



**SMART INDIA
HACKATHON
2022**

Graphical Password Authentication

[DR705]

by – Team Eldians

**for – All India Council of Technical
Education (AICTE)**

Team Leader – Harsh Jannawar

**Members – Abhijeet Pawar, Anish Zaveri, Kshitij Jande,
Sachin Rai, Shruti Makhwan**

Institute – Symbiosis Skills and Professional University

Technology Bucket – Block chain & Cybersecurity

Date – 26/03/2022

Problem Statement –

Passwords are ubiquitous today on any platform, on possibly any website. But to remember so difficult passwords and that too on numerous websites seems daunting and therefore you can devise a project illustrating graphical password strategy. This will allow the user to set passwords in the form of graphical presentation in a certain pattern and later use that pattern to login o the system. Summary: Remembering numerous passwords from various different sites can be difficult for a user. So to provide some flexibility we can provide users a graphical password authentication system where instead of creating a password a user has to select graphical objects in a particular order to keep it as their password.

Objective: : In this method, the user is required to select some images (let's say different chocolates) in a specific pattern (for example dairy milk is followed by 5 stars which is in turn followed by KitKat and so on). Next time the user tries to log in, the images would have been shuffled, but the user will be required to follow the same pattern which was used initially. Every time the user will have to use the same sequence while the images are placed in different ways. This type of authentication is difficult to break since neither brute force nor dictionary attacks could breach it. We need techniques that can be easily implemented and provide better results to this process.

Abstract -

People are very used to text based passwords. Users usually keep these as phrases or words from their memory. These can be easily obtained through means of social engineering. When it comes to creating strong passwords, you need to create it with a combination of various alphanumeric and non-alphanumeric characters. These passwords are usually lengthy and including non-alphanumeric characters makes it harder for the user to remember them. This is where graphical passwords come into play by providing same if not better level of security while also making it easier for user to remember passwords.

Designed Solution -

The solution provided by our team goes through complex steps to ensure the integrity of the passwords are maintained while also making it easy for the user to remember his password.

The following are the steps to an user needs to follow

1) Registering an user –

- User is taken to a window similar to *Figure 1*, where he has to choose 5 images from given 4 categories.
- Each category consists of 25 images each.
- The user is given entire freedom of how he wants to choose images. He can chose multiple images of same category or even the same images multiple times.

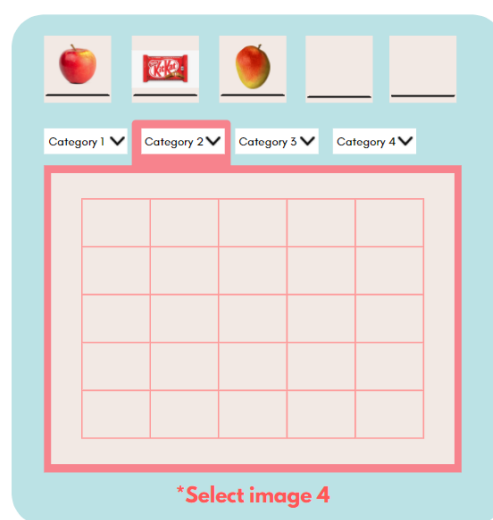


Figure 1

2) Storing the password

First we'll like to explain how the backend works. There are 2 databases,

- i) First stores the total of 100 images along with their identification ID's
- ii) Second stores a user's ID and password. The password is stored in form of a String created from the image ID's in the order the user selected the images

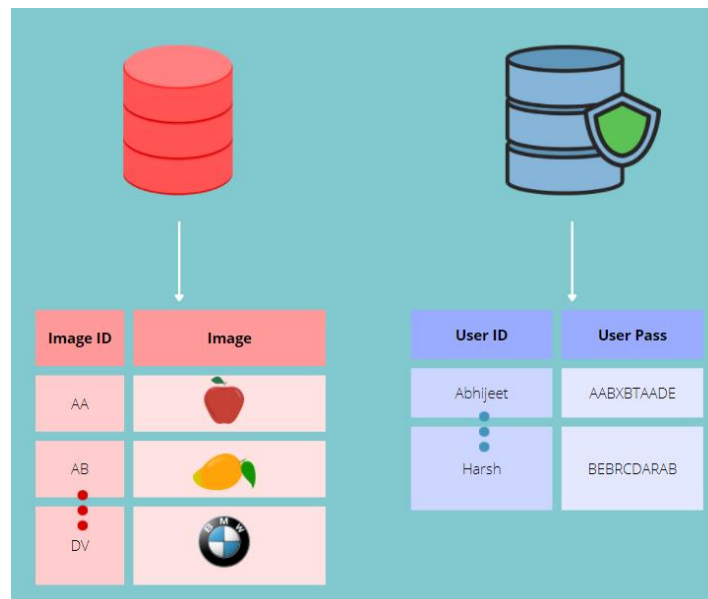


Figure 2

3) a) Login phase (Frontend)

