2021

PASSWORD MANAGEMENT SYSTEM

COMPUTER PROJECT

GAURI RAMABHADRAN

CLASS-XII-C
ROLL NO.-18611611

INDEX

ACKNOWLEDGEMENT

I undertook this Project work, as the part of my XII-Computer Science course. I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing password management systems is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help and good wishes of the people near you, who have considerable experience and idea.

I would like to extend my sincere thanks and gratitude to my principal **Manju Sharma Ma'am.** I am very much thankful to my teacher **Vasudha Ma'am** for giving valuable time and moral support to develop this software.

I would like to take the opportunity to extend my sincere thanks and gratitude to my parents for being a source of inspiration and providing time and freedom to develop software projects.

I feel indebted to my friends for the valuable suggestions during the project work.

OBJECTIVE

This project aims to developing software that can be used for securing their email IDs and passwords. This project is useful because we generally forget our email ID or passwords and if we note down on a diary or any paper anyone can access it and misuse our email id. So here is a solution - you can store your data in password management system. It is simple and convenient for security.

According to the above facts the objectives of password management systems are:

- Identifying the importance of passwords as it concerns the advantages and disadvantages in their daily use in home and corporate environments.
- Identifying the weaknesses raised from these poorly chosen passwords and describe the modern attacking techniques against these passwords. Besides proposing possible countermeasures to address and eliminate these attacks.
- Conducting a critical analysis of different techniques used to facilitate users to remember strong passwords easily.
- Analyzing the operating principles of the Password Management System and the processes that it enforces in order to produce "safe passwords".
- Test this password generator system for the strength of all passwords it generates.

PROBLEM DEFINITION

Remembering passwords is a pain. Around 85% of the people forget the passwords they have used for each and every website. Creating different passwords each time is a huge problem to remember. Since most of the websites accept strong complex passwords it becomes really difficult to remember them all. Keeping in mind all of these issues, we have created a password management system that can save the passwords of various websites along with the email and phone number.

You now don't have to worry, and be stressed about all the password remembering. Our site not only helps you to store important information but also detects which password is weak and which is strong. One can add new records, delete unnecessary records, update and sort by the records according to his/her need.

You can register as well as log into the website according to your convenience.

ANALYSIS

INPUT-OUTPUT DESCRIPTION

1. If you do not have an account click on sign up and enter a unique username, click on submit

2. Now choose a strong password and you have created you password management account

3. If you already have an account click on 'log in', type in your password and begin

4. You can perform different functions on password management system like:
    - **Adding records**
      Click on 'Add new' from Menu and fill in – website name, username, password, email Id and phone number. Press 'Submit' and a new record is added to your password manager table
    - **Deleting record**
      To delete a record from your password manager table click on 'Delete' from Menu.
      Type the record ID, you would like to delete and press on 'Submit'.
    - **Updating records**

To update records click on ‘Update’ from Menu. Choose the column you would like to update and type ID of the record to change and the new information below it
Press ‘Submit’ and your record has been updated

- **Sorting your records**
  To sort your records for convenience, click on ‘Sort by’ and choose a column from the list that appears below. Your table is sorted
- **To check the strength of your password**
  Click on ‘Go to password check up’ from Menu. You will notice a new column in your table – check, indicating whether your password is weak or strong

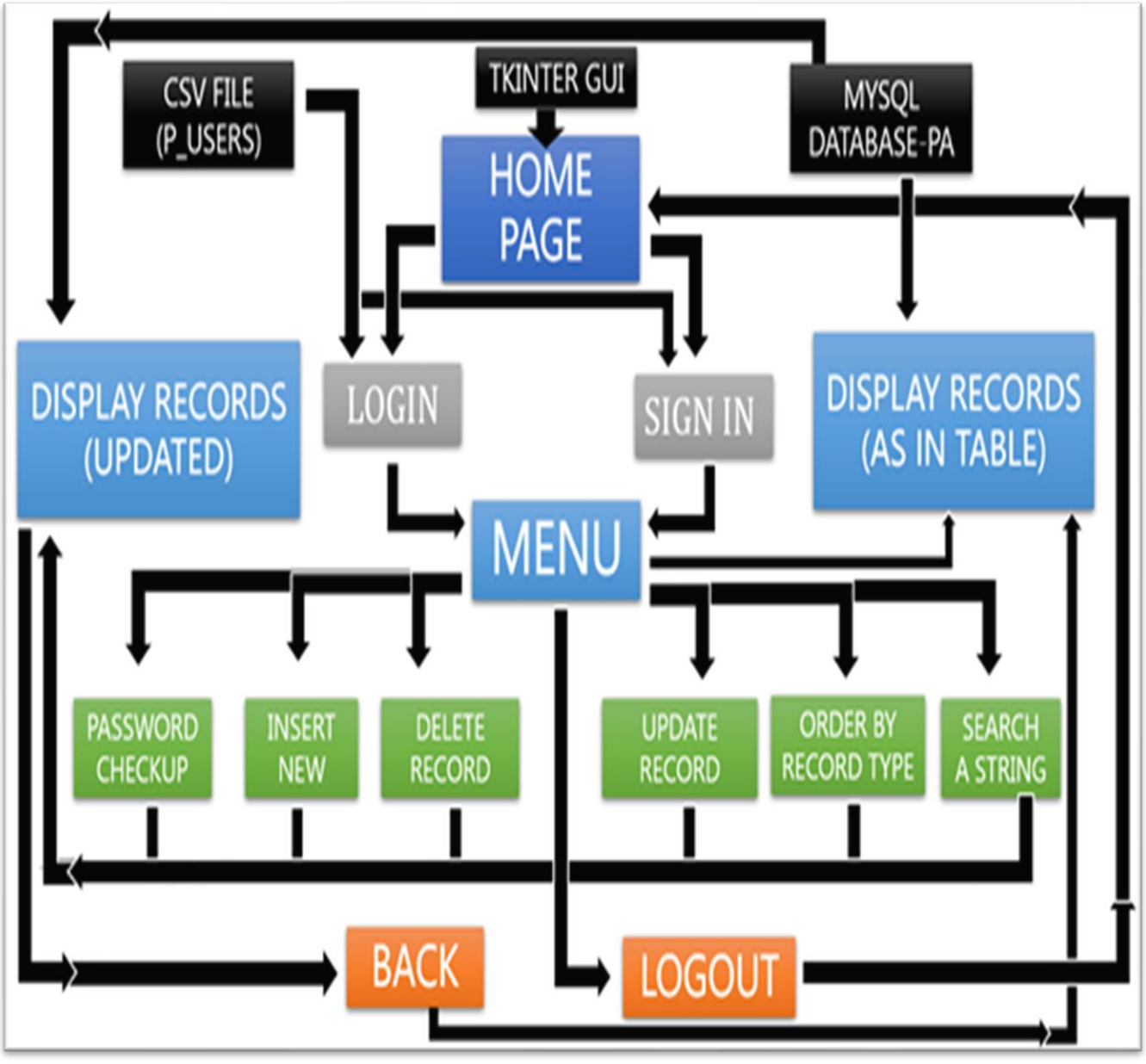
5. Refreshing your password manager can be done by clicking on ‘MENU’

6. Click on ‘back’ button to clear your changes

7. To search for anything present in your table, click on the search bar and type in the information you are searching for

8. After you are done working on password manager you can log out by clicking on ‘Logout’ which leads you to the main page

BLOCK DIAGRAM



LANGUAGES USED

MYSQL

A database is a collection of information related to a particular subject or purpose. Using any RDBMS application software like MySQL, you can store your information in a single database file. Within the file, divide your data into separate storage containers called tables. You may and retrieve the data using queries.

Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organizes data into columns (called fields) and rows (called records).

A Primary key is one or more fields whose value or values uniquely identify each record in a table. In a relationship, a primary key is used to refer to specific record in one table from another table.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for Password Management system.

PYTHON

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently. There are two major Python versions - Python 2 and Python 3. Both are quite different. Python 3 is used for this project.

