TEAM 47

Graphical Password and Authentication System

GUIDE: Dr. Muneeswaran V

TEAM MEMBERS

AYUSH TIWARI (20BCY10139)

OMKAR VILAS NADE (20BCY10227)

ANKIT KUMAR OJHA (20BCY10196)

MADHUR MUNOT (20BCY10175)

INTRODUCTION

Human factors are often considered the weakest link in a computer security system. If we point out that there are three major areas where human-computer interaction is important: authentication, security operations, and developing secure systems. Here we focus on the authentication problem. User authentication is a fundamental component in most computer security contexts. Studies showed that since user can only remember a limited number of passwords, they tend to write them down or will use the same passwords for different accounts.

To address the problems with traditional username-password authentication, alternative authentication methods, such as biometrics, have been used. In this paper, however, we will focus on another alternative: using image as passwords.

In some graphical password schemes, the system must retain knowledge of some details of the shared secret, i.e., user specific profile data e.g., in recognition schemes, the system must know which images belong to a user’s portfolio to display them. This information must be stored such that its original form is available to the system (possibly under reversible encryption), and thus may be available to anyone gaining access to the stored information. E.g., Phishing attack and shoulder surfing attack.

PROBLEM STATEMENT

In this project, we are developing a graphical password schemes, in which the system retains knowledge of some details of the shared secret, i.e., user specific profile data e.g., in recognition schemes, and knows which images belong to a user’s portfolio to display them.

Existing Work with Limitations

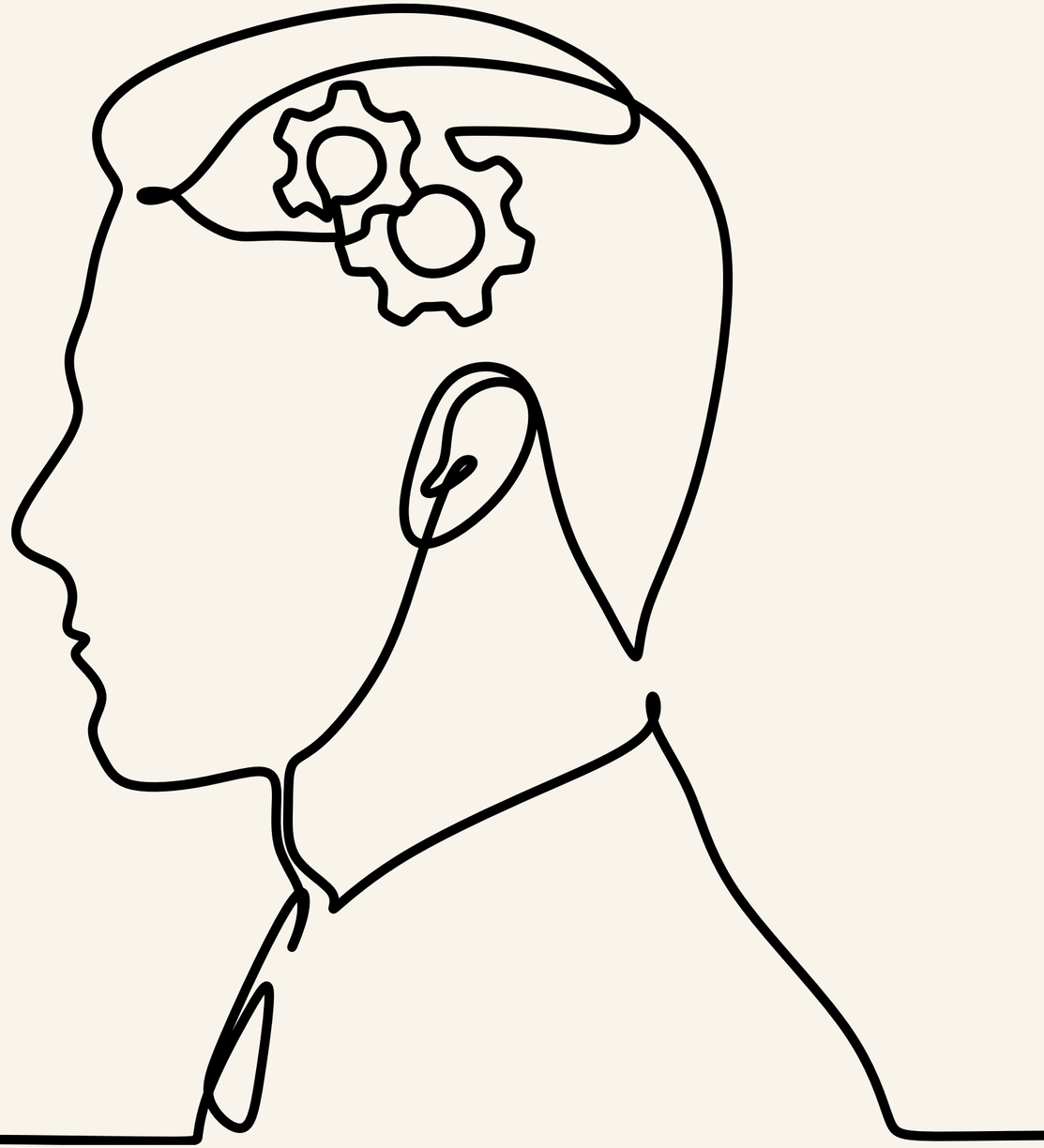
Currently this method is used by dividing a single image into multiple segments and the user selects some specific pattern from it in order to choose for his/her authentication.