The user is presented with a screen similar to what is shown in *Figure 3*. The user has to recall his sequence of images and identify them from the given set of 25 images in the 5 by 5 grid.

To enter the password, user has to enter the number corresponding to their images in the matrix in the sequence they have provided.

Upon pressing the login button, the information is encrypted and relayed to server to verify and authenticate the user. If proper password is entered, user is authenticated or else user is provided with another set of shuffled images.

*Enter the numbers corresponding to your password images in the sequence you provided

01	02 	03	04	05
06	07	08	09	10
11	12	13 	14	15
16	17	18	19	20
21	22	23	24	25


SUBMIT 

Figure 3

3) b) Login phase (Backend)

On the backend side, the server recognizes a request has been made by a user. It then follow the following steps –

- 1) Retrieve the user's password from Password database.
- 2) Separate the string and get individual image ID's
- 3) Retrieve this images from their ID's from the Images database along with other random images to fill up the 25 by 25 grid
- 4) Shuffle these images in a 5 by 5 matrix and assign them temporary ID's according to their position in the grid
- 5) The user's password images are noted with their temporary ID and a big integer is formed which will later help with verification
- 6) The 5 by 5 grid is securely sent to the user so that he can enter his temporary password with help of images
- 7) This password is sent back to the server where it is compared to the session password for authentication

