# Idea/Approach Details

**Ministry/organization name:** **AICTE(All India Council for Technical Education)**

**Problem Statement:** **Graphical Password Authentication**

**Team Name:** **CYBER WIZARDS**

**Team Leader Name:** **N Rishi Raj Reddy**

**College Code:** **WSTM**

## INTRODUCTION:

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 users to create passwords in the form of a graphical presentation in a specific pattern, which they can then use to login to the system. **Summary**: It can be tough for a user to remember multiple passwords from various websites. So, instead of setting a password, we can present users with a graphical password authentication system in which they must choose graphical objects in a specific order to maintain it as their password. The goal is to: In this strategy, the user must select a set of images (for example, different chocolates) in a specified order (for example, dairy milk is followed by 5 stars, which is followed by KitKat). The photos will have been shuffled the next time the user tries to log in, but the user will be expected to follow the same procedure as before. The user will have to utilize the same sequence each time, even when the images are arranged differently. This sort of authentication is tough to crack since it is resistant to both brute force and dictionary assaults. We require strategies that are simple to use and yield better results in this process.

## PROPOSED SOLUTION:

Here we develop a web-based application that uses **Graphical Password Authentication**. There will be two modules of user authentication. The first module will be the **Registration phase** in which the user needs to fill in their basic details like name, phone no. and email. These all are encrypted and stored in the database.

In the password section, the users are required to select a category of images from the given set of categories. To choose his or her pattern type, the user can choose any number of categories. After selecting the categories, the user will be displayed a set of images in a grid view of NxN size where N is the grid size. The images will be fetched through a cloud source that will be used in this project. These images in the grid will be randomly shuffled. The users are required to select multiple images as their passwords. During the password selection process, an image can only be used once. Duplication of images in the password selection is not allowed because it decreases the randomness of the proposed system. For that, the system will not display the same images multiple times to increase randomness

During the pattern selection, the user when clicking on the image, the image will be faded out to reduce shoulder surfing. **Shoulder surfing** is a criminal crime in which thieves steal personal information from you while you are using a laptop, ATM, public kiosk, or other electronic devices in public. Now, after the pattern is selected by the user, the pattern with the images will be sent to the cloud. The cloud will be containing two main algorithms that will be applied to the images before sending them to the database. Those two algorithms will be applied on the two separate copies of the pattern respectively. The copies of the pattern will be destroyed after sending them to the database. That means, even the person, having access to the database, couldn't be able to find the pattern.

Even if a data breach occurs, the attacker will be unable to crack the pattern. We will be using **Hashing** and **DeepImageSearch** engine in the cloud to secure them before sending them to the database.

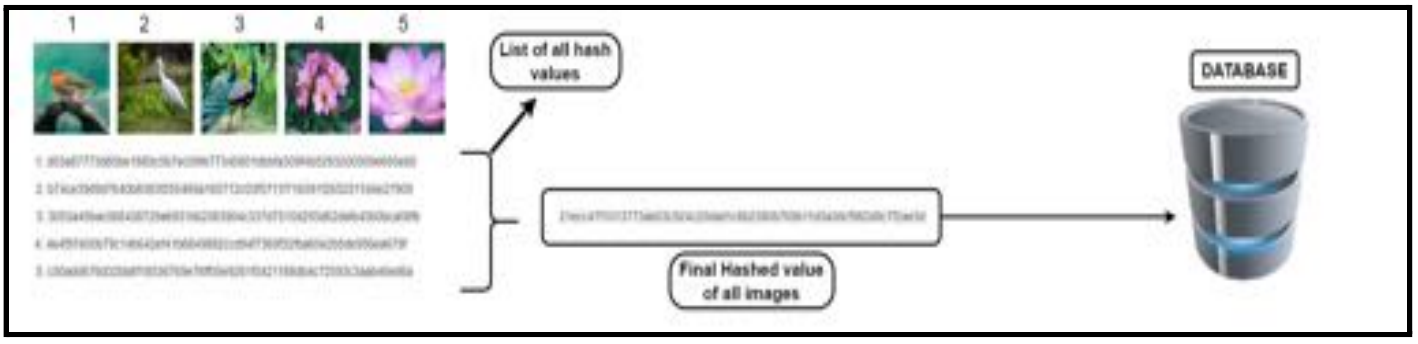
1. The first one will be the **Hashing** technique. We will be using this technique on the first copy of the pattern. We will be using the **Two-Level Hashing** technique to secure the images in the database. In the first level of hashing, we will be using **Image-hashing**. **Image hashing** is the concept of hashing images such that even if data is leaked, the hacker will have a hard time recovering the hash key obtained by the hashing process. We will be using **SHA-256** which is the best level of hashing format to hash files. The **SHA-256** is a patented cryptographic hash function that outputs a value that is 256 bits long. The key will be displayed something like this: **f9c8ff00149f08bf**. This is 256 bits generated value that the hashing function generates. This value can't be reverted back to its original file.

In the second level of hashing, we will simply merge all the images i.e., we will be concatenating all the selected images that are in a pattern, and then we will apply another hashing algorithm called **Bcrypt Algorithm** that uses salts to generate a hash value. **Bcrypt** uses a 128-bit salt and encrypts a 192-bit magic value. It takes advantage of the expensive key setup in ***eksblowfish***. A salt is used as an additional input to a hash function that hashes a password. The slower the process is, the better hash it can generate by adding salts to the hashed value. An example of using the Bcrypt Algorithm on concatenated hash values of images is shown below:

**image_1** = f9c8ff00149f08bf
**image_2** = ffc9ff00149f08ff
**image_3** = f9c8ff00149f08bf
**image_4** = f9c8ff00149f08bf