Limitations:-

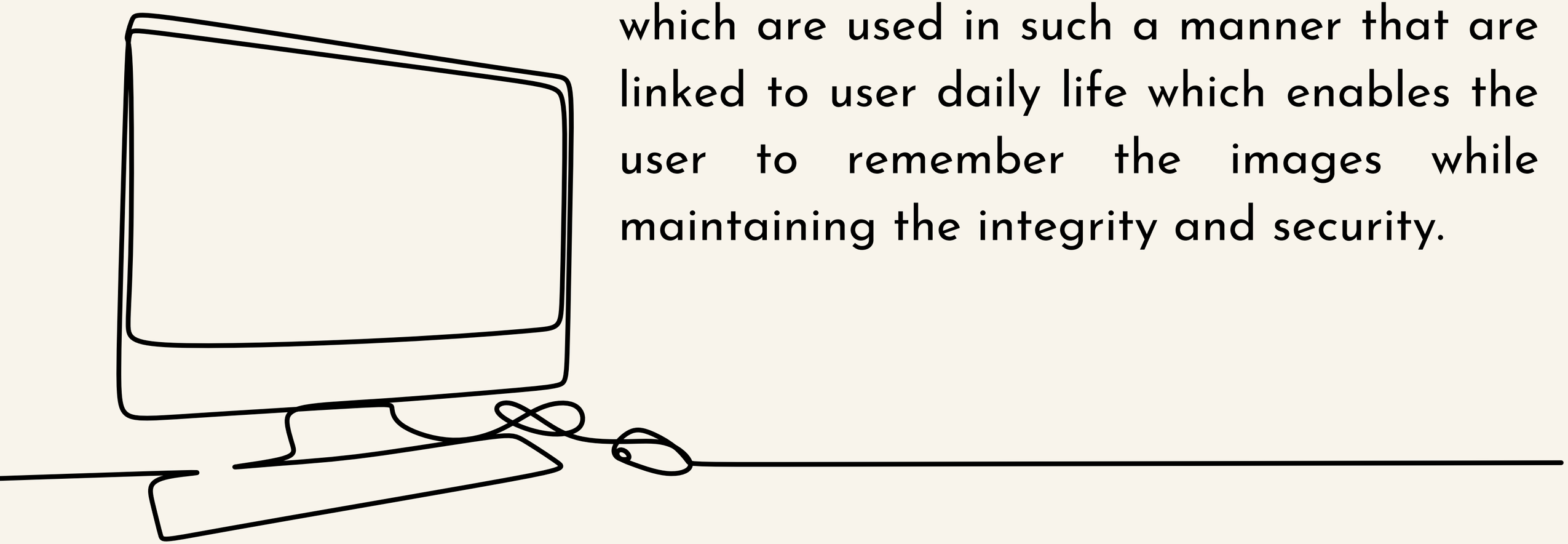
Current method imposes threats of brute force as image segmentation uses pixel destruction method which can be easily broken out.