Reason for increasing popularity:

1. Emphasis on code readability, shorter codes, ease of writing
2. Programmers can express logical concepts in fewer lines of code in comparison
   to languages such as C++ or Java.
3. There exists inbuilt functions for almost all of the frequently used concepts.
4. Philosophy is ‘‘Simplicity is the best’’.

CSV

The CSV (Comma Separated Values) format refers to tabular data that has been saved as plaintext where data is separated by commas.

In CSV Format:

Each row of the table is stored in one row i.e. the number of rows in a CSV file are equal to the number of rows in the table.

The field-values of row are stored together with commas after every field value; but after the last field's value in a line/row, no comma is given, just the end of line.

Advantages:

- A simple, compact and ubiquitous format for data storage.
- A common format for data interchange.
- It can be opened in popular spreadsheet packages like MS-Excel, Calc etc.

LIBRARIES USED

TKINTER

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. GUI is nothing but a desktop app that provides you with an interface that helps you to interact with the computers and enriches your experience of giving a command (command-line input) to your code. They are used to perform different tasks in desktops, laptops, and other electronic devices, etc.

Methods used

a. geometry( ) : This method is used to set the dimensions of the Tkinter

window as well as it is used to set the position of the main window on the

user's desktop.

b. Frame( ) : It works like a container, which is responsible for arranging the

position of other widgets. It uses rectangular areas in the screen to organize

the layout and to provide padding of these widgets.

c. grid( ) : This geometry manager organizes widgets in a table-like structure in

the parent widget.

d. pack ( ) : This geometry manager organizes widgets in blocks before placing

them in the parent widget.

e. Label ( ) : This widget implements a display box where you can place text or

images. The text displayed by this widget can be updated at any time you

want.

f. Button ( ) : The Button widget is used to add buttons in a Python application.

These buttons can display text or images that convey the purpose of the

buttons. You can attach a function or a method to a button which is called

automatically when you click the button.

g. Entry ( ) : The Entry widget is used to accept single-line text strings from a user

PYMYSQL

PyMySQL is an interface for connecting to a MySQL database server from Python. It implements the Python Database API v2.0 and contains a pure-Python MySQL client library.

Methods used:

a. connect( ) : Establish a connection to the MySQL database.

b. cursor ( ) : Creates a cursor object. This is the object you use to interact with

the database.

c. execute ( ) : This method executes the given database operation (query or

command). The parameters found in the tuple are bound to the variables in

the operation.

d. fetchall ( ) : The fetchall() method retrieves all (remaining) rows of a query

result, returning them as a sequence of sequences.

e. commit ( ): This method commits the current transaction.

f. rollback ( ) : This method reverts the changes made by the current transaction.