**Concatenated_value** = f9c8ff00149f08bfffc9ff00149f08fff9c8ff00149f08bff9c8ff00149f08bfe781a5a5d983c3e7

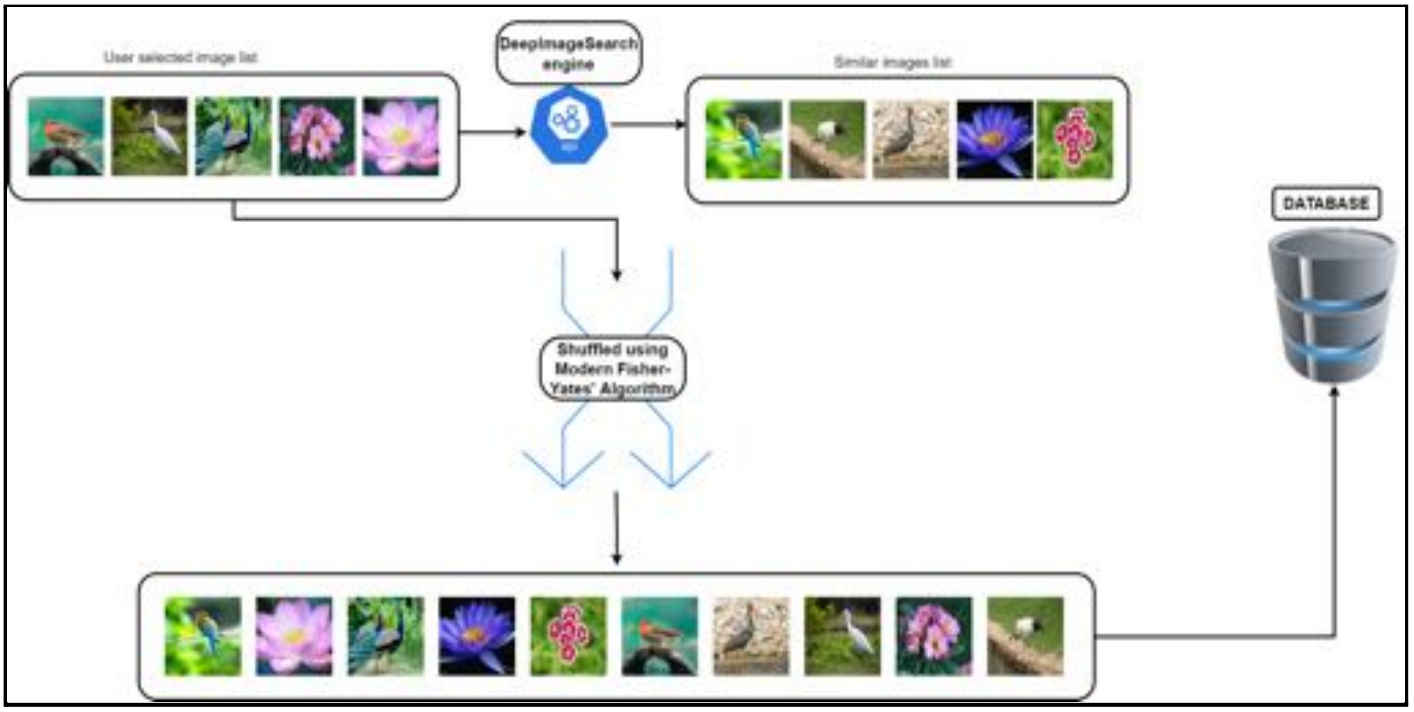
**After applying bcrypt** = $2a$10$WFrh8XI87Lcp.65jFLwbM.a75iVBxEqahBn1msJ0KnFiSTgZwUV..
Now the hashed value of the first copy will be sent to the database.




2 . The second will be for finding similar images of the images that the user has selected in a pattern. We will be using an open-source engine that is known as **DeepImageSearch** which is an AI-based image search engine that includes *Deep transfer learning* features *Extraction* and *tree-based vectorized search technique*. This project is developed by *Nilesh Verma*. Using this engine, we can find similar images based on the images we have selected. So, after hashing, **DeepImageSearch** will utilize the second copy of the pattern to find similar images of the given pattern.

After finding similar images, the image list will be sent to the database to the same location where the hashed value is stored. Note: Only the image names will be stored in the database. Because all the images will be retrieved through the cloud/server. So we don't need to save the images in the database as it will require a huge bandwidth & will slow down the process.

So now, we will be having two fields stored for the same pattern of images. 1. is the hashed value of the pattern and 2. is the list of similar images names shuffled with the images in the selected pattern.




Now coming to the second module, it will be the **Login phase**. In this phase, the user needs to select the same pattern of the images that he/she has selected during the registration phase. After entering the email id, the user will be shown a set of images in a grid box. This set of images will contain the images that he/she has selected during registration and also the similar images that were generated using the **DeepImageSearch** engine. Let's say, the user has selected a total of 8 images in a pattern. So after finding similar images for each image & shuffling all the images, the total images will be 16. Now the grid box contains a total of 25 boxes. The 16 images can be placed on the 16 boxes. But the remaining boxes shouldn't be left empty. So, we will be placing a set of random images from all categories using an algorithm known as **Modern Fisher-Yates** algorithm.

The modern version of the **Fisher–Yates** shuffle, designed for computer use, was introduced by Richard Durstenfeld in 1964 and popularized by Donald E. Knuth in *The Art of Computer Programming* as "Algorithm P (Shuffling)". This algorithm will be using a time complexity of O(n) compared to O(n^2) for the naïve implementation.This change gives the following algorithm (for a zero-based array).

*-- To shuffle an array a of n elements (indices 0..n-1):*

for $i$ from $n-1$ `downto 1 do`

$j \leftarrow$ random integer such that $0 \le j \le i$

exchange $a[j]$ and $a[i]$

After placing the random images on the grid box, the user will be displayed the set of images in the grid box. After the successful selection of the pattern, the user will be authenticated successfully to the dashboard.

**OVERVIEW:**

- This web application demonstrates the **graphical password method**.
- This application helps us to create a secure graphical password rather than alpha numeric password in order to authenticate a user.
- This application displays a set of images in a grid view from different set of categories.
- The user can select particular set of categories to display only those set of images in order to select the pattern.

**The solution for the above problem statement is given based upon the real-time scenarios. They are:**

**A. Scenario-1:**

**Shoulder Surfing**

*Solution:*

When we click on a particular image it will fade out . By using this feature it will allow us hide the image from other. When someone is trying to watch our patterns from shoulder while we click on a image it will completely fades out which will helps us to hide the image from them. We can also use number on image disappearance , it will displays a number on the image which we are selecting. It will also completely fades the image which others cannot recognize it. By using both these type we can fade the each and every image we are selecting which will avoid others from knowing the pattern we are using.