PROPOSED WORK

The project aims to improve security of users' credentials on various social networking sites and other platforms by proposing the concept of graphical password authentication system. In this, the users will have to select a certain sequence of some provided images from various image categories to set their profile password at the time of registering. The sequence will be kept in a database in encrypted format and will be matched each time the user tries to login in their account.



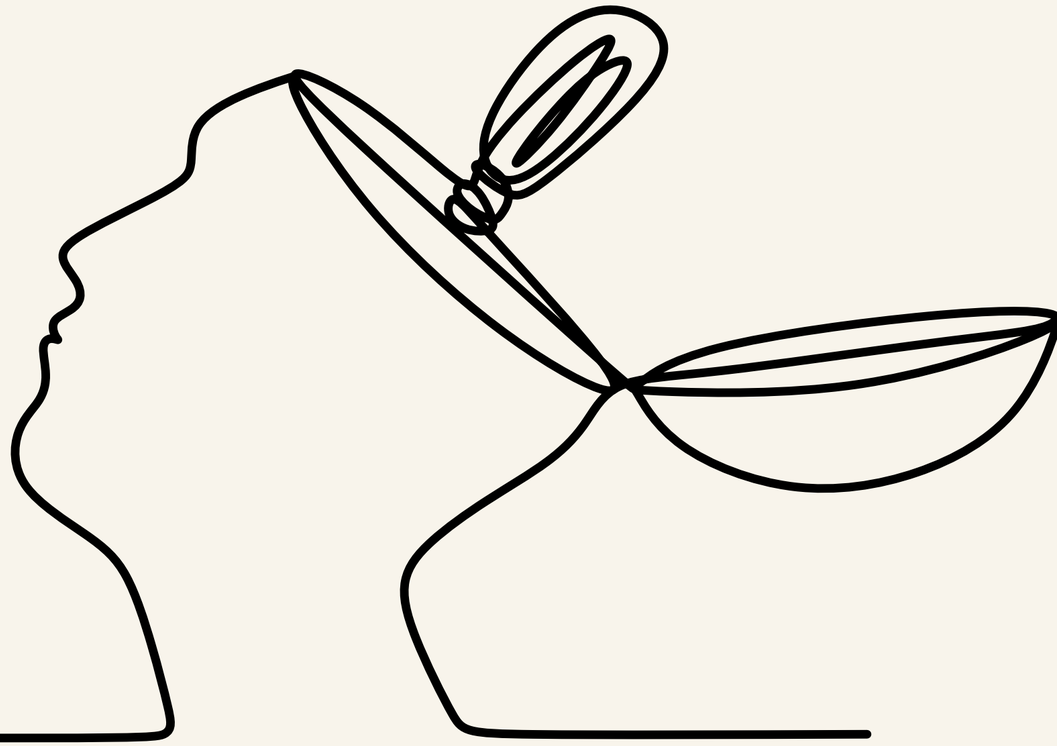
NOVELTY OF THE PROJECT

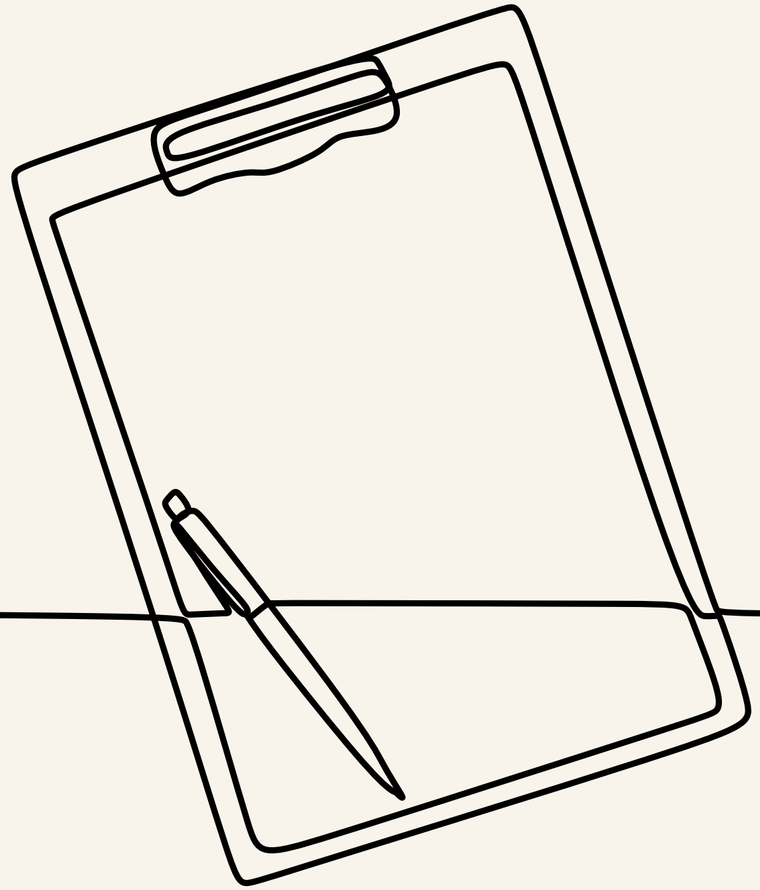


The project enables user to select the individual images from the given images, which are used in such a manner that are linked to user daily life which enables the user to remember the images while maintaining the integrity and security.

REAL TIME USAGE

This type of authentication system can be used high grade security areas where the access is restricted to a limited no. Of people. It can also be used as for authentication of normal applications like chatting apps, net banking etc. In a manner by reducing no. Of images and increasing minimum number of images.



