SOMERVILLE SCHOOL, GREATER NOIDA

COMPUTER SCIENCE (CODE 083)

PROJECT FILE

PROJECT NAME –

PASSWORD MANAGER

NAME : ABHISHEK GUPTA

CLASS : XII

SECTION : C

SESSION : 2023-24

SUBMITTED TO : MRS. SHALLY BHATIA

CERTIFICATE

This is to certify that ABHISHEK GUPTA of class XII-C has successfully completed and submitted his Computer Science (083) project report entitled 'PASSWORD MANAGER' in the academic year 2023-24 under the guidance of his Computer Science teacher, MRS. SHALLY BHATIA. This project was undertaken and prepared by him only, and is the result of his personal efforts.

Principal Signature	Subject Teacher Signature

ACKNOWLEDGEMENT

I would like to express a deep sense of thanks & gratitude to my Computer Science teacher, Mrs. Shally Bhatia ma'am for guiding me immensely through the course of the project. She always evinced keen interest in my work. Her constructive advice & constant motivation have been responsible for the successful completion of this project. My sincere thanks goes to Our principal ma'am, Dr. Mary Thomas for her co-ordination in extending every possible support for the completion of this project.

I also extend my heartfelt gratitude to my parents for their motivation & support. I also thank my classmates for their timely help, support and coordination during the development of this project. Last but not the least, I would like to thank all those who had helped directly or indirectly towards the completion of this project.

-ABHISHEK GUPTA

XII C

INDEX

CONTENTS **PAGE NO**

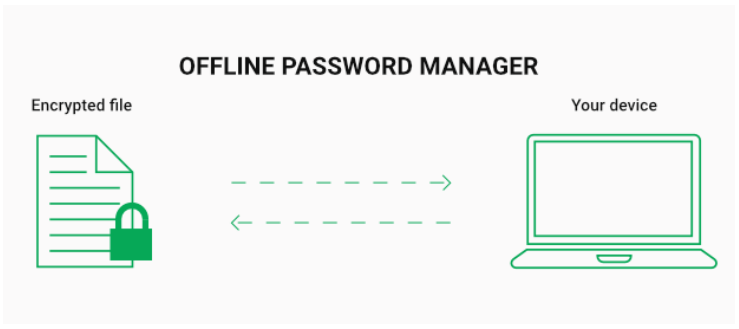
1. Introduction to password managers ……………….1-2
2. Project Overview……………………………………………..3-5
3. Source code…………………………………………………6-11
4. Output screens……………………………………….…12-14
5. Bibliography…………………………………………………15

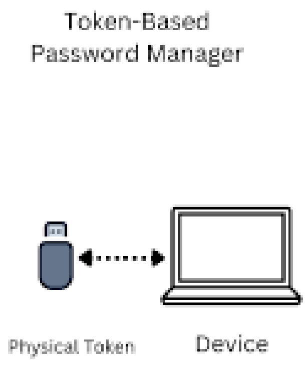
INTRODUCTION TO PASSWORD MANAGERS

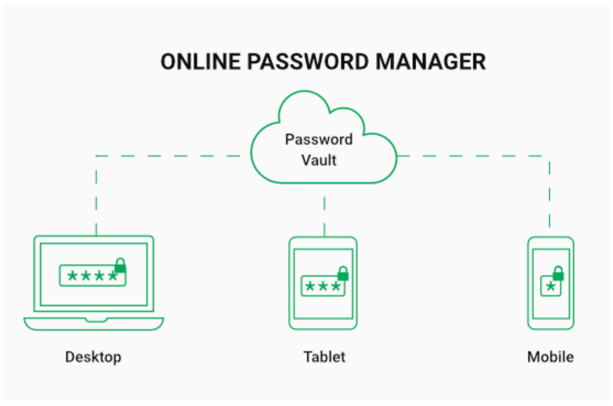
A **password manager** is a software application that is used to store and manage the passwords that a user has for various online accounts and security features. Password managers store the passwords in an encrypted format and provide secure access to all the password information with the help of a master password.