**B. Scenario-2:**

**In case of Data Breach**

*Solution:*

The most crucial aspect of any website or web application is security. In terms of security, we are using two-level hashing techniques so that even if there is a data breach, the attacker will have a very hard time cracking the pattern & will make it completely impossible in cracking the pattern.

**WED APPLICATION FEATURES:**

1. Security.

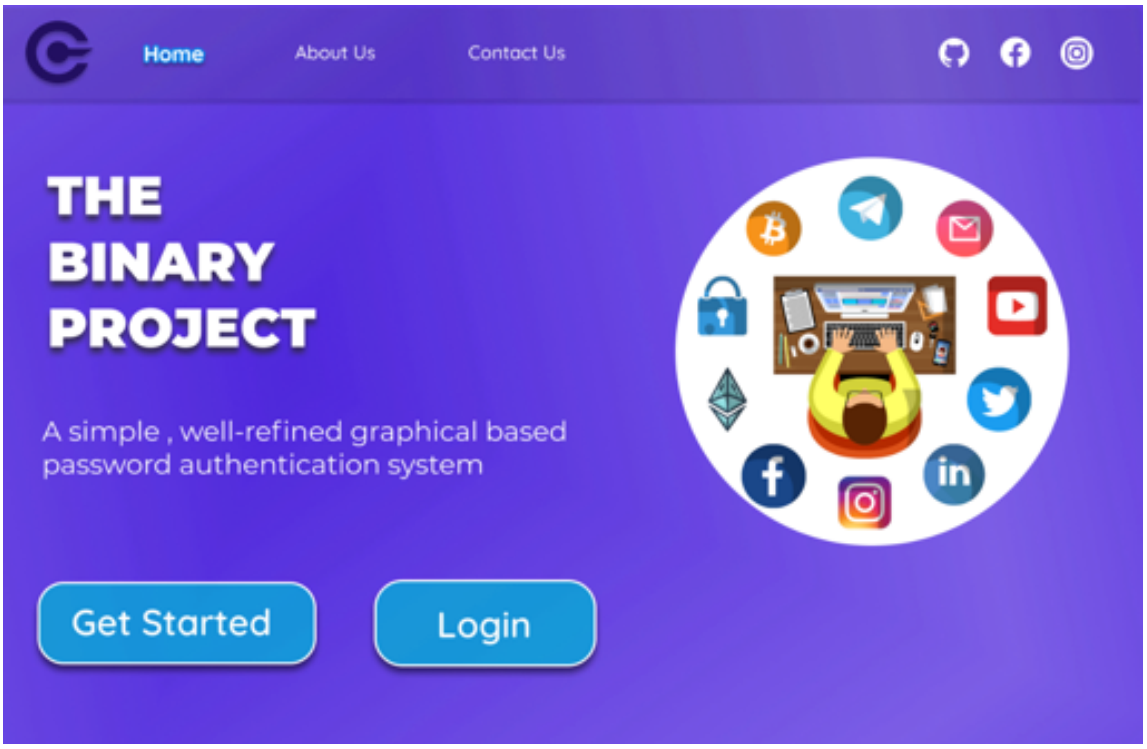
2. Fading Out images to Reduce Shoulder Surfing.

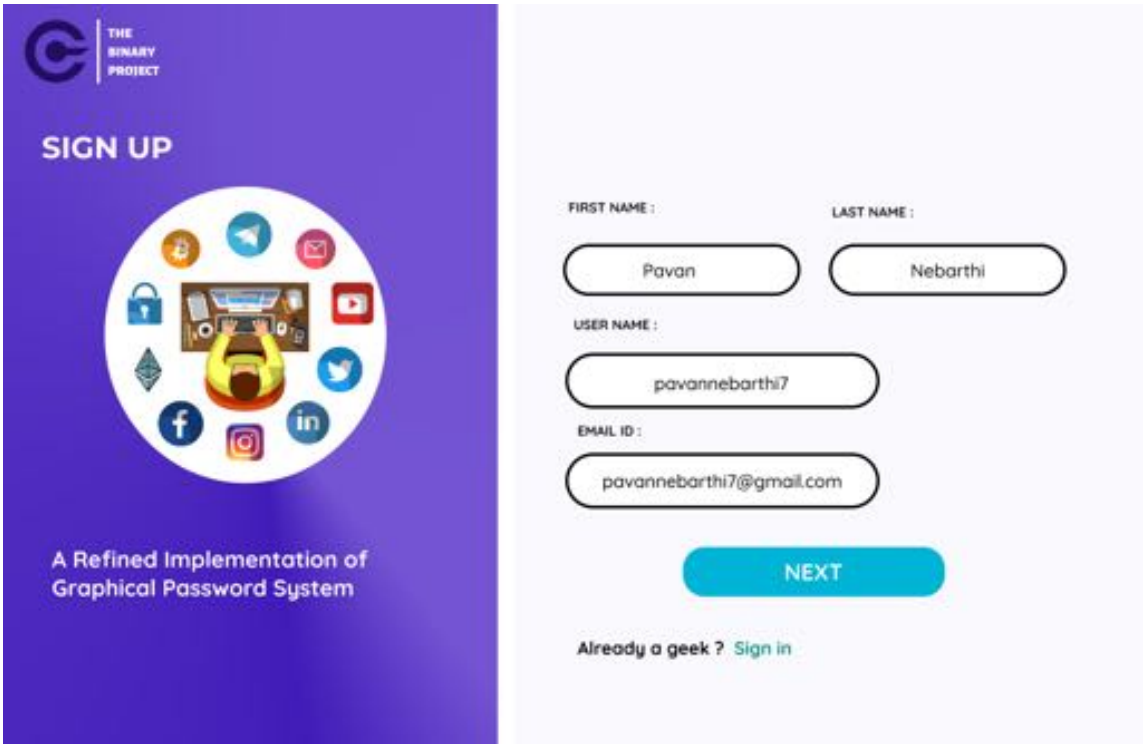
3. Using similar images.