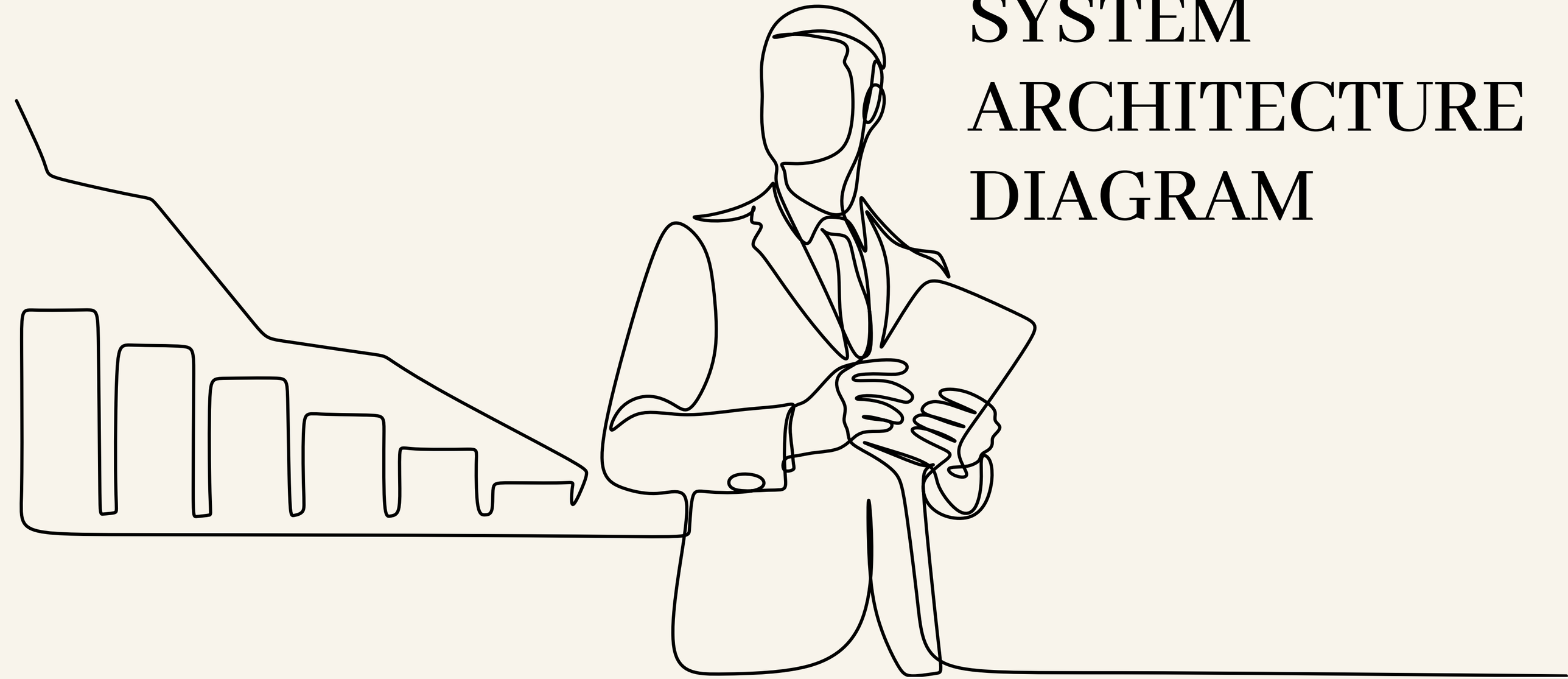


HARDWARE AND SOFTWARE REQUIREMENTS

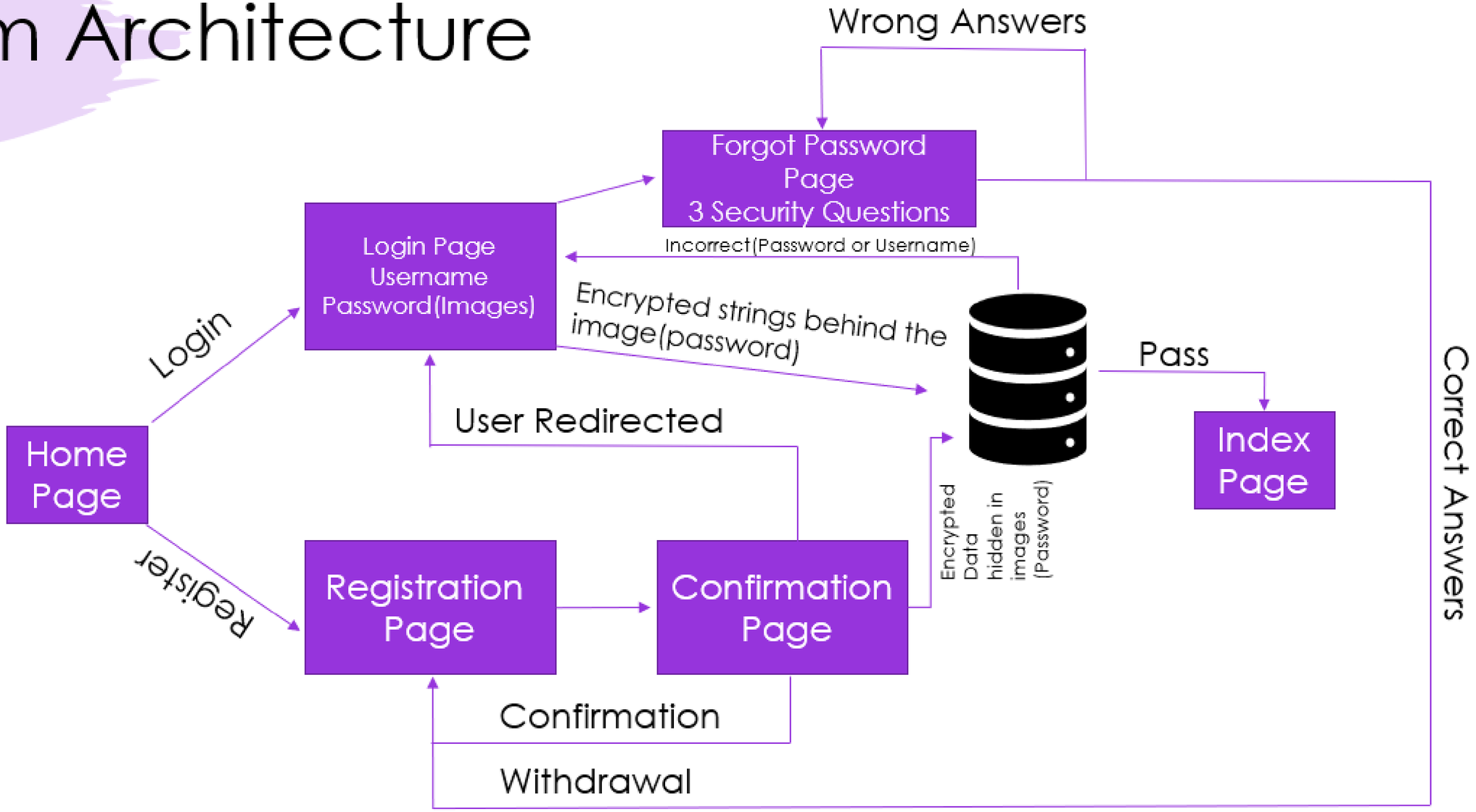
- Operating System – Windows 10 or higher, Ubuntu 18.4+,Debian 10+,OS X El Capitan 10.11 +