Fundamental types of password managers are:

1. Web browser-based
2. Cloud-based
3. Desktop
4. Portable
5. Stateless
6. Token-based



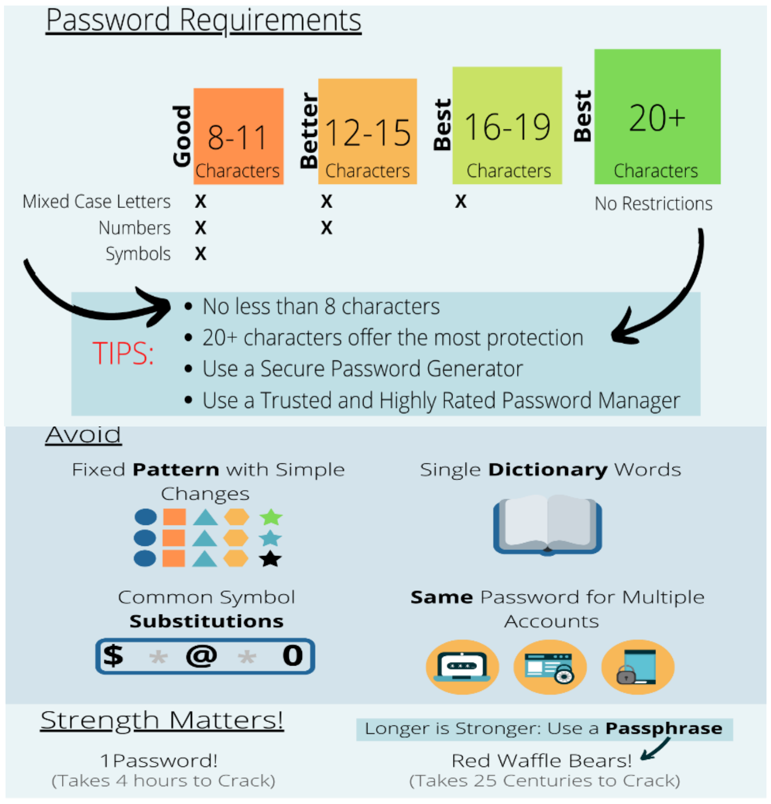




->Main features of different password managers include:

- **Password generator**
  lets you create strong and unique passwords or pass-phrases with various variables such as password length, numbers, letters, capital letters and special characters.
- **Check passwords and if they were in any recent breach**
  helpful to check if you need to change your password and take security measures.
- **Import or Export your data**
  This feature lets you import or export your passwords in different encrypted file formats. This is a useful feature if you need to keep a backup of your passwords or if you want to move your passwords from one password manager application to another.
- **Autofill feature**
  - presents the required login credential you have saved in a password manager with just one click, this feature saves your time by avoiding multiple clicks of copying and pasting your username and password on every website.
- **Multi Factor Authentication**
  - MFA is a multi-step account login process that requires users to enter more information than just a password. For example, along with the password, users might be asked to enter a code sent to their email, answer a secret question, or scan a fingerprint.

PASSWORD HYGIENE


Password Requirements
Good 8-11 Characters
Better 12-15 Characters
Best 16-19 Characters
Best 20+ Characters
Mixed Case Letters X X X
Numbers X X
Symbols X
No Restrictions
TIPS:
• No less than 8 characters
• 20+ characters offer the most protection
• Use a Secure Password Generator
• Use a Trusted and Highly Rated Password Manager
Avoid
Fixed Pattern with Simple Changes
Single Dictionary Words
Common Symbol Substitutions
$ * @ * 0
Same Password for Multiple Accounts
Strength Matters!
1Password!
(Takes 4 hours to Crack)
Longer is Stronger: Use a Passphrase
Red Waffle Bears!
(Takes 25 Centuries to Crack)