The Detailed Explanation about the features:

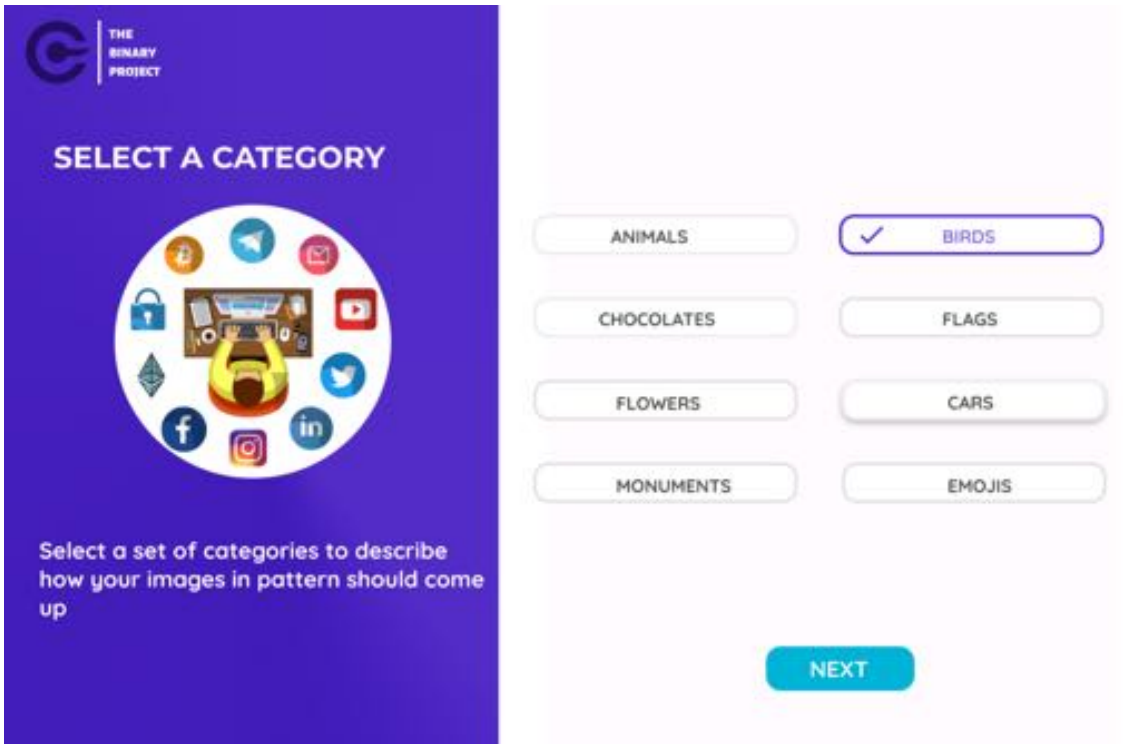
- **Security:** Our program has a high level of security, which will assist us protect our data from being hacked. We use a hashing algorithm to transform images into hash values, which is a one-way operation; the hashed data cannot be converted back into images. Also, to prevent shoulder surfing, the images will be faded out, making it impossible for others to see whatever images the user choice.
- **Fading out images:** When the user will select a particular images those images will fade out to avoid others to know the images which the user has selected. This will avoid the outsiders to know the pattern which the user has selected.
- **Using similar images:** They will be given similar images according to the pattern selected by the user. When the user logins into the page the patterns will be shuffled to trick the shoulder surfing, This will be difficult for them to identify the pattern of images.