- Ram – 4 GB or Higher
- Browser – Chrome, Mozilla Firefox, Microsoft Edge, Safari
- Software – MongoDB, Apache Server

SYSTEM ARCHITECTURE DIAGRAM



System Architecture



LITERATURE REVIEW

The current image authentication system uses segmentation method in which a single image is divided into segments of 6 or 8 and user must rearrange that segments in order that was chosen at time of registration. But the problem with the current system is that no. Of combinations is too less which enable hacker to guess combination very easily.

For example: If the image is divided in segment of 8. It will result a shuffled grid of segmented images of size 4*4. Which result in the no. Of possible combinations 16^4 = 65536 which is to less.

MODULES

•**HTML** - The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser.

•**CSS** - Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.

•Flask- Django is a web development framework that assists in building and maintaining quality web applications. Django helps eliminate repetitive tasks making the development process an easy and time saving experience

•MongoDB - MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

MODULE WORKFLOW

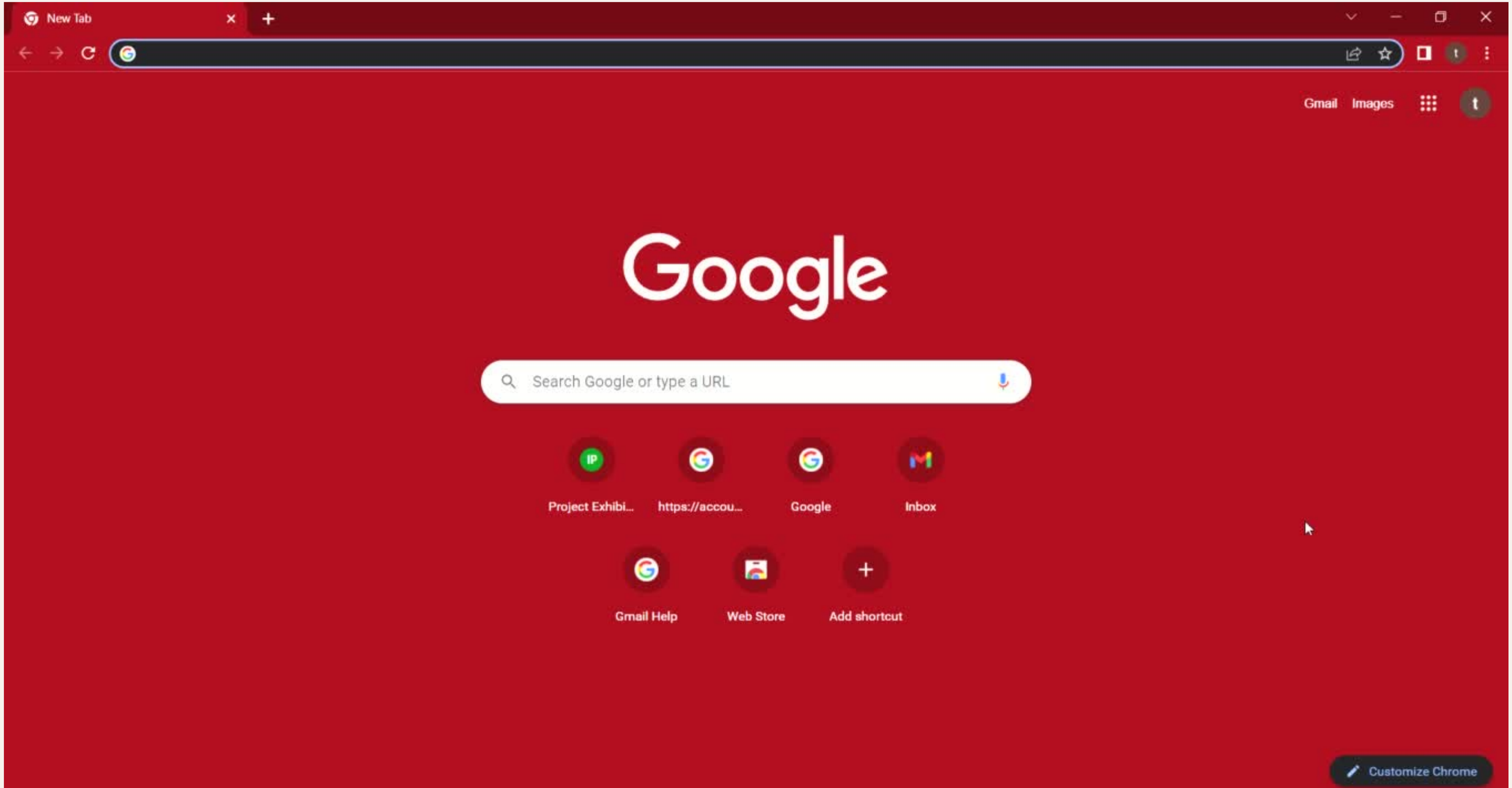
Implementation and coding

```
from flask import Flask, render_template, request, flash, redirect
import codecs
from flask.json.tag import PassDict
from flask_mail import Mail,Message
from random import randint

from flask.templating import render_template_string
import pymongo

# Create an instance of the Flask class that is the WSGI application.
# The first argument is the name of the application module or package,
# typically __name__ when using a single module.
app = Flask(__name__)
app = Flask(__name__, template_folder='html')
myclient = pymongo.MongoClient("mongodb+srv://rmaini:rajatmai@cluster0.a1te0.mongodb.net/myFirstDatabase?retryWrites=true&w=majority")
mydb = myclient["testing"]
mycol = mydb["testing"]
mail = Mail(app)
app.config["MAIL_SERVER"]='smtp.gmail.com'
app.config["MAIL_PORT"]=465
app.config["MAIL_USERNAME"]='xyzasak@gmail.com'
app.config['MAIL_PASSWORD']='Amit@2020'                    #you have to give your password of gmail account
app.config['MAIL_USE_TLS']=False
app.config['MAIL_USE_SSL']=True
mail=Mail(app)
otp=randint(000000,999999)
```