CODE

```
#importing all the necessary files
from tkinter import *
import mysql.connector as m
import csv

#connecting to sql server
con=m.connect(host='localhost',user='root',passwd='0000',database='
pa')

#program

def search_rec(a,z):
    #to search records

    #creating a new frame
    f=Frame(w,height=800,width=1050,bg='gray11').place(x=500,y=0)

    #header line
    Label(f,text="ALL  RECORDS  ARE",bg='gray11',fg='snow',font=('mv
boli',40),bd=4).place(y=25,x=700)

    def sub(a):
        #extracting vallue of searched record
        z3=z2.get()
        search_rec(a,z3)

    #declaring search value
    z2=StringVar()

    #to enter text for user to search
```

```
Entry(f,textvariable=z2,fg='gray11',bg='snow',font=('coureir','18'),).pl
ace(y=125,x=1050)
   #bytton to search

Button(f,image=pic,command=lambda:sub(a),bg="gray11",relief='fla
t').place(x=1300,y=120)
   #button to go back
   Button(f,text="BACK",
width=6,bg='gray11',fg='snow',font=('coureir','18'),command=lambda
:show_all(a),relief='ridge').place(y=125,x=600)

   #extracting all values from sql
   c=con.cursor()
   s="select * from {}".format(a)
   c.execute(s)
   d=c.fetchall()

   m=200
   n=550

   z=z.lower()

   #displaying header

Label(f,text='ID',width=3,fg='gray11',bg='honeydew3',font=('@micros
oft yahei ui','12'),pady=2,).place(y=m,x=n-45)

Label(f,text='WEBSITE',width=14,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n-5)

Label(f,text='USERNAME',width=18,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+145)

Label(f,text='PASSWORD',width=13,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+340)
```

```
Label(f,text='EMAILID',width=27,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+490)
   Label(f,text='PHONE
NO',width=11,fg='gray11',bg='honeydew3',font=('@microsoft     yahei
ui','12'),pady=2,).place(y=m,x=n+775)
   m+=50

   #to display records
   for i in d:

      #to display only for which it is seached
      if z in i[1].lower() or z in i[2].lower() or z in i[3].lower() or z in
i[4].lower() or z in i[5].lower():

         Label(f,text=str(i[0]),width=3,
fg='gray11',bg='mintcream',font=('coureir','12'),pady=2,).place(y=m,x
=n-45)

Label(f,text=str(i[1]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n-5)

Label(f,text=str(i[2]),width=20,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+145)

Label(f,text=str(i[3]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+340)

Label(f,text=str(i[4]),width=30,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+490)

Label(f,text=str(i[5]),width=12,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+775)
         m+=30


def show_all(a):
```

```
    #to display all records

     #creating a new frame
    f=Frame(w,height=800,width=1050,bg='gray11').place(x=500,y=0)
    #heading line
    Label(f,text="ALL  RECORDS  ARE",bg='gray11',fg='snow',font=('mv
boli',40),bd=4).place(y=25,x=700)

    def sub(a):
        #extracting search value and calling the function
        z2=z.get()
        search_rec(a,z2)

    #declaring variable for search
    z=StringVar()
    #giving user space to type what to search

Entry(f,textvariable=z,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=125,x=1050)
    #button to search

Button(f,image=pic,command=lambda:sub(a),bg="gray11",relief='fla
t').place(x=1300,y=120)

    #ectracting values from sql
    c=con.cursor()
    s="select * from {}".format(a)
    c.execute(s)
    d=c.fetchall()

    m=200
    n=550

    #displaying heading

Label(f,text='ID',width=3,fg='gray11',bg='honeydew3',font=('@micros
oft yahei ui','12'),pady=2,).place(y=m,x=n-45)
```

```
Label(f,text='WEBSITE',width=14,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n-5)

Label(f,text='USERNAME',width=18,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+145)

Label(f,text='PASSWORD',width=13,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+340)

Label(f,text='EMAILID',width=27,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+490)
   Label(f,text='PHONE
NO',width=11,fg='gray11',bg='honeydew3',font=('@microsoft     yahei
ui','12'),pady=2,).place(y=m,x=n+775)
   m+=50

   #displaying all records
   for i in d:

      Label(f,text=str(i[0]),width=3,
fg='gray11',bg='mintcream',font=('coureir','12'),pady=2,).place(y=m,x
=n-45)

Label(f,text=str(i[1]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n-5)

Label(f,text=str(i[2]),width=20,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+145)

Label(f,text=str(i[3]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+340)

Label(f,text=str(i[4]),width=30,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+490)
```

```
Label(f,text=str(i[5]),width=12,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+775)
        m+=30


def p_check(a):
    #to check if passowrd is strong or not

    def strongpass(s):
        #checks password

        a,b,c,d=0,0,0,0

        #to see if length greater than 8
        if len(s)>=8:

            #checks if all components are there
            for i in s:

                #lowercase
                if i.islower():
                    a+=1

                #uppercase
                elif i.isupper():
                    b+=1

                #numbers
                elif i.isdigit():
                    c+=1

                #symbols
                else:
                    d+=1

            if a>0 and b>0 and c>0 and d>0:
```

```
            return True
            #strong passowrd
        else:
            return False
            #weak password
    else:
        return False
        #weak password

  #creating a new frame
  f=Frame(w,height=800,width=1050,bg='gray11').place(x=500,y=0)
  #heading label
  Label(f,text="ALL  RECORDS  ARE",bg='gray11',fg='snow',font=('mv
boli',40),bd=4).place(y=25,x=700)

  def sub(a):
    #to extract search value and call function
    z2=z.get()
    search_rec(a,z2)

  #declaring variable for search
  z=StringVar()
  #giving space to search

Entry(f,textvariable=z,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=125,x=1050)
  #calling search function using button

Button(f,image=pic,command=lambda:sub(a),bg="gray11",relief='fla
t').place(x=1300,y=120)
  #to go back to main table
  Button(f,text="BACK",
width=6,bg='gray11',fg='snow',font=('coureir','18'),command=lambda
:show_all(a),relief='ridge').place(y=125,x=600)

  #extracting all values from sql
  c=con.cursor()
```

```
    s="select * from {}".format(a)
    c.execute(s)
    d=c.fetchall()

    m=200
    n=550

    #displaying heading row

Label(f,text='ID',width=3,fg='gray11',bg='honeydew3',font=('@micros
oft yahei ui','12'),pady=2,).place(y=m,x=n-45)

Label(f,text='WEBSITE',width=14,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n-5)

Label(f,text='USERNAME',width=18,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+145)

Label(f,text='PASSWORD',width=13,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+340)

Label(f,text='EMAILID',width=27,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+490)
    Label(f,text='PHONE
NO',width=11,fg='gray11',bg='honeydew3',font=('@microsoft     yahei
ui','12'),pady=2,).place(y=m,x=n+775)

Label(f,text='CHECK',width=6,fg='gray11',bg='honeydew3',font=('@
microsoft yahei ui','12'),pady=2,).place(y=m,x=n+905)
    m+=50

    #variables to analyse number of strong or weak password
    s1=0
    s2=0

    #displaying all records
    for i in d:
```

```
        Label(f,text=str(i[0]),width=3,
fg='gray11',bg='mintcream',font=('coureir','12'),pady=2,).place(y=m,x
=n-45)

Label(f,text=str(i[1]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n-5)

Label(f,text=str(i[2]),width=20,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+145)

Label(f,text=str(i[3]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+340)

Label(f,text=str(i[4]),width=30,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+490)

Label(f,text=str(i[5]),width=12,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+775)

        #after checking if password is strong or not, it displays as it is
        if strongpass(i[3]):


Label(f,text="STRONG",width=7,fg='gray11',bg='darkolivegreen3',fo
nt=('coureir','12'),pady=2,).place(y=m,x=n+905)
            m+=30
            s1+=1

        else:

Label(f,text='WEAK',width=7,fg='gray11',bg='indianred1',font=('cour
eir','12'),pady=2,).place(y=m,x=n+905)
            m+=30
            s2+=1
```

```
   #creating new frame in menu to show compelete analysis of
password
   f1=Frame(w,height=375,width=500,bg='gray11').place(x=0,y=350)

   #displaying analysis
   Label(f1,text="PASSWORD ANALYSIS -
".format(s1),bg='gray11',fg='snow',relief='flat',font=('@Microsoft
YaHei UI',18,)).place(x=20, y=400)
   Label(f1,text="{} websites have strong
password.".format(s1),bg='gray11',fg='snow',relief='flat',font=('@Micr
osoft YaHei UI',16)).place(x=20, y=500)
   Label(f1,text="{} websites have weak
password.".format(s2),bg='gray11',fg='snow',relief='flat',font=('@Micr
osoft YaHei UI',16)).place(x=20, y=550)
   Label(f1,text="Advised to change these
passwords.",bg='gray11',fg='snow',relief='flat',font=('@Microsoft
YaHei UI',16)).place(x=20, y=600)



def insert_new(a):
   #to ask user if he wants to insert new values

   #creating a new frame
   f=Frame(w,height=375,width=500,bg='gray11').place(x=0,y=350)

   #header
   Label(f,text="ENTER VALUES
HERE",bg='gray11',fg='snow',relief='flat',font=('@Microsoft YaHei
UI',16)).place(y=350,x=50)

   def sub(a):
      #extracting values
      n2=n.get()
      o2=o.get()
      p2=p.get()
      q2=q.get()
      r2=r.get()
```

```
        #to indert values in table using these values
        query(a,n2,o2,p2,q2,r2)

    d=400
    e=50

    #defining which entry is for what
    Label(f,text="Website                                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=d,x=e)
    Label(f,text="Username                                                   :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=d+50,x=e)
    Label(f,text="Password                                                   :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=d+100,x=e)
    Label(f,text="Emailid                                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=d+150,x=e)
    Label(f,text="Phone_no                                                   :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=d+200,x=e)

    #assigning variables
    n=StringVar()
    o=StringVar()
    p=StringVar()
    q=StringVar()
    r=StringVar()

    #user to type all the variables

Entry(f,textvariable=n,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=d,x=e+130)

Entry(f,textvariable=o,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=d+50,x=e+130)

Entry(f,textvariable=p,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=d+100,x=e+130)
```

```
Entry(f,textvariable=q,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=d+150,x=e+130)

Entry(f,textvariable=r,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=d+200,x=e+130)

   #submit button

Button(f,text="SUBMIT",command=lambda:sub(a),bg='ivory4',fg='sno
w',font=('coureir','18')).place(y=660,x=200)

   def query(a,n,o,p,q,r):
      #to enter values in sql

      c=con.cursor()
      s="select * from {}".format(a)
      c.execute(s)
      d=c.fetchall()
      ro=c.rowcount

      #extracting value of id number
      if ro==0:
         m=1
      else:
         m=d[ro-1][0]+1

      #inserting all values
      s="insert into {} values ({},'{}','{}','{}','{}','{}')".format(a,m,n,o,p,q,r)
      c.execute(s)
      con.commit()
      show_all(a)


def delete_rec(a):
   #to delete a record
```

```
   #to make a new frame
   f=Frame(w,height=375,width=500,bg='gray11').place(x=0,y=350)
   #header
   Label(f,text="Enter the Record ID to be
deleted",height=6,width=20,bg='gray11',fg='snow',relief='flat',font=('
@Microsoft YaHei UI',20),wraplength=250).place(y=350,x=80)

   def sub(a):
      #extracting values
      y2=y.get()
      #to call the function
      query(a,int(y2))

   #asking what to enter
   Label(f,text="ID                                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=550,x=175)

   #defining a variable
   y=StringVar()

   #asking to enter

Entry(f,textvariable=y,fg='gray11',bg='snow',font=('coureir','18'),widt
h=5).place(y=550,x=225)
   #submit button

Button(f,text="SUBMIT",command=lambda:sub(a),bg='ivory4',fg='sno
w',font=('coureir','18')).place(y=650,x=200)

   def query(a,m):
      #to ask query

      #deleting from sql
      c=con.cursor()
      s="delete from {} where id={}".format(a,m)
      c.execute(s)
      con.commit()
```

```
        ro=c.rowcount

        #to show its deleted
        Label(f,text="{} records have been
deleted".format(ro),bg='gray11',fg='firebrick1',font=('coureir','14')).pl
ace(y=600,x=175 )

        #displaying all records
        show_all(a)


def update_menu(a):
    #to update menu