PROJECT OVERVIEW

->Softwares/languages used:-

1. Python 3.11.2 and python libraries
2. MySql version 8.0.34
3. Notepad/any text editor

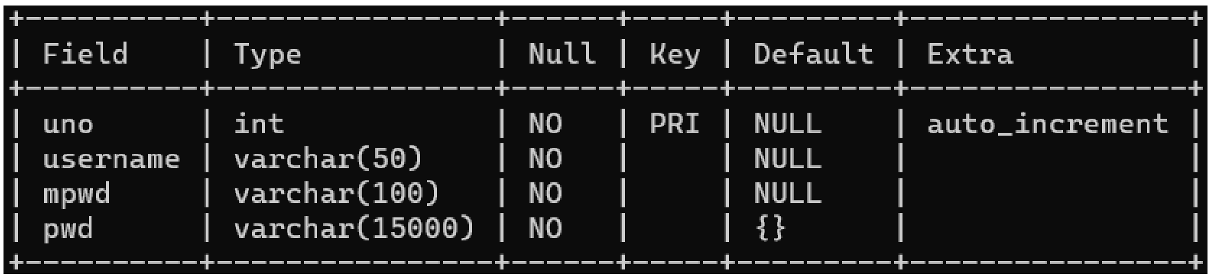
->Main python files/ user created modules:

1. pass_manager_main.py – acts as the interface between user and backend
2. passfunction.py – acts as gateway to the backend

->Modules Imported:

1. pass_manager_main.py imports :-
   a. time
   b. maskpass
   c. passfunction
2. passfunction.py imports:-
   a. Keyboard – read_key(), add_hotkey()
   b. Maskpass – advpass()
   c. Pickle
   d. Mysql.connector
   e. Time → time(),ctime()
   f. Random → randrange()

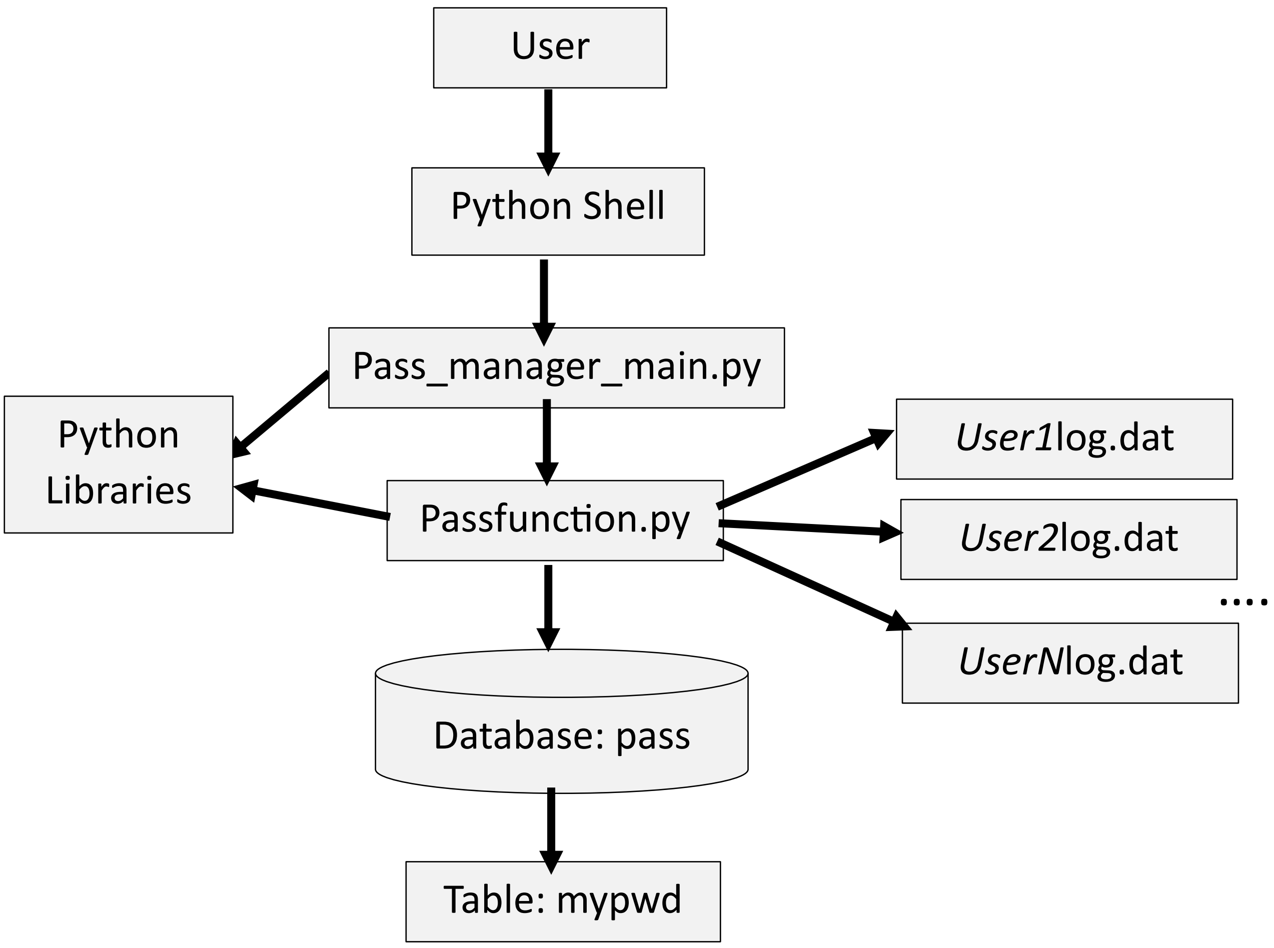
MySql structure: Database pass → table mypwd