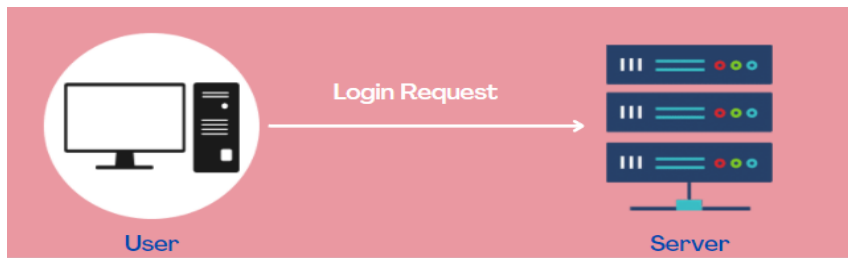


Figure 4-1

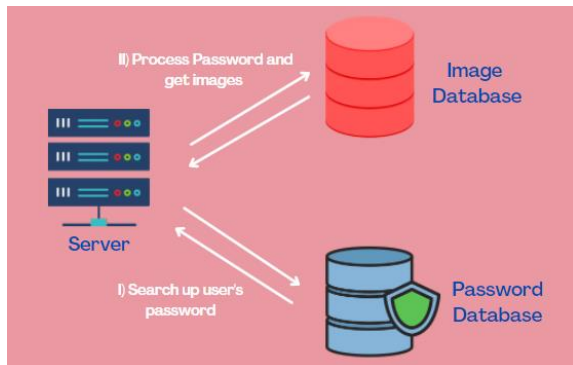


Figure 4-2

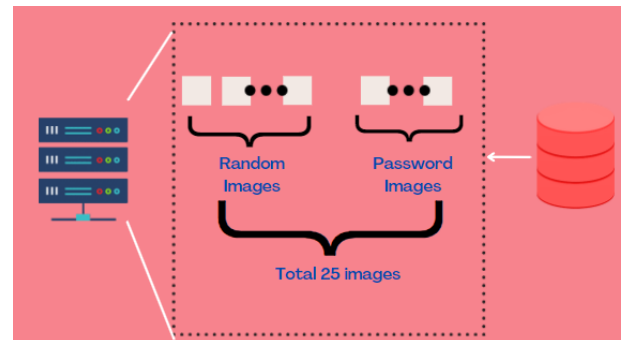


Figure 4-3

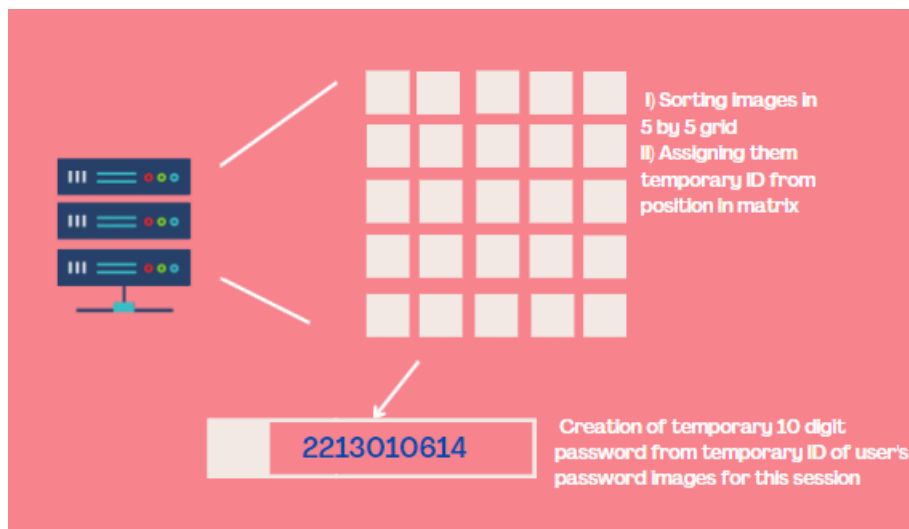


Figure 4-4

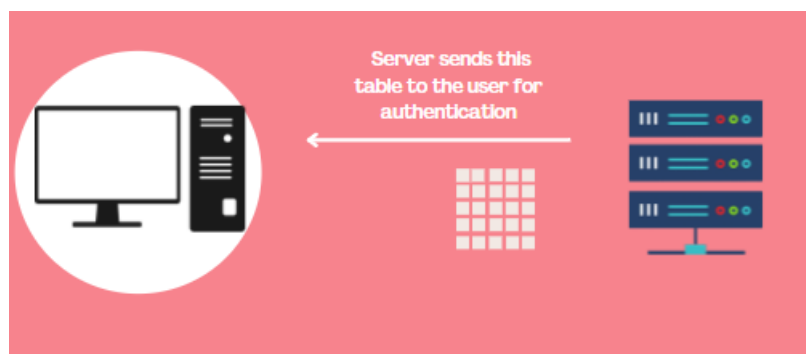


Figure 4-5

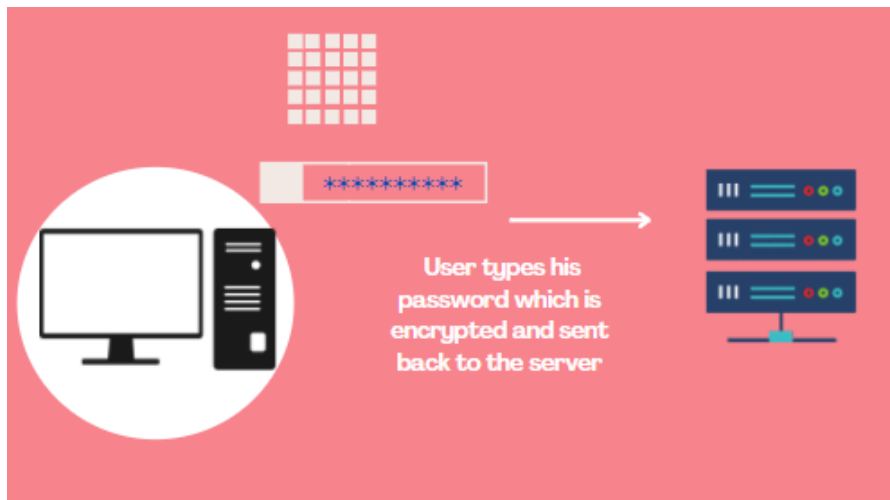


Figure 4-6

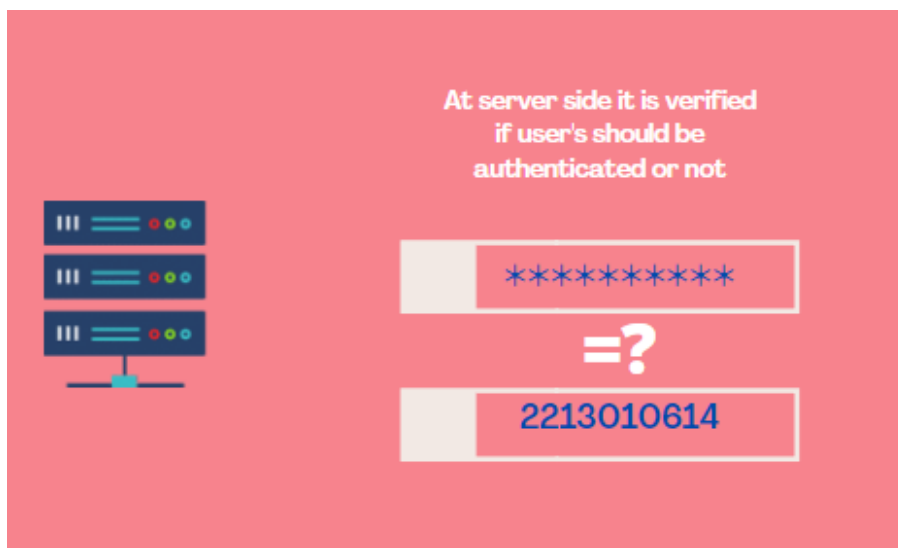


Figure 4-7

Technology Stack –

- 1) NodeJS (Express): For the backend processing and designing interactive elements in the webpage
- 2) HTML/CSS: For designing the form and the Authentication section on a webpage
- 3) MongoDB: For storing of images in the database as well as storing passwords securely of another database

Analysis –

Our current 'strong' text passwords consist of 8 characters which are both alphanumeric and non-alphanumeric characters. They have certain rules like compulsory inclusion of a lowercase and uppercase character, a number and a special character. If you calculate the total number of possibilities for this type of password for length 5 –

- i) 26 Alphabets
- ii) 10 digits
- iii) 33 special characters

$$(26) \times (26) \times (10) \times (33) \times (95) \\ = \mathbf{21.1 \text{ million combinations}}$$

