

NEXT →

FIG A.2



Level One

Please choose username and password. Username must be at least 3 characters and password must be at least 6 characters.

batch19

username

.....

password

.....

Retype password

NEXT →

FIG A.3

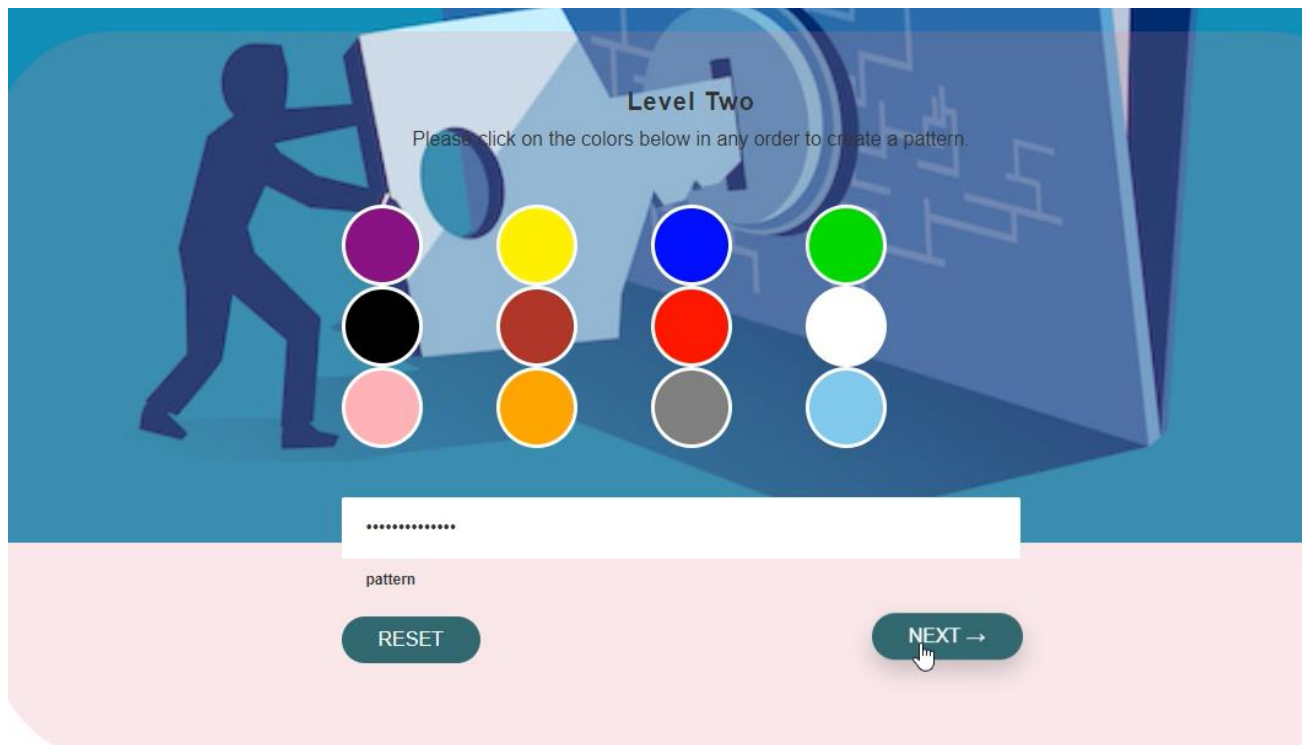


FIG A.4

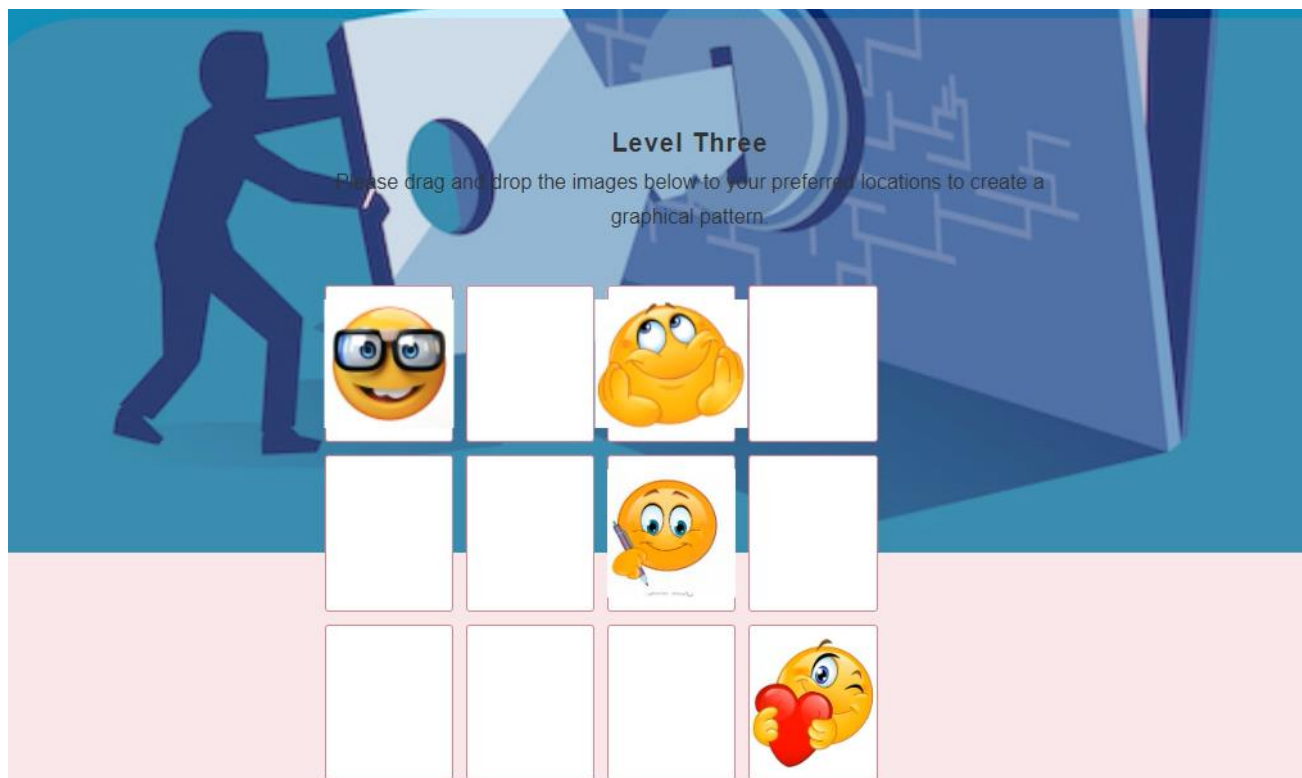


FIG A.5

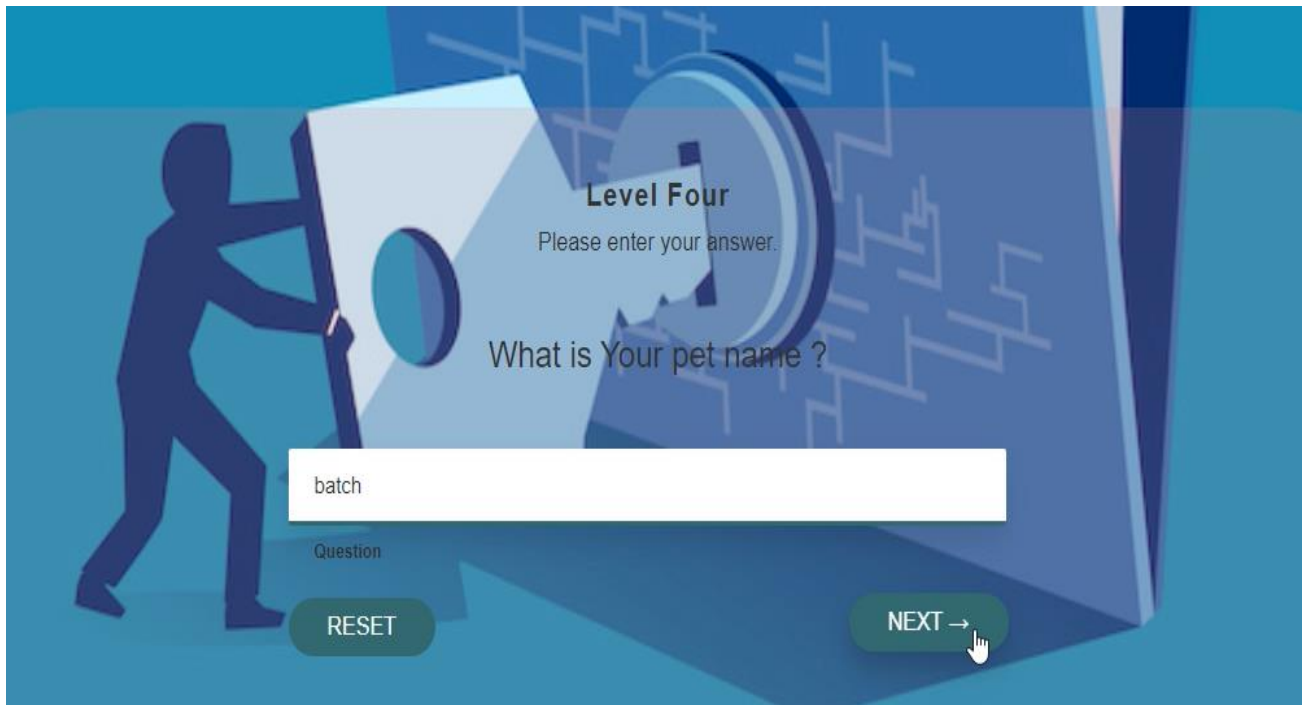


FIG A.6