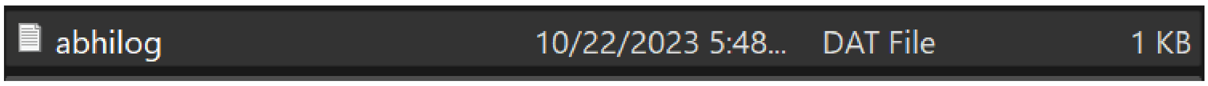
```
+----------+----------------+------+-----+---------+----------------+
| Field    | Type           | Null | Key | Default | Extra          |
+----------+----------------+------+-----+---------+----------------+
| uno      | int            | NO   | PRI | NULL    | auto_increment |
| username | varchar(50)    | NO   |     | NULL    |                |
| mpwd     | varchar(100)   | NO   |     | NULL    |                |
| pwd      | varchar(15000) | NO   |     | {}      |                |
+----------+----------------+------+-----+---------+----------------+
```

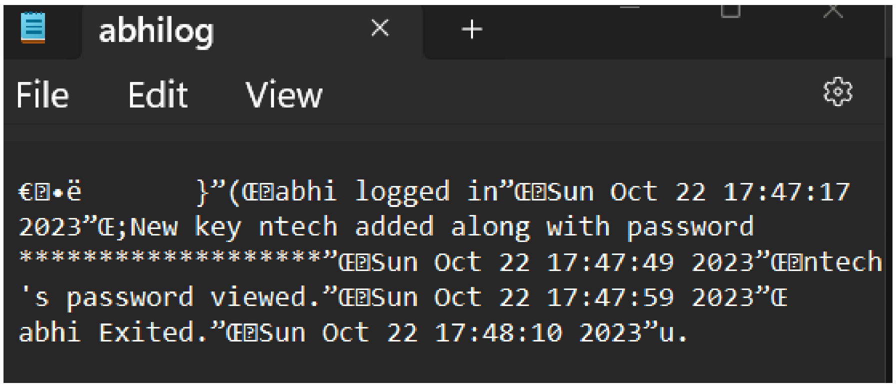
1. uno – stores user Id (primary key of mypwd)
2. username – stores unique usernames signed up on pass_manager_main.py
3. mpwd – stores a single master password of each user. It is the key to access a user's vault
4. pwd – stores accountname/keyname along with passwords as key-value pair in a single dictionary. This dictionary acts as the user's vault.