On the other hand our solution provides user with possibilities to choose from 100 images without any restriction at all, thus the total number of combinations for our solution for same length is –

$$(100) \times (100) \times (100) \times (100) \times (100) \\ = \mathbf{10 \text{ billion combinations}}$$

That's **473** times the number of possibilities while also making it easier for the user to remember his password.

The idea behind using numpad for input -

Using numpad will help us prevent from shoulder surfing attacks as the on-looker has to keep track of what numbers the user is typing as well the images on screen. Since the images on screen are shuffled every time, it gets harder for a simple on-looker to keep track of the password being entered.

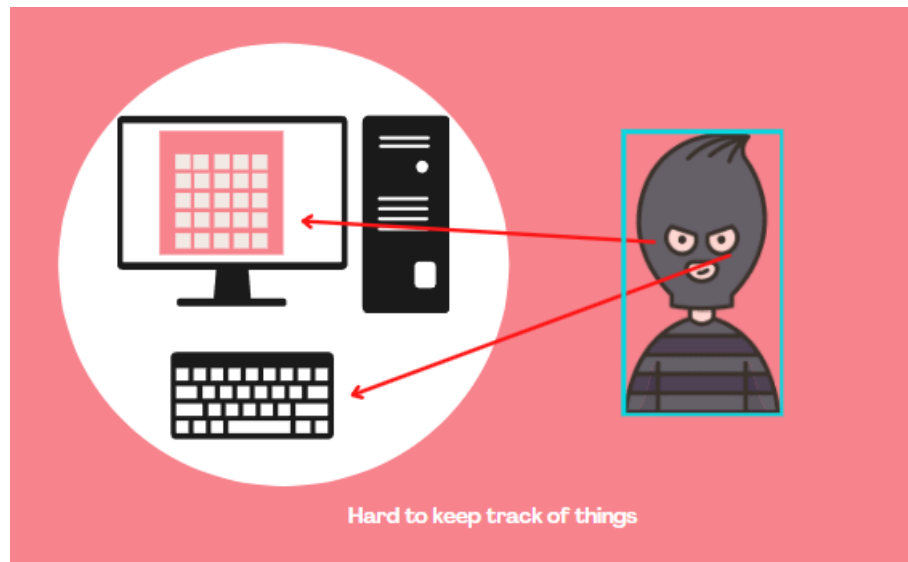


Figure 5

Conclusion –

We would like to conclude with saying that we have designed a viable solution which is in the very early stages, Of course this can be improved upon further as the development takes place. We have been able to prove that GPA might take over conventional text based passwords for their easy remembrance and better security options.