    def use_now(a,q,q1):
        #to update the particular option chosen for

        #creating new frame for each

f=Frame(w,height=200,width=500,bg="gray11").place(x=0,y=500)

        def sub(a,q):
            #extracting value of the id to be created
            x2=x.get()
            y2=y.get()

            #calling the function
            query(a,int(x2),y2,q)

        #giving instructions
        Label(f,text="Enter the record ID for which new {} is to be
entered".format(q1),bg='gray11',fg='snow',relief='flat',font=('@Micros
oft YaHei UI',12)).place(y=500,x=10)

        #asking for values
        Label(f,text="ID                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=550,x=50)
```

```
    Label(f,text="{}                                                    :
".format(q1),bg='gray11',fg='snow',font=('coureir','18')).place(y=600,x
=50)

    #declaring variable for entries
    x=StringVar()
    y=StringVar()

    #asking user value

Entry(f,textvariable=y,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=600,x=175)

Entry(f,textvariable=x,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=550,x=175)
    #letting user to submit values

Button(f,text="SUBMIT",command=lambda:sub(a,q),bg='ivory4',fg='s
now',font=('coureir','16')).place(y=675,x=200)

    def query(a,x,y,q):
        #updating in sql

        #quesry to be updated
        c=con.cursor()
        s="update {} set {}='{}' where id={}".format(a,q,y,x)
        c.execute(s)
        con.commit()

        #displaying number of records updated
        ro=c.rowcount
        Label(f,text="{}          records          have          been
updated".format(ro),bg='gray11',fg='firebrick1',font=('coureir','14')).p
lace(y=650,x=120)

        #displaying all records updated
        show_all(a)
```

```
    #creating a new frame
    f=Frame(w,height=375,width=500,bg='gray11').place(x=0,y=350)

    #header
    Label(f,text="Select what to update :
",bg='gray11',fg='snow',relief='flat',font=('@Microsoft YaHei
UI',16)).place(y=350,x=150)

    m=350
    n=75

    #giving choices
    Button(f,text="Website",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:use_now(a,'website','Website'),relief='ridge').place(y=m+50,x=n)
    Button(f,text="Username                                    "
,width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lamb
da:use_now(a,'username','Username'),relief='ridge').place(y=m+50,x
=n+160)
    Button(f,text="Password",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:use_now(a,'password','Password'),relief='ridge').place(y=m+100,x=
n-60)
    Button(f,text="Email ID", width=10,
bg='gray11',fg='snow',font=('coureir','18'),command=lambda:use_no
w(a,'emailid','Email ID'),relief='ridge').place(y=m+100,x=n+100)
    Button(f,text="Phone                                    No.",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:use_now(a,'phone_no','Phone
NO.'),relief='ridge').place(y=m+100,x=n+260)


def sort_menu(a):
    #to sort table by its components

    def g_user(a,m):
```

```
    #to display records in sorted form after userhas chose options


    #creating a new frame

f=Frame(w,height=800,width=1050,bg='gray11').place(x=500,y=0)
    #heading label
    Label(f,text="ALL                                                    RECORDS
ARE",bg='gray11',fg='snow',font=('mv
boli',40),bd=4).place(y=25,x=700)

    def sub(a):
      #to extract search value and call function
      z2=z.get()
      search_rec(a,z2)

    #declaring variable for search
    z=StringVar()
    #giving space to search

Entry(f,textvariable=z,fg='gray11',bg='snow',font=('coureir','18'),).pla
ce(y=125,x=1050)
    #calling search function using button

Button(f,image=pic,command=lambda:sub(a),bg="gray11",relief='fla
t').place(x=1300,y=120)
    #to go back to main table
    Button(f,text="BACK",
width=6,bg='gray11',fg='snow',font=('coureir','18'),command=lambda
:show_all(a),relief='ridge').place(y=125,x=600)

    #to extract values from sql in the order
    c=con.cursor()
    s="select * from {} order by {}".format(a,m)
    c.execute(s)
    d=c.fetchall()
```

```
    m=200
    n=550

    #displaying heading

Label(f,text='ID',width=3,fg='gray11',bg='honeydew3',font=('@micros
oft yahei ui','12'),pady=2,).place(y=m,x=n-45)

Label(f,text='WEBSITE',width=14,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n-5)

Label(f,text='USERNAME',width=18,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+145)

Label(f,text='PASSWORD',width=13,fg='gray11',bg='honeydew3',font
=('@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+340)

Label(f,text='EMAILID',width=27,fg='gray11',bg='honeydew3',font=('
@microsoft yahei ui','12'),pady=2,).place(y=m,x=n+490)
    Label(f,text='PHONE
NO',width=11,fg='gray11',bg='honeydew3',font=('@microsoft    yahei
ui','12'),pady=2,).place(y=m,x=n+775)
    m+=50

    #displaying all records
    for i in d:

        Label(f,text=str(i[0]),width=3,
fg='gray11',bg='mintcream',font=('coureir','12'),pady=2,).place(y=m,x
=n-45)

Label(f,text=str(i[1]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n-5)

Label(f,text=str(i[2]),width=20,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+145)
```

```
Label(f,text=str(i[3]),width=15,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+340)

Label(f,text=str(i[4]),width=30,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+490)

Label(f,text=str(i[5]),width=12,fg='gray11',bg='mintcream',font=('cou
reir','12'),pady=2,).place(y=m,x=n+775)
        m+=30

   #creating a new frame
   f=Frame(w,height=375,width=500,bg='gray11').place(x=0,y=350)

   #header
   Label(f,text="Select what to sort by :
",bg='gray11',fg='snow',relief='flat',font=('@Microsoft YaHei
UI',16)).place(y=350,x=150)
   m=350
   n=175

   #giving choices
   Button(f,text="Website",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:g_user(a,'website'),relief='ridge').place(y=m+50,x=n)
   Button(f,text="Username                                    "
,width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lamb
da:g_user(a,'username'),relief='ridge').place(y=m+100,x=n)
   Button(f,text="Password",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:g_user(a,'password'),relief='ridge').place(y=m+150,x=n)
   Button(f,text="Email ID", width=10,
bg='gray11',fg='snow',font=('coureir','18'),command=lambda:g_user(
a,'emailid'),relief='ridge').place(y=m+200,x=n)
   Button(f,text="Phone                                    no.",
width=10,bg='gray11',fg='snow',font=('coureir','18'),command=lambd
a:g_user(a,'phone_no'),relief='ridge').place(y=m+250,x=n)
```

```
def menu(a):

   #creating a new frame
   f=Frame(w,height=800,width=500,bg="gray11").place(x=0,y=0)

   #creating menu options
   Button(f,text="MENU",fg='white',bg='gray11',bd=4,font=('mv
boli',36),command=lambda:menu(a),relief='flat').place(y=0,x=150)

   m=130
   n=50

   #creating all buttons and showing table
   show_all(a)
   Button(f,text="GO TO PASSWORD CHECK UP",
bg="gray26",fg="light cyan",relief='groove',font=('yu
gothic',18),command=lambda:p_check(a)).place(y=m,x=n+20)
   Button(f,text="ADD NEW ",bg="gray26",fg="light
cyan",relief='groove',font=('yu
gothic',18),width=10,command=lambda:insert_new(a)).place(y=m+7
5,x=n)
   Button(f,text="DELETE ",bg="gray26",fg="light
cyan",relief='groove',font=('yu
gothic',18),width=10,command=lambda:delete_rec(a)).place(y=m+7
5,x=n+250)
   Button(f,text="UPDATE",bg="gray26",fg="light
cyan",relief='groove',font=('yu
gothic',18),width=10,command=lambda:update_menu(a)).place(y=m
+150,x=n)
   Button(f,text="SORT BY ",bg="gray26",fg="light
cyan",relief='groove',font=('yu
gothic',18),width=10,command=lambda:sort_menu(a)).place(y=m+1
50,x=n+250)

   #to log out
```

```
    Button(f,text="LOGOUT",bg="gray26",fg="light
cyan",relief='groove',font=('yu
gothic',18),width=10,command=lambda:main()).place(y=m+600,x=n
+125)


def log_in():
    #to login the user

    #creating a new frame

f=Frame(w,height=400,width=1550,bg="gray11").place(x=0,y=400)

    #giving options like before
    Button(f,text="LOG                    IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:log_i
n(),relief='flat').place(y=400,x=200)
    Button(f,text="SIGN                    IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:sign
_in(),relief='flat').place(y=600,x=200)

    #asking for user to typer username and password
    Label(f,text="Username                                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=450,x=900)
    Label(f,text="Password                                                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=550,x=900)

    def sub():
        #to get variables and go for next function
        x2=x.get()
        y2=y.get()
        check(x2,y2)

    #calling variables
    x=StringVar()
    y=StringVar()
```

```
   #giving entry options

Entry(f,textvariable=x,fg='gray11',bg='snow',font=('coureir','18')).plac
e(y=450,x=1050)

Entry(f,textvariable=y,show="*",fg='gray11',bg='snow',font=('coureir'
,'18')).place(y=550,x=1050)
   #button to get the variables

Button(w,text="SUBMIT",command=lambda:sub(),bg='gray31',fg='sn
ow',font=('coureir','18'),width=10,height=2).place(y=650,x=1000)

   def check(x,y):
      #to check username and passowrd and log in the user

      #opening the file
      a=open('users.csv','r')
      b=csv.reader(a)

      #to check if username is there
      for i in b:

         if x.lower()==i[0]:

            if y==i[1]:
               #to check if its same as password
               menu(x)
               #calling the menu, x is the username of the person and also
the table name for the user
               break

            else:
               #to display message if passowrd is worng
               Label(f,text="Wrong Password entered! Please try
again.",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=600,x
=925)
               break
```

```
        else:
            #if username is not there in the file
            Label(f,text="Username doesn't exist! Please try
again.",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=500,x
=925)

        a.close()
        #closing the csv file


def sign_in():
    #to sign in the user

    #creating a new frame

f=Frame(w,height=400,width=1550,bg="gray11").place(x=0,y=400)

    #giving options like before
    Button(f,text="LOG                IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:log_i
n(),relief='flat').place(y=400,x=200)
    Button(f,text="SIGN                IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:sign
_in(),relief='flat').place(y=600,x=200)

    #asking user to choose a username
    Label(f,text="Choose                Username                :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=450,x=950)

    def sub():
        #extracting the value
        x2=x.get()
        check(x2)



    #declaring variable for username
```

```
    x=StringVar()

    #asking for it

Entry(f,textvariable=x,fg='gray11',bg='snow',font=('coureir','18')).plac
e(y=500,x=950)
    #button to get the variables

Button(f,text="SUBMIT",command=lambda:sub(),bg='gray31',fg='sno
w',font=('coureir','18'),width=10,height=2).place(y=650,x=975)

    def password(a):
        #to let user take password and enter it in a file

        #creating a new frame for passoword

f=Frame(w,height=400,width=650,bg="gray11").place(x=900,y=400)
        #asking user to enter values
        Label(f,text="Create                    Password                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=450,x=850)
        Label(f,text="Confirm                    Password                    :
",bg='gray11',fg='snow',font=('coureir','18')).place(y=550,x=850)

        def sub():
            #extracting values
            x2=x.get()
            y2=y.get()
            #to check if password is okay
            check(x2,y2)

        #assigning variables
        x=StringVar()
        y=StringVar()

        #giving entry options
```

```
Entry(f,textvariable=x,fg='gray11',bg='snow',font=('coureir','18'),show
="*").place(y=450,x=1100)

Entry(f,textvariable=y,show="*",fg='gray11',bg='snow',font=('coureir'
,'18')).place(y=550,x=1100)
    #button to get the variables

Button(w,text="SUBMIT",command=lambda:sub(),bg='gray31',fg='sn
ow',font=('coureir','18'),width=10,height=2).place(y=650,x=1000)


    def create_table(a):
        #creating table in sql

        c=con.cursor()
        s="""create table {}(
    id int ,
    website varchar(50),
    username varchar(50),
    password varchar(50),
    emailid varchar(50),
    phone_no varchar(12))""".format(a)
        c.execute(s)

    def entry(x,y):
        #to enter username and password in csv file

        #openingthe file
        p=open('users.csv','a',newline='')
        q=csv.writer(p)

        q.writerow([x.lower(),y])

        #creating table for that username in database
        create_table(x)
```