**PROTOTYPES:**

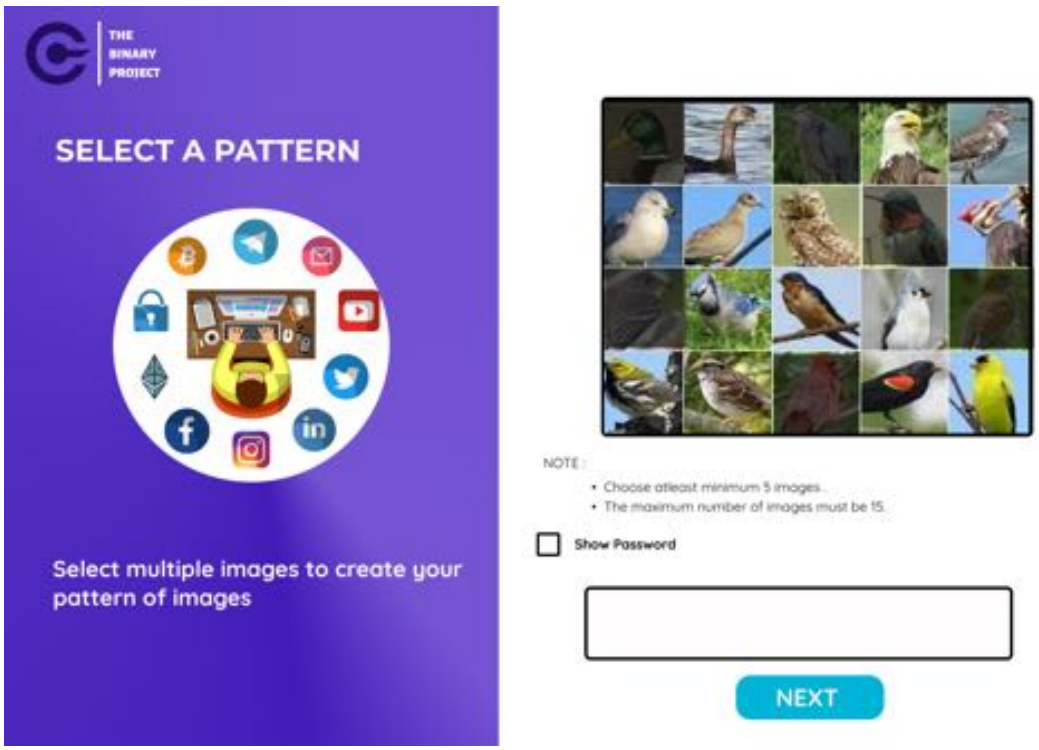


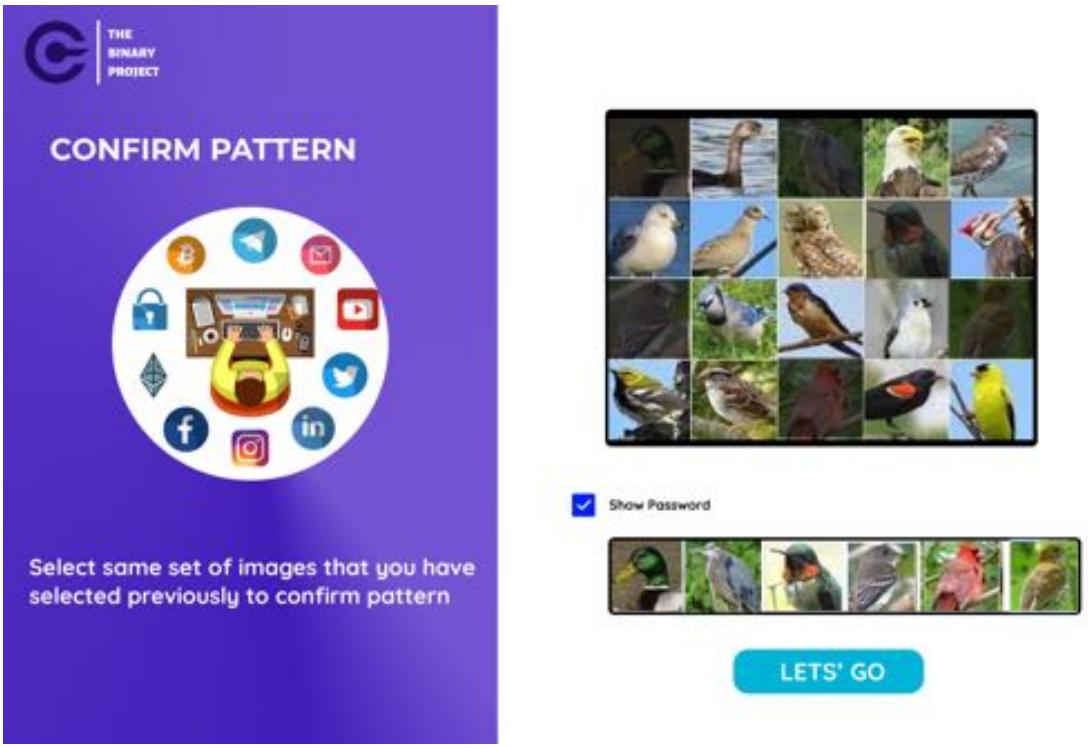

2. Sign Up Screen



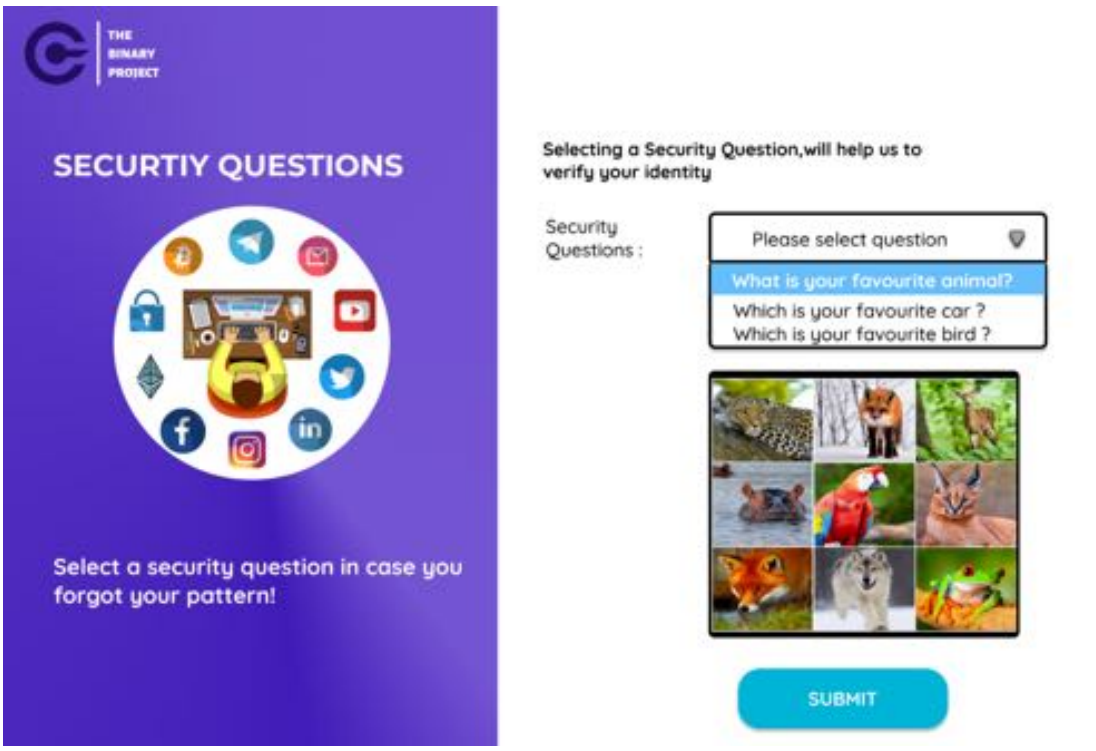

3. Categories Screen



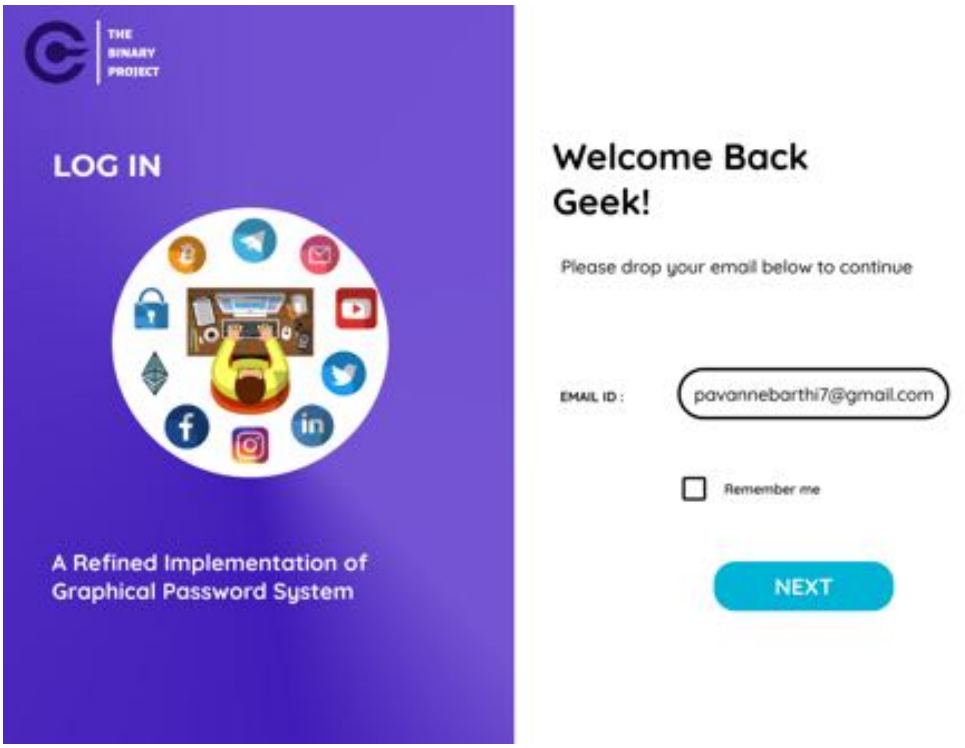

4. Pattern Screen



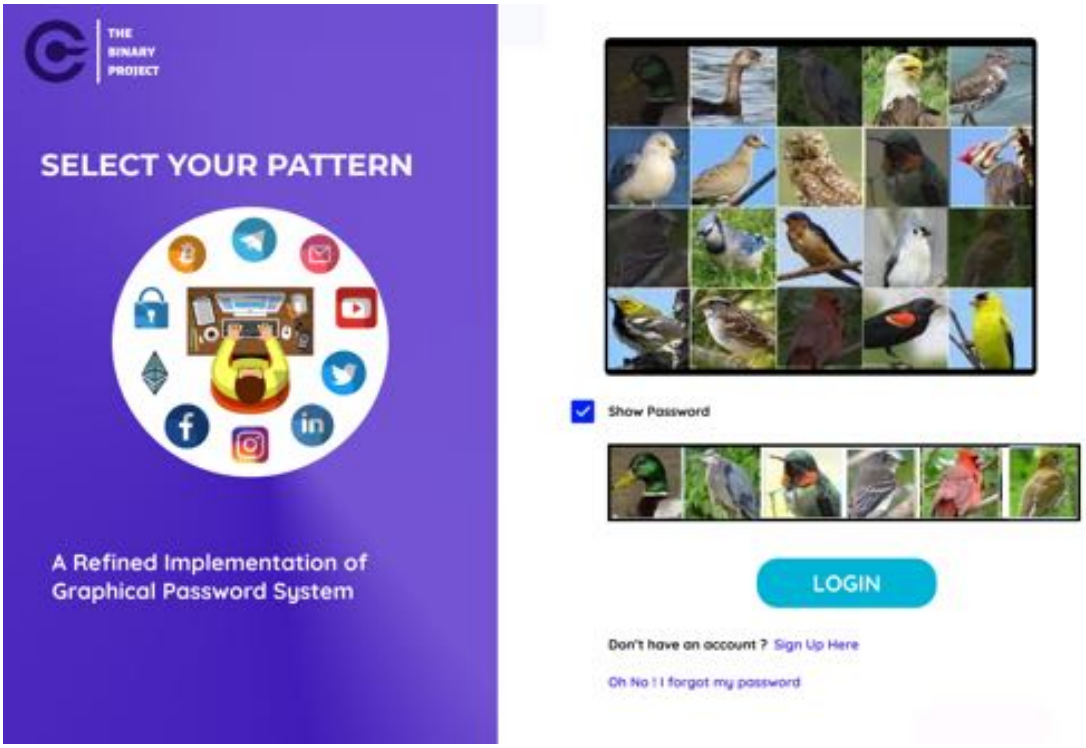

5. Confirm Pattern Screen



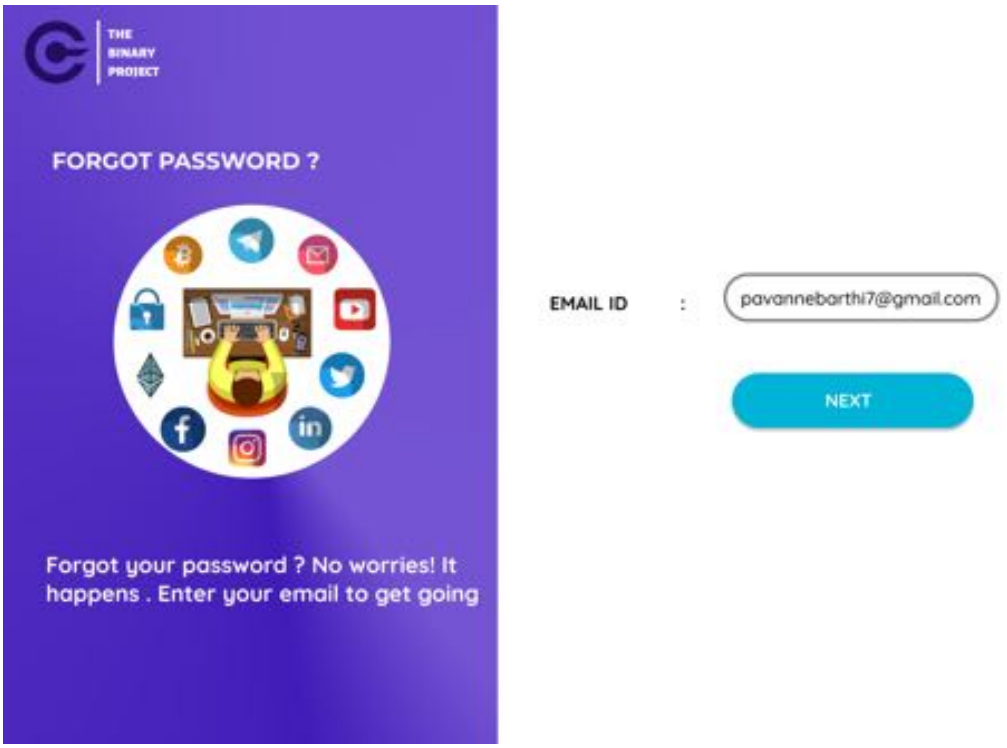

6. Security Questions Screen