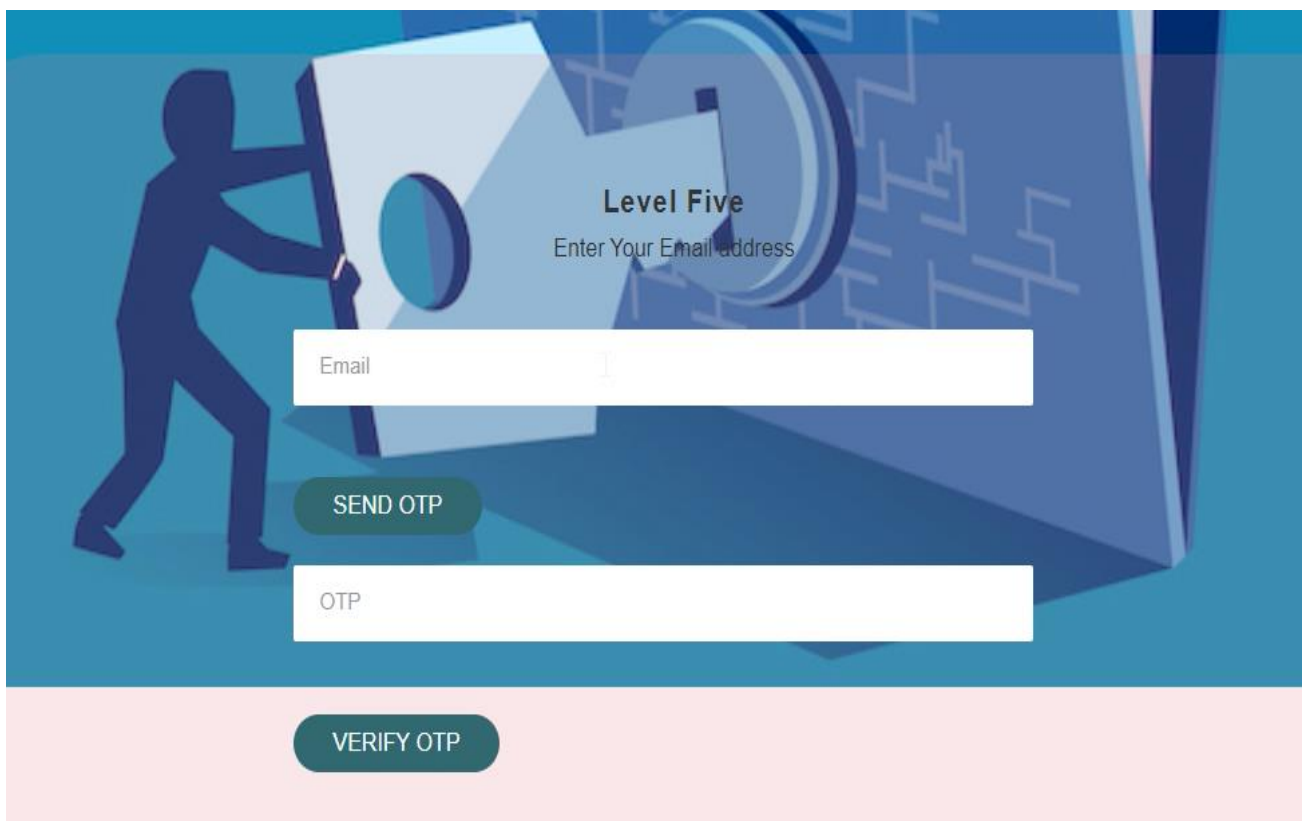


FIG A.7

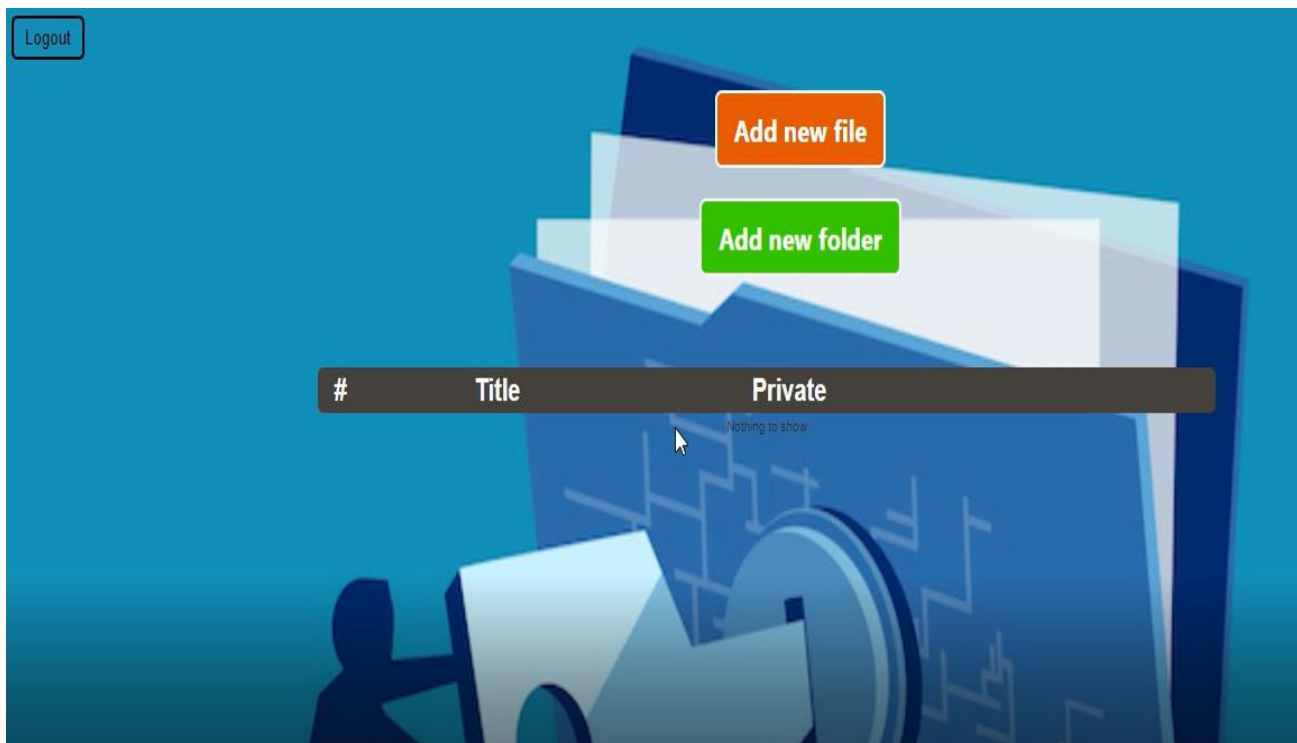


FIG A.8

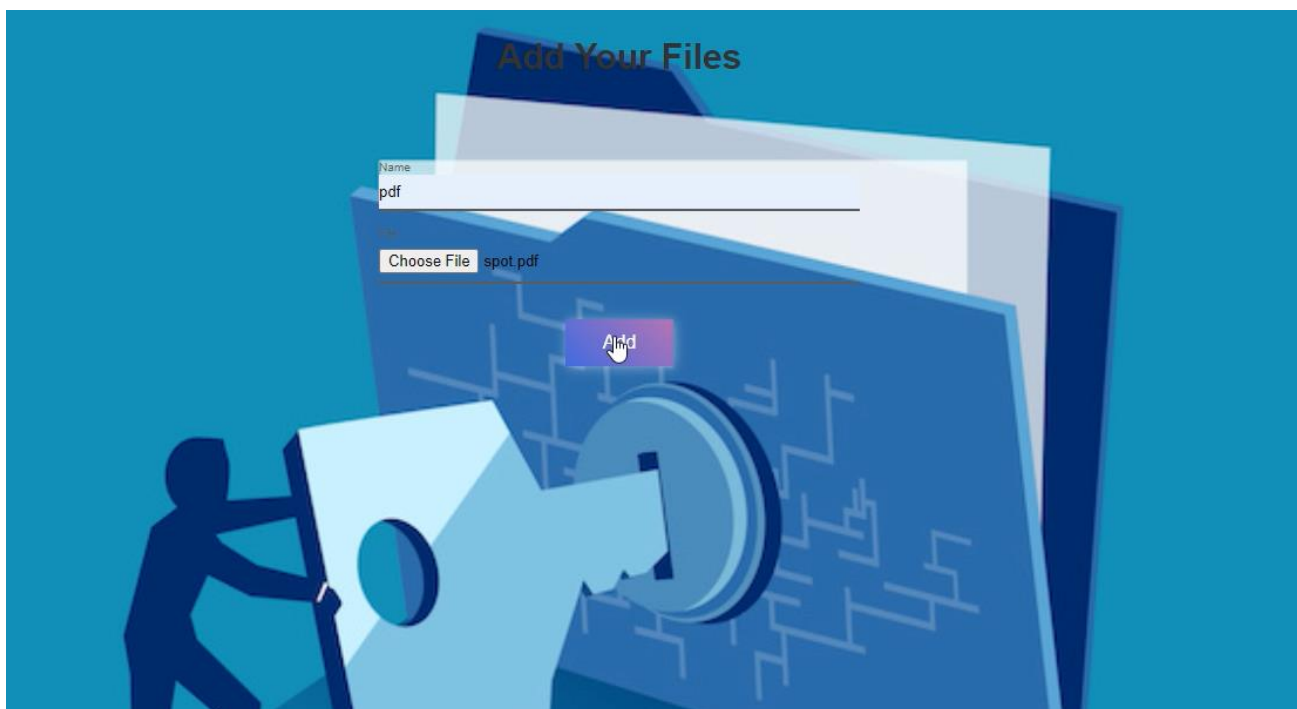


FIG A.9

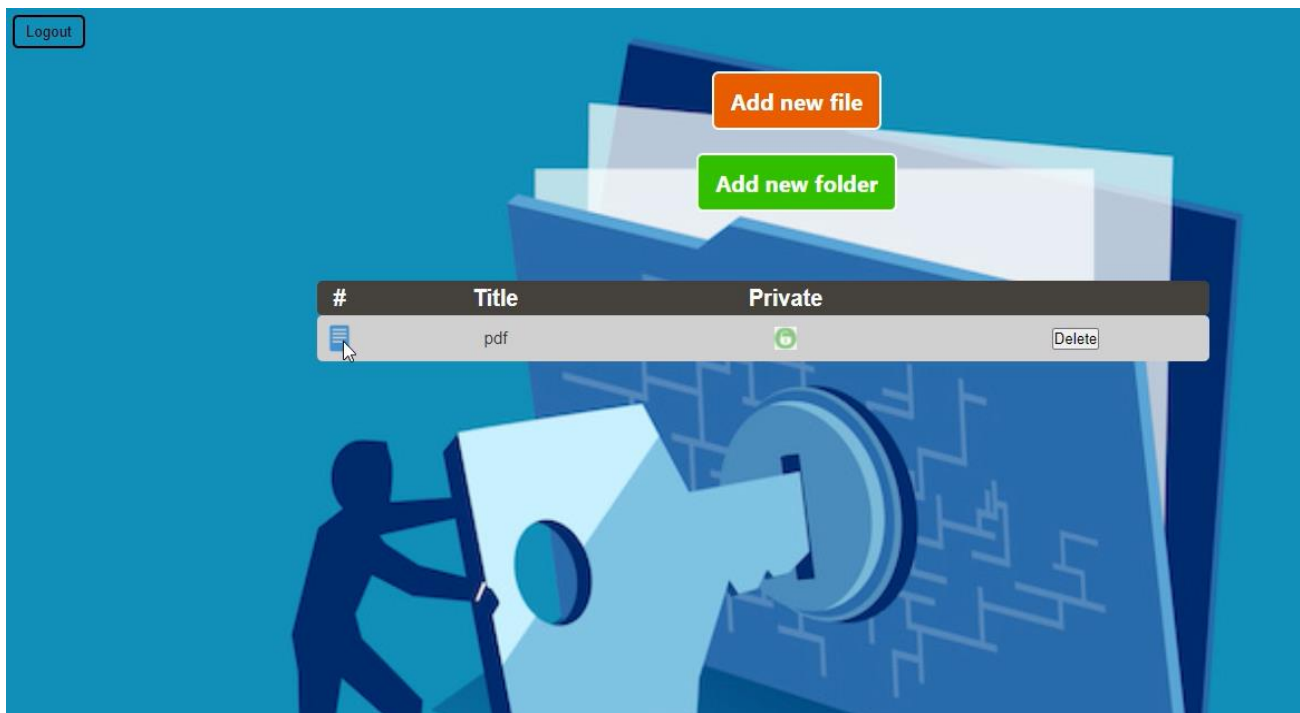


FIG A.10

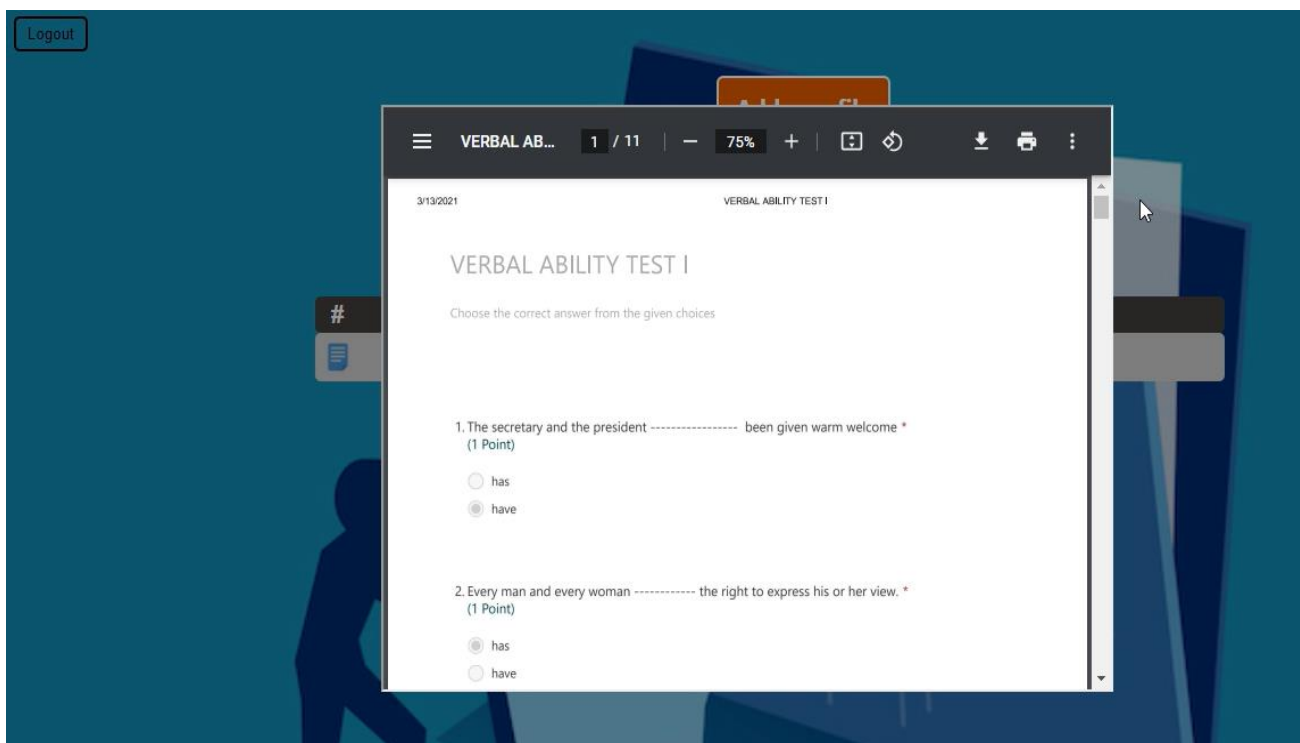


FIG A.11

REFERENCES:

- [1] A. H. Lashkari, S. Farmand, D. Zakaria, O. Bin, and D. Saleh, Year 2009, "Shoulder surfing attack in graphical password authentication", International