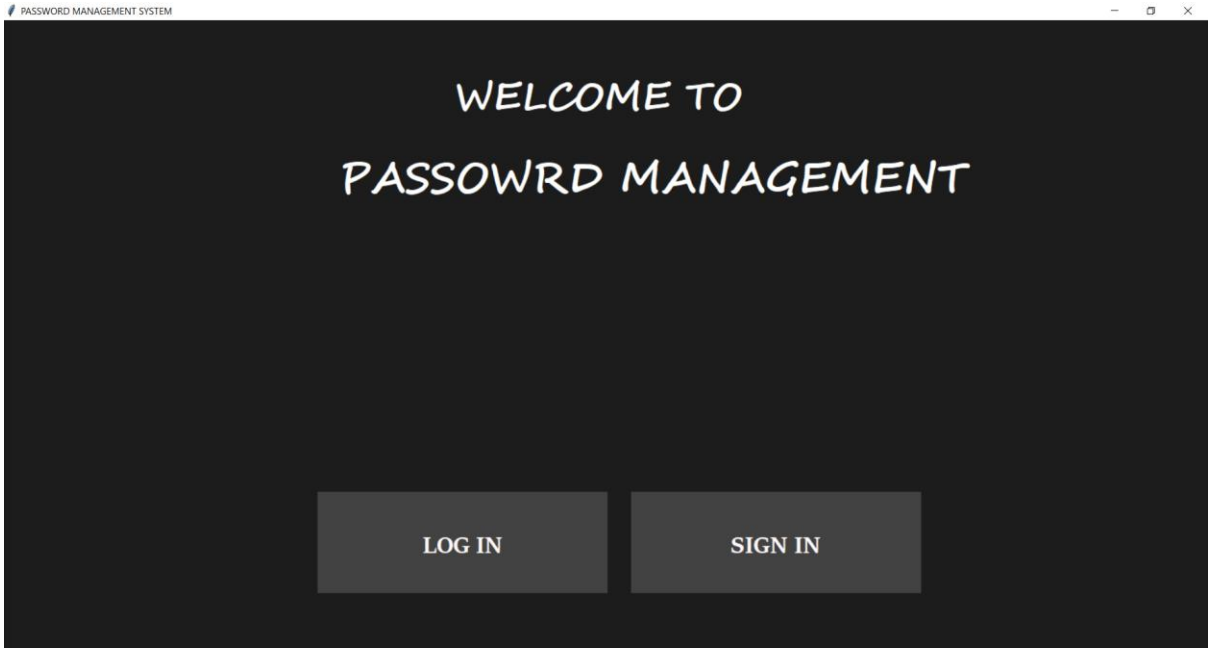
```
        #letting the person go to menu after that
        menu(x)

        #closing the file
        p.close()

    def check(x,y):

        #to check if password is minimum of 8 characters
        if len(x)>7:

            if x==y:
                #to check if passwords are same
                entry(a,y)
                #to enter in the csv file

            else:
                Label(f,text="Passwords dont match! Please try
again!",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=600,x
=950)

        else:
            Label(f,text="Length of password should be greater than
7.",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=500,x=95
0)


  def check(x):
    #checks if username already exists or not

    #to open the file
    a=open('users.csv','r')
    b=csv.reader(a)

    for i in b:

        if len(x)<1:
```

```
            #checks if its length is greater than 1
            Label(f,text="Username length should be more than 1.
",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=550,x=950)
            break

        else:
            if x[0].isdigit():
                #checks if it starts with alphabet
                Label(f,text="Username should always start with an
alphabet.",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=55
0,x=950)
                break

            else:
                if x.lower()==i[0]:
                    #checks in file
                    Label(f,text="Username already taken! Please try again.
",bg='gray11',fg='firebrick1',font=('coureir','14')).place(y=550,x=950)
                    break
        else:
            #if username is allotted it goes to check for password
            password(x)
        #closing the file
        a.close()

def create_userfile():
    a=open('users.csv','w',newline='')
    b=csv.writer(a)
    b.writerow(['username','password'])

def main():
    #1st page to start up with

    #creating a frame
    f=Frame(w,height=800,width=1550,bg='gray11').place(x=0,y=0)

    #to print the welcome message and give it a aesthetic look
```

```
   Label(f,text="WELCOME TO ",font=("Segoe
Script",36,"bold"),height=1,width=40,bg='gray11',fg='white',bd=4,pa
dx=4,pady=4,relief='flat').place(y=50,x=100)
   Label(f,text="PASSOWRD MANAGEMENT",font=("Segoe
Script",40,"bold"),height=1,width=40,bg='gray11',fg='white',bd=4,pa
dx=4,pady=4,relief='flat').place(y=150,x=100)

   #giving user the choices
   Button(f,text="LOG IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:log_i
n(),relief='flat').place(y=600,x=400)
   Button(f,text="SIGN IN",width=20,height=2,font=("sitka
heading",24,"bold"),bg="gray26",fg="snow",command=lambda:sign
_in(),relief='flat').place(y=600,x=800)


#create_userfile()
#to be called only when everything needs to be refreshed
#make sure that database is cleared so tables dont clash with each
other


#cteating window
w=Tk()
#adding background to window
w.configure(bg='GRAY11')
#adding title to window
w.title("PASSWORD MANAGEMENT SYSTEM")
#setting up size of the window
w.geometry('1550x800')


#importing a pic for search
pic=PhotoImage(file="sea.png")

#main to start the function
main()
```