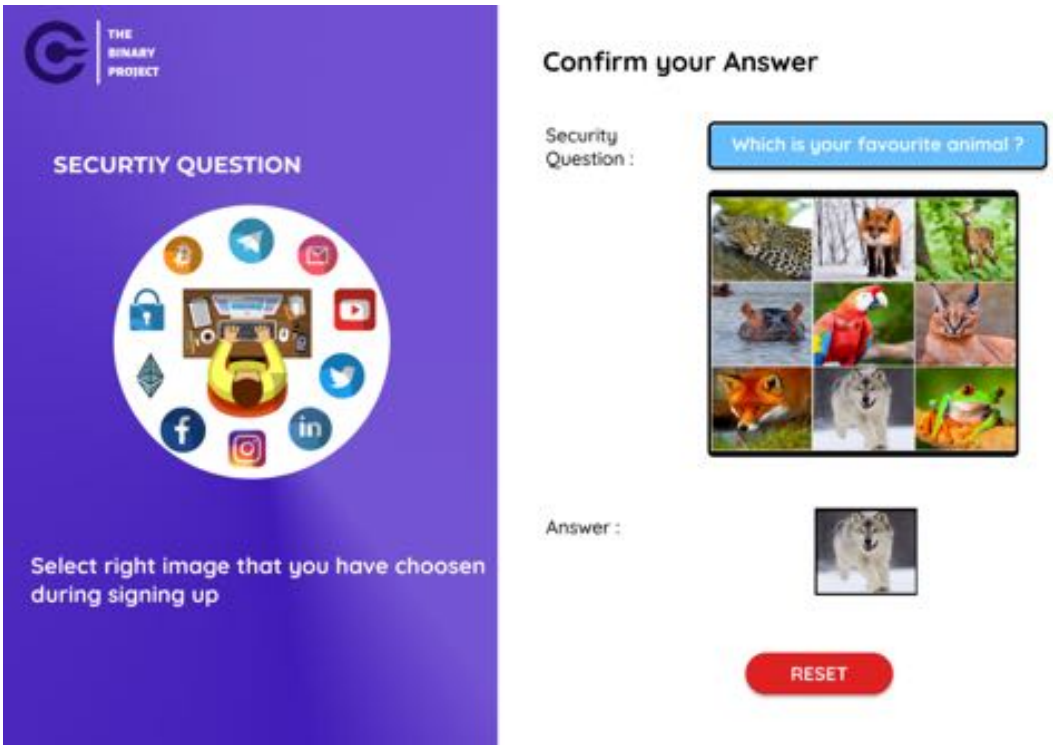

7. Login Screen




8. Login Pattern Selection

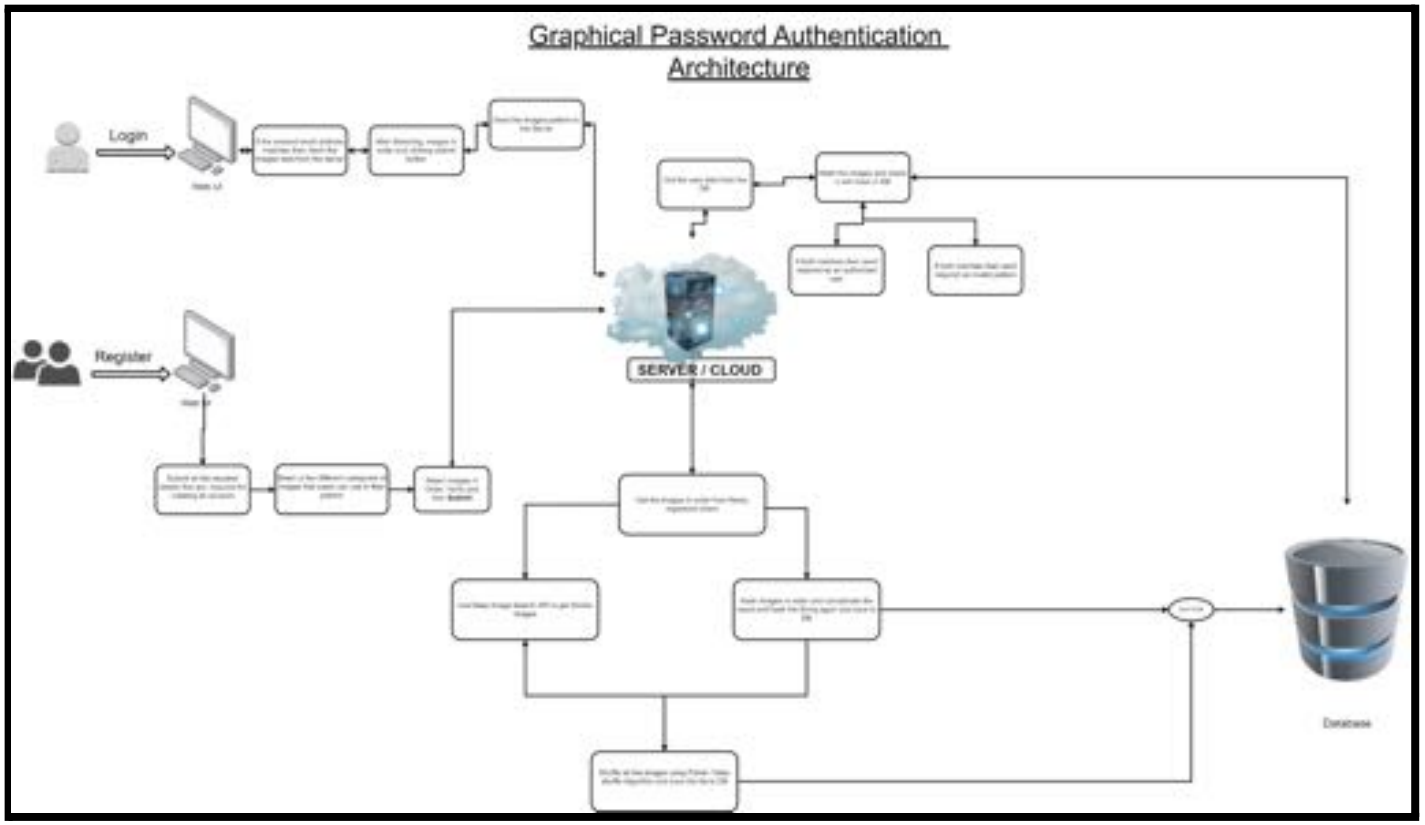



9. Forgot Password Screen




10. Forgot Password Security Question

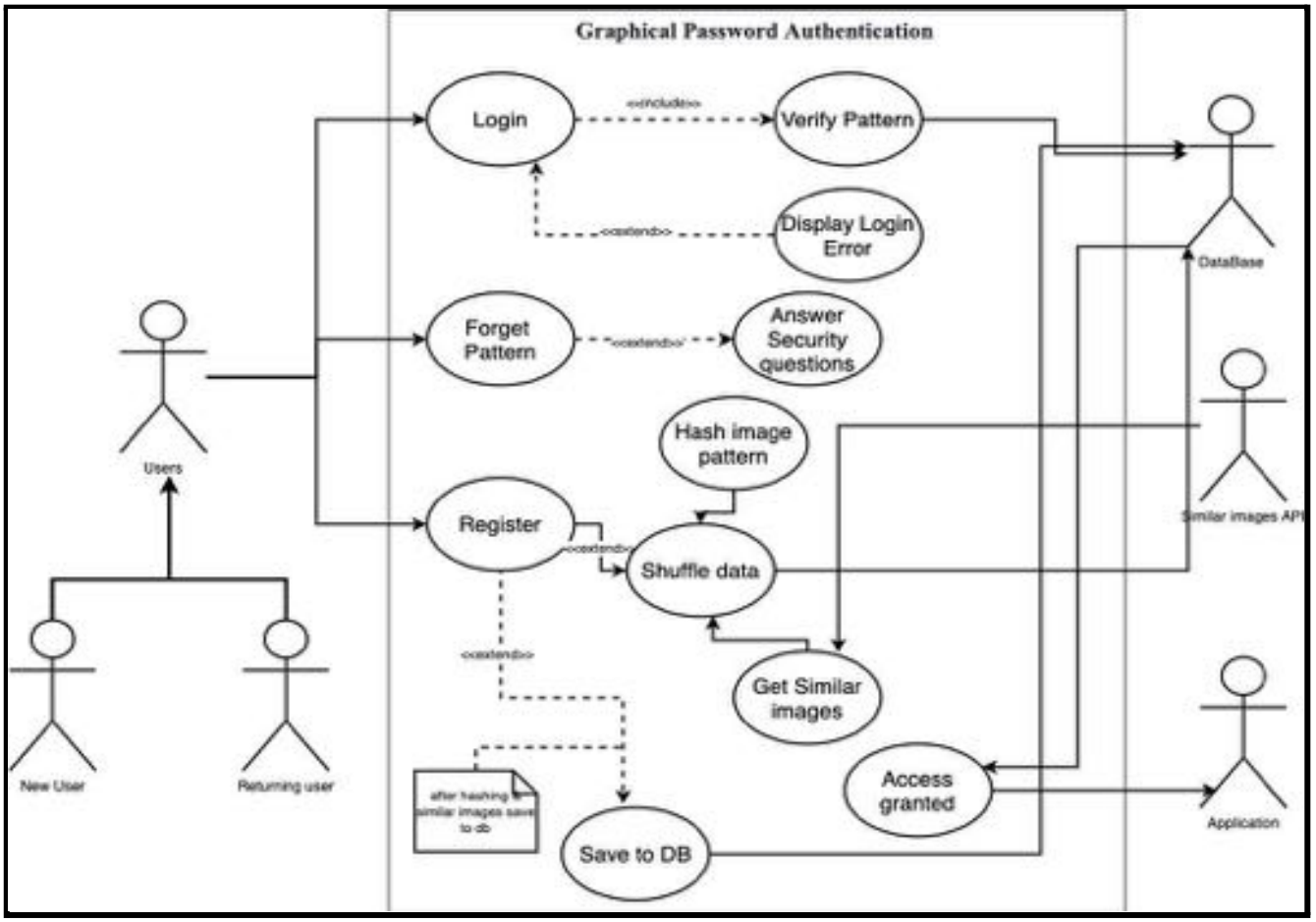
**ARCHITECTURE:**




**TECHNOLOGIES:**

| | |
|---|---|
| **Frontend** | ***ReactJS, Bootstrap*** |
| **Backend** | ***NodeJS, ExpressJS*** |
| **Database** | ***MongoDB*** |
| **Cloud/Server** | ***Digital Ocean*** |

**USE CASE:**




**Team Leader Name: N Rishi Raj Reddy**

Branch :**Btech** Stream :CSE Year : IV/IV

**Team Member 1 Name: J.Sai Kumar**

Branch: **Btech** Stream: CSE Year : IV/IV

**Team Member 2 Name: N.Pavan**

Branch : **Btech** Stream : CSE Year : IV/IV

**Team Member 3 Name: Ch.Chandra Mouli**

Branch: **Btech** Stream : CSE Year : IV/IV

**Team Member 4 Name: D.V.S.N.Uma Swetha**

Branch : **Btech** Stream : CSE Year : IV/IV

**Team Member 5 Name: D.S.L.Satya Priya**

Branch: **Btech** Stream : CSE Year : IV/IV

**Team Mentor 1 Name: Kalisetty Venkata Lakshmi** **Industry, ML** Experience :11 years

**Team Mentor 2 Name: Chokkakula Karthik** **Industry, ML** Experience: 11 years