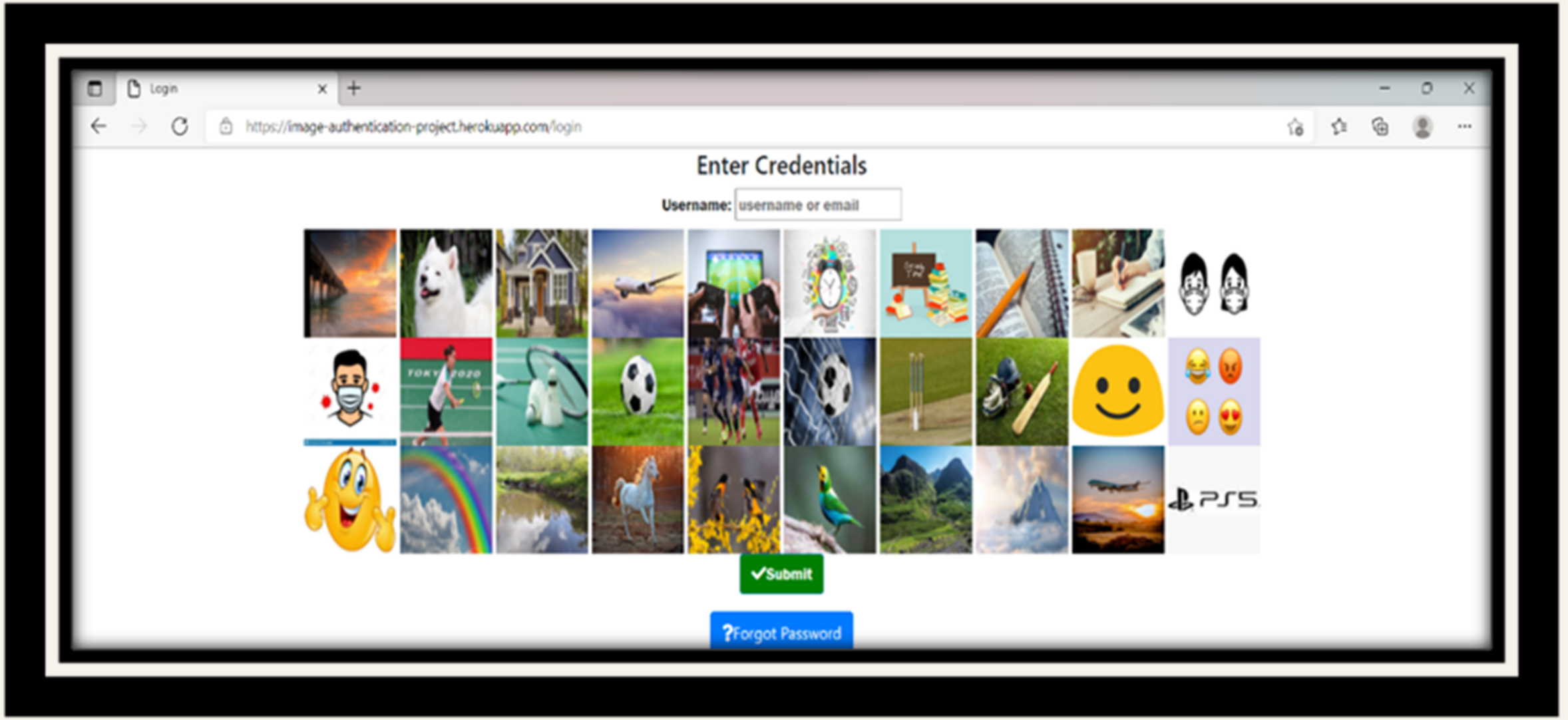
Implementation and coding

```
<!DOCTYPE html>
<link href="https://fonts.googleapis.com/css2?family=Bebas+Neue&family=Merienda&display=swap" rel="stylesheet">
<link href="https://fonts.googleapis.com/css2?family=Bebas+Neue&family=Merienda&family=Roboto:wght@900&display=swap" rel="stylesheet">
<link href="https://fonts.googleapis.com/css2?family=Bebas+Neue&family=Courgette&family=Merienda&family=Patrick+Hand&family=Roboto:wght@9
    <style>
        h1 {
            background-color: #B266FF;
            text-align: center;
            font-style: bold;
            font-size: 50px;
            font-family: Merienda;
        }
        #b1, #b2 {
            font-size: 45px;
            background-color: #04AA6D;
            border-radius: 15px;
            padding: 10px;
            cursor: pointer;
            box-shadow: 5px 10px rgba(0,0,0,0.19);
        }
        #b1:hover, #b2:hover {
            background-color: #3e8e41;
        }
        .button {
            text-align: center;
            margin: auto;
        }
        h2, h3 {
            font-style: bold;
            text-align: center;
            font-size: 50px;
            font-family: Roboto;
        }
```

Demo Video


New Tab
Gmail
Images
Google
Search Google or type a URL
Project Exhibi...
https://accou...
Google
Inbox
Gmail Help
Web Store
Add shortcut
Customize Chrome

MODULE IMPLEMENTATION AND DEMO




Final page
Bhopal (VIT)
https://image-authentication-project.herokuapp.com/final
Welcome,test

https://image-authentication-pr

https://image-authentication-project.herokuapp.com/11

Username: test

Select image combinations

Proceed

REFERENCES

- https://devdocs.io/html/
- https://devdocs.io/css/
- https://devdocs.io/react/
- https://devdocs.io/redux/
- https://docs.djangoproject.com/en/3.2/
- http://httpd.apache.org/docs/current/
- https://docs.mongodb.com/