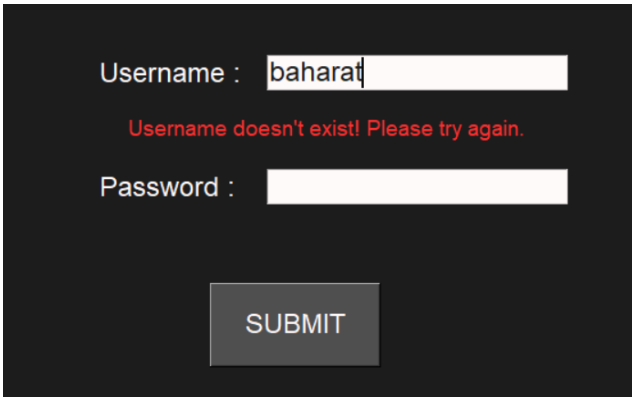
SAMPLE OUTPUTS

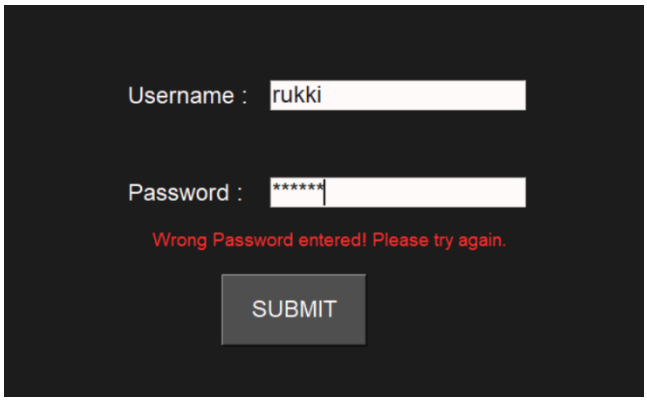
Home page:



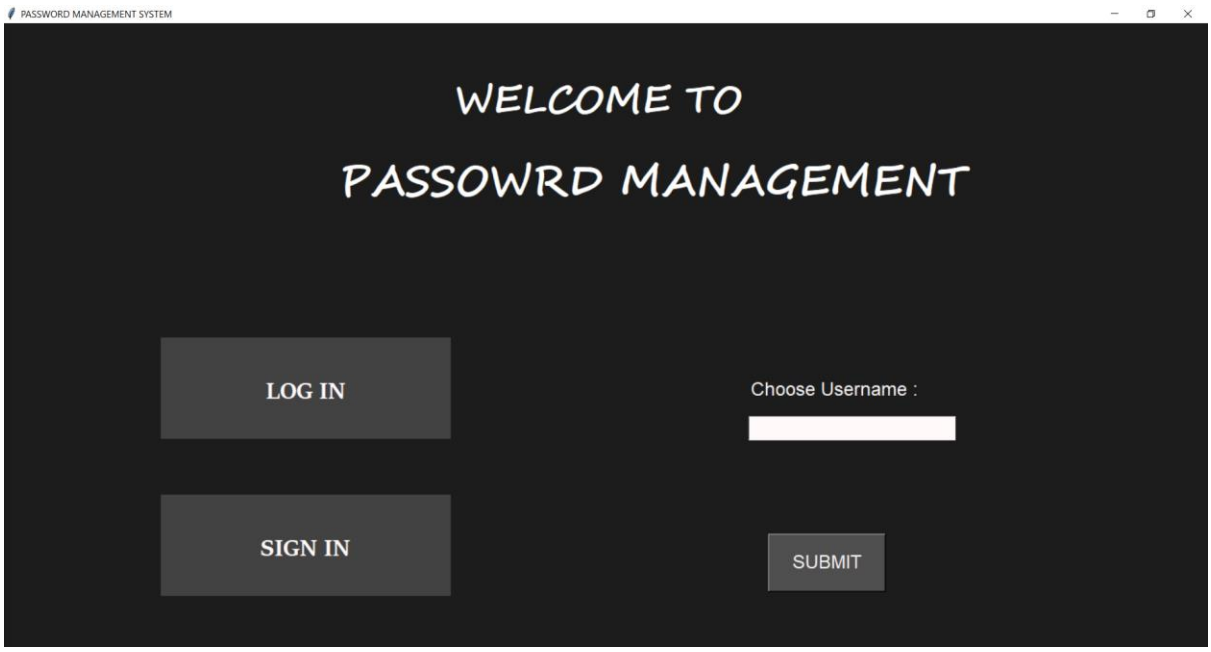
Login page:

NOTE:
A **‘Please try again’** message appears if username is invalid



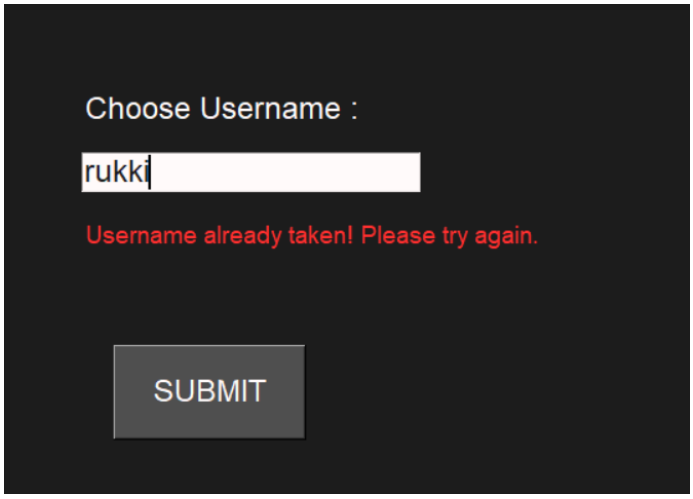


sign up page:

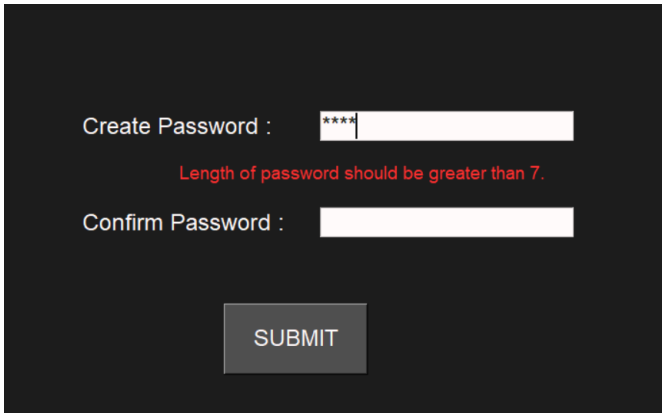


A **‘Please try again’** message appears if

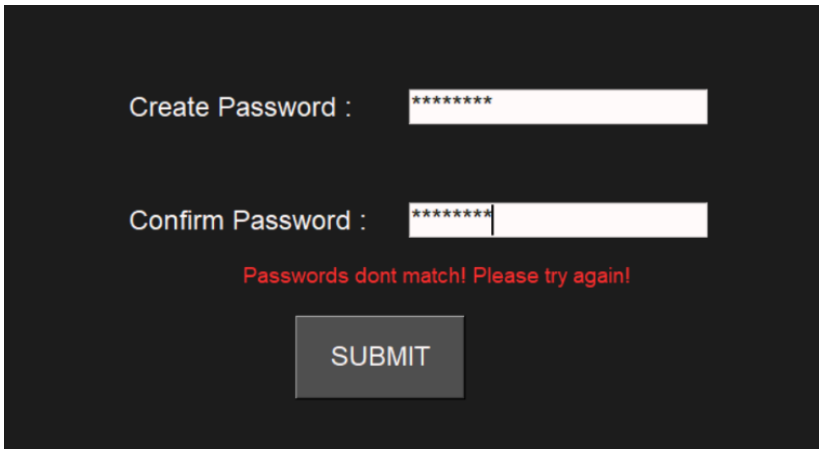
1. Username already exists
2. Username starts with a number
3. Username is a white space