Journal of Computer Science and Information Security, Volume . 6, Page. 145-154.
- [2] M. V. Ruiz-Blondet, Z. Jin, and S. Laszlo, Year 2016, "Cerebre: A novel method for very high accuracy event-related potential biometric identification", IEEE Transactions on Information Forensics and Security, Volume . 11, no. 7, Page 1618–1629,.
- [3] J. M. Mandler and G. H. Ritchey, Year: 1977 "Long-term memory for pictures", Journal of Experimental Psychology: Human Learning and Memory, Volume. 3, no. 4, Page 386.
- [4] J. Yan, A. Blackwell, R. Anderson and A. Grant, Year 2004, "Password Memorability and Security: Empirical Results," in Proceedings of IEEE Security & Privacy, Volume . 2, Page. 25–31.
- [5] Bin B. Zhu, Jeff Yan, Guanbo Bao, Maowei Yang, and Ning Xu, Year 2014, "Captcha as Graphical Passwords A New Security Primitive Based on Hard AI Problems," in Proceedings of the IEEE Transactions on Information Forensics and Security, Volume. 9, Page. 891–904.
- [6] Title : Doodle-Based Authentication Technique Using Augmented Reality. Author: Waqas wazir , Hasan ali khattak , (senior member, ieee), Ahmad almogren , (senior member, ieee), Mudassar ali khan and Ikram ud din , (senior member, ieee), Year: 2019, Journal name: IEEE.
- [7] Title : A Secure Hybrid Authentication Scheme using PassPoints and Press Touch Code. Author: Saiful azad1, Noor elya afiqah che nordin , Nur nadhirah ab rasul , Mufti mahmud, (senior member, ieee), Kamal z. zamli1 , (senior member, ieee), Year: 2019, Journal name: IEEE.
- [8] Title : Authentication using text and graphical password. Author: Abhilash M Joshi, Balachandra Muniyal. Year: 2018, Journal name: IEEE .
- [9] Title : A Secure and User-Friendly Graphical Password Scheme. Author: Gi-Chul Yang, Year: 2017, Journal name: IEEE.
- [10] Title : A Cued-Recall and Emotion Classification Graphical Password Authentication Scheme . Author: Danilo E. Vieira1 , Tonny L. Mesquita Abreu1 , Max E. Vizcarra Melgar1 , Luz A. M. Santander2 , Year: 2017, Journal name: IEEE.