NOTE:
Length of your password must be more than 7 characters

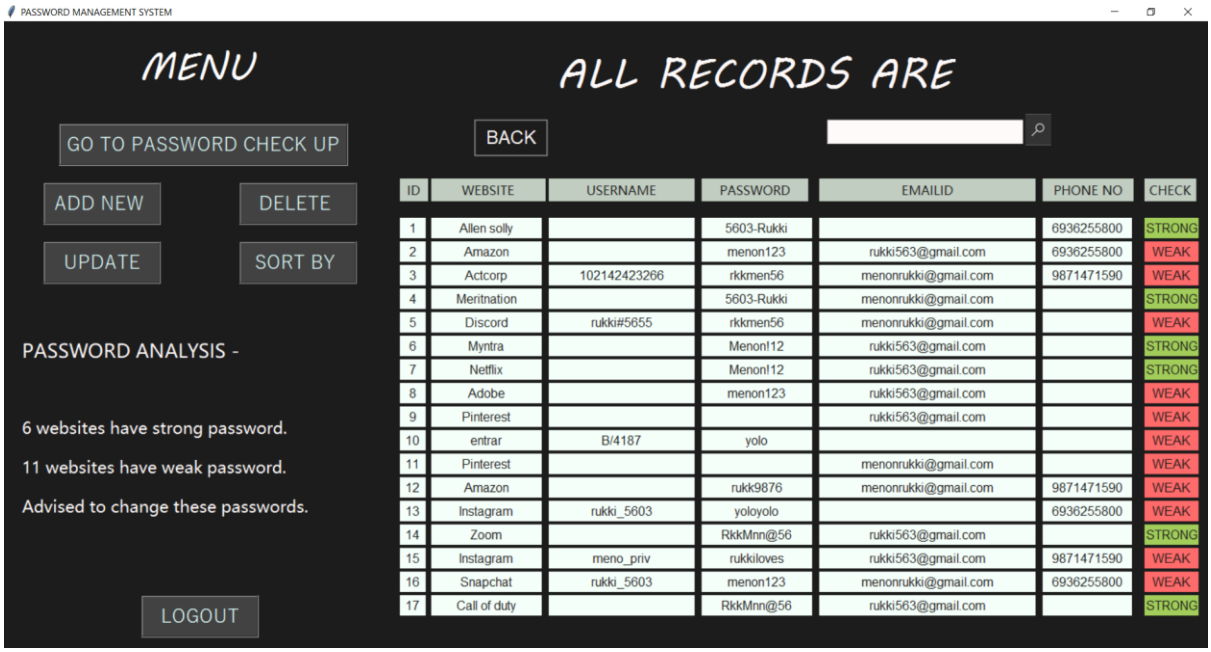


When confirming your password, **'Please try again'** message appears if password is not the same as created password



Menupage:

Password check up – to check if your password is strong or weak



NOTE:

Click on **'BACK'** to go back to menu page

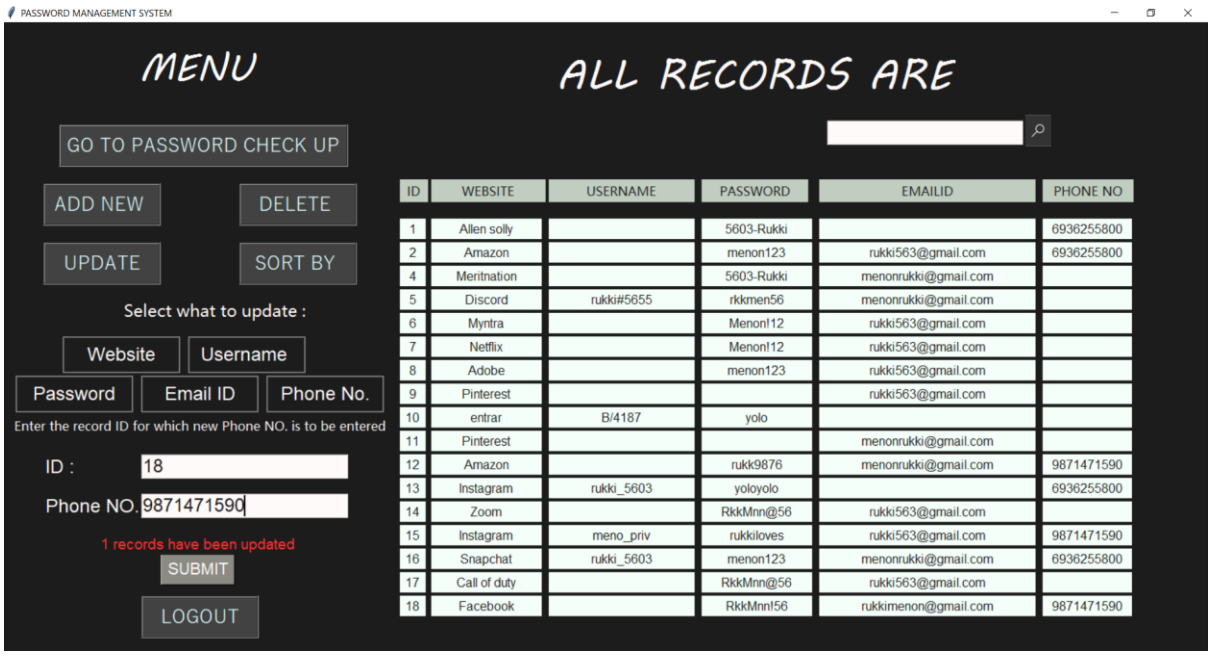
Adding a new record - after submitting record values, new record gets added at the end of the table

ID	WEBSITE	USERNAME	PASSWORD	EMAILID	PHONE NO
1	Allen solly		5603-Rukki		6936255800
2	Amazon		menon123	rukki563@gmail.com	6936255800
3	Actcorp	102142423266	rkkmen56	menonrukki@gmail.com	9871471590
4	Meritnation		5603-Rukki	menonrukki@gmail.com	
5	Discord	rukki#5655	rkkmen56	menonrukki@gmail.com	
6	Myntra		Menon!12	rukki563@gmail.com	
7	Netflix		Menon!12	rukki563@gmail.com	
8	Adobe		menon123	rukki563@gmail.com	
9	Pinterest			rukki563@gmail.com	
10	entrar	B/4187	yolo		
11	Pinterest			menonrukki@gmail.com	
12	Amazon		rukk9876	menonrukki@gmail.com	9871471590
13	Instagram	rukki_5603	yoloyolo		6936255800
14	Zoom		RkkMnn@56	rukki563@gmail.com	
15	Instagram	meno_priv	rukkiloves	rukki563@gmail.com	9871471590
16	Snapchat	rukki_5603	menon123	menonrukki@gmail.com	6936255800
17	Call of duty		RkkMnn@56	rukki563@gmail.com	
18	Facebook		RkkMnn!56	rukkimenon@gmail.com	

Deleting a record- required record gets deleted after submitting record ID

ID	WEBSITE	USERNAME	PASSWORD	EMAILID	PHONE NO
1	Allen solly		5603-Rukki		6936255800
2	Amazon		menon123	rukki563@gmail.com	6936255800
4	Meritnation		5603-Rukki	menonrukki@gmail.com	
5	Discord	rukki#5655	rkkmen56	menonrukki@gmail.com	
6	Myntra		Menon!12	rukki563@gmail.com	
7	Netflix		Menon!12	rukki563@gmail.com	
8	Adobe		menon123	rukki563@gmail.com	
9	Pinterest			rukki563@gmail.com	
10	entrar	B/4187	yolo		
11	Pinterest			menonrukki@gmail.com	
12	Amazon		rukk9876	menonrukki@gmail.com	9871471590
13	Instagram	rukki_5603	yoloyolo		6936255800
14	Zoom		RkkMnn@56	rukki563@gmail.com	
15	Instagram	meno_priv	rukkiloves	rukki563@gmail.com	9871471590
16	Snapchat	rukki_5603	menon123	menonrukki@gmail.com	6936255800
17	Call of duty		RkkMnn@56	rukki563@gmail.com	
18	Facebook		RkkMnn!56	rukkimenon@gmail.com	

Updating a record- choose the column to be updated and updated record will be displayed after clicking on 'SUBMIT'

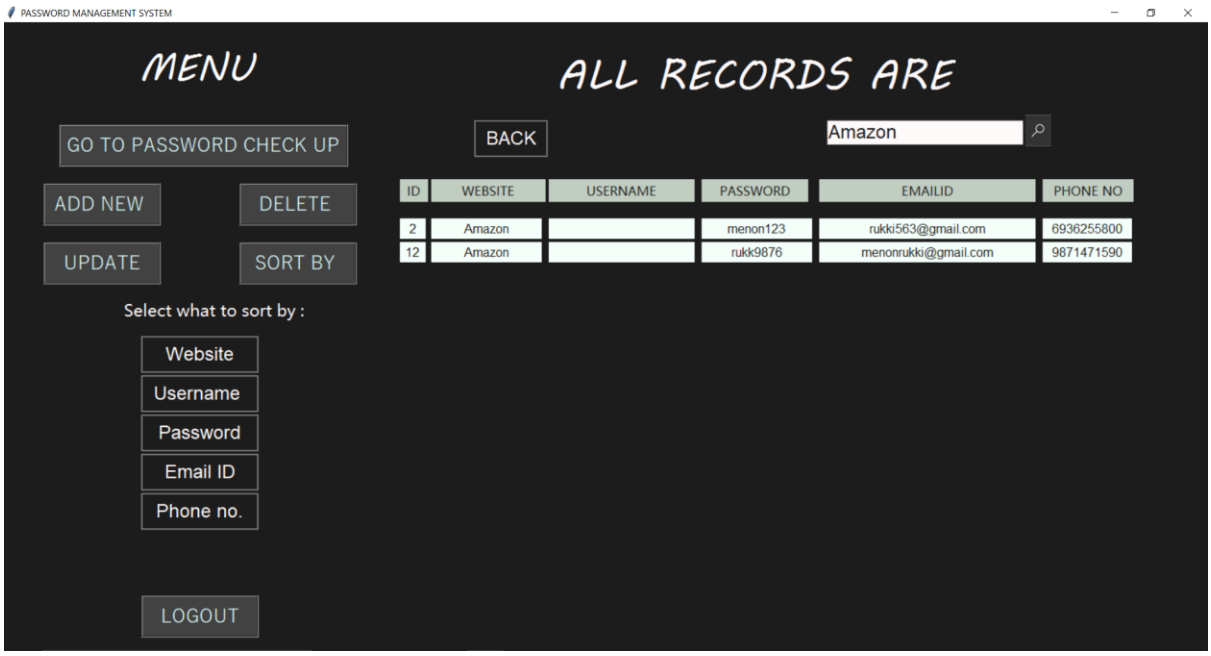


Sorting a Record- To sort records by a particular column, choose the column with respect to which you would like to sort the table (in this case website)

NOTE:

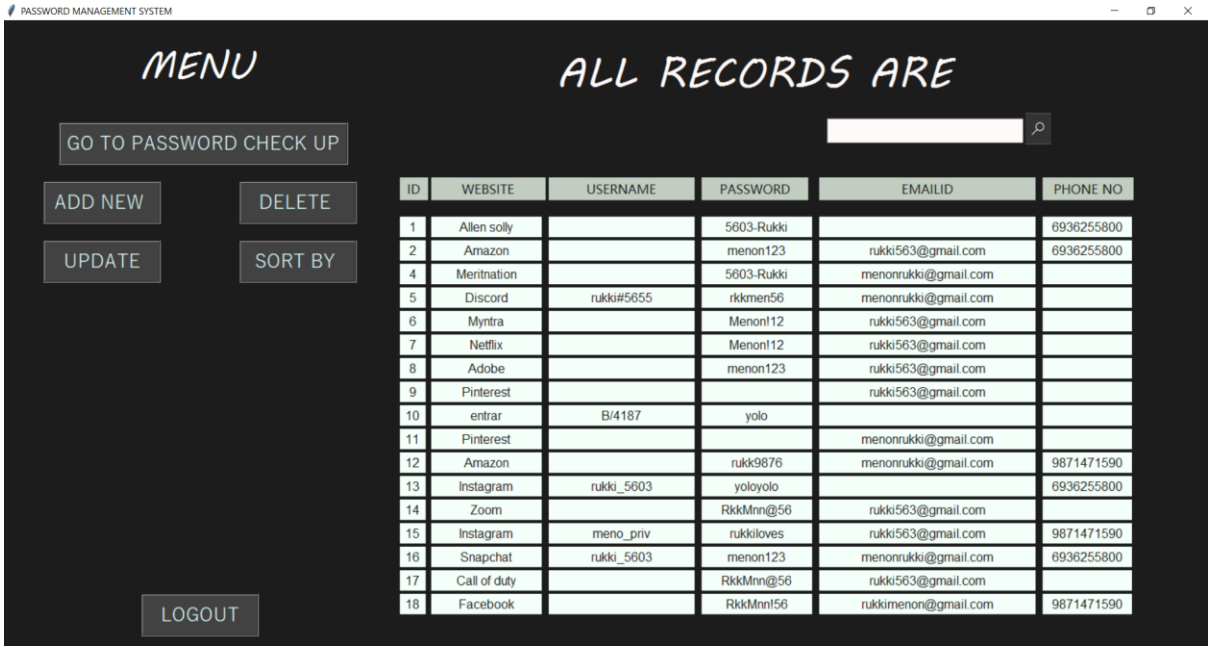
Click on **'BACK'** button to display the table sorted by ID

Search- searches for the required item in all columns (in this case Amazon)



NOTE:
Click on **'BACK'** to return to original table

The **MENU** button on left of the screen is a hidden refresh button for convenience and updating the password management system



Click on '**Logout**' to return to home page

REFERENCES

In order to work on this project, the following books and literature were referred by us during the various phases of development of the project.

1) http://www.mysql.org/
2) http://www.python.org/
3) Computer Science for class XII –by Sumita Arora
4) Various websites of discussion forum software development activities.
5) tkinter-

    a. basic- https://realpython.com/python-gui-tkinter/

    b. window- https://pythonexamples.org/python-tkinter-window-background-color/

    c. frames- https://www.tutorialspoint.com/python/tk_frame.htm

    d. label- https://www.tutorialspoint.com/python/tk_label.htm

    e. button-

https://www.tutorialspoint.com/python/tk_button.htm

f. font- https://www.tutorialspoint.com/python/tk_fonts.htm#:~:text=Font%20object%20Fonts&text=weight%20%E2%88%92%20%22bold%22%20for%20boldface,overstruck%20text%2C%200%20for%20normal.

g. colors- http://www.science.smith.edu/dftwiki/images/3/3d/TkInterColorCharts.png

h. styles- https://stackoverflow.com/questions/39614027/list-available-font-families-in-tkinter

Other than the above-mentioned books, the suggestions and supervision of my teacher and my class experience also helped me to develop this software project.