Structure of entire Password Manager Application



-> Binary .dat files created on creation of every new user.
Example:


abhilog 10/22/2023 5:48... DAT File 1 KB

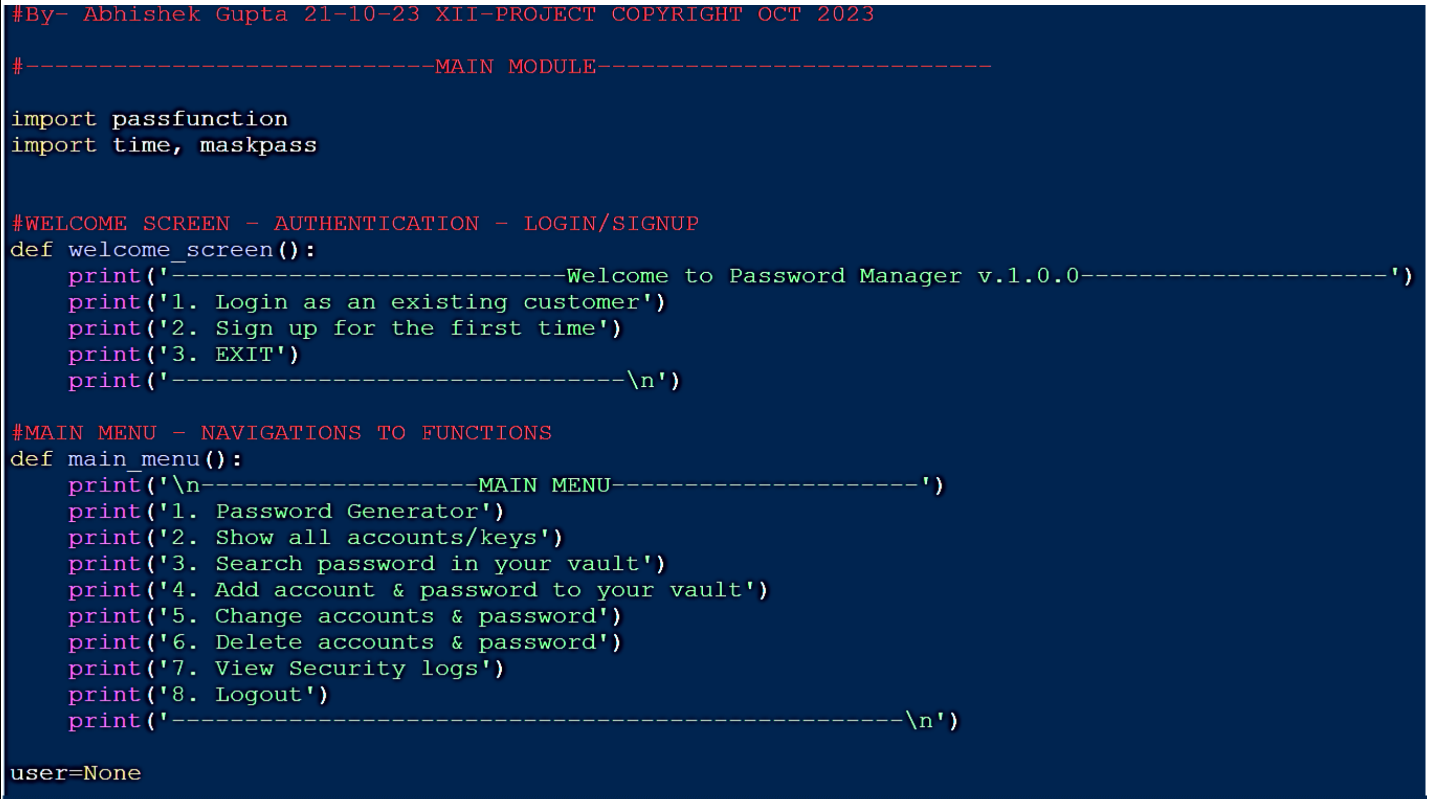


```
€•ë       }”(Œabhi logged in”ŒSun Oct 22 17:47:17
2023”Œ;New key ntech added along with password
********************”ŒSun Oct 22 17:47:49 2023”Œntech
's password viewed.”ŒSun Oct 22 17:47:59 2023”Œ
abhi Exited.”ŒSun Oct 22 17:48:10 2023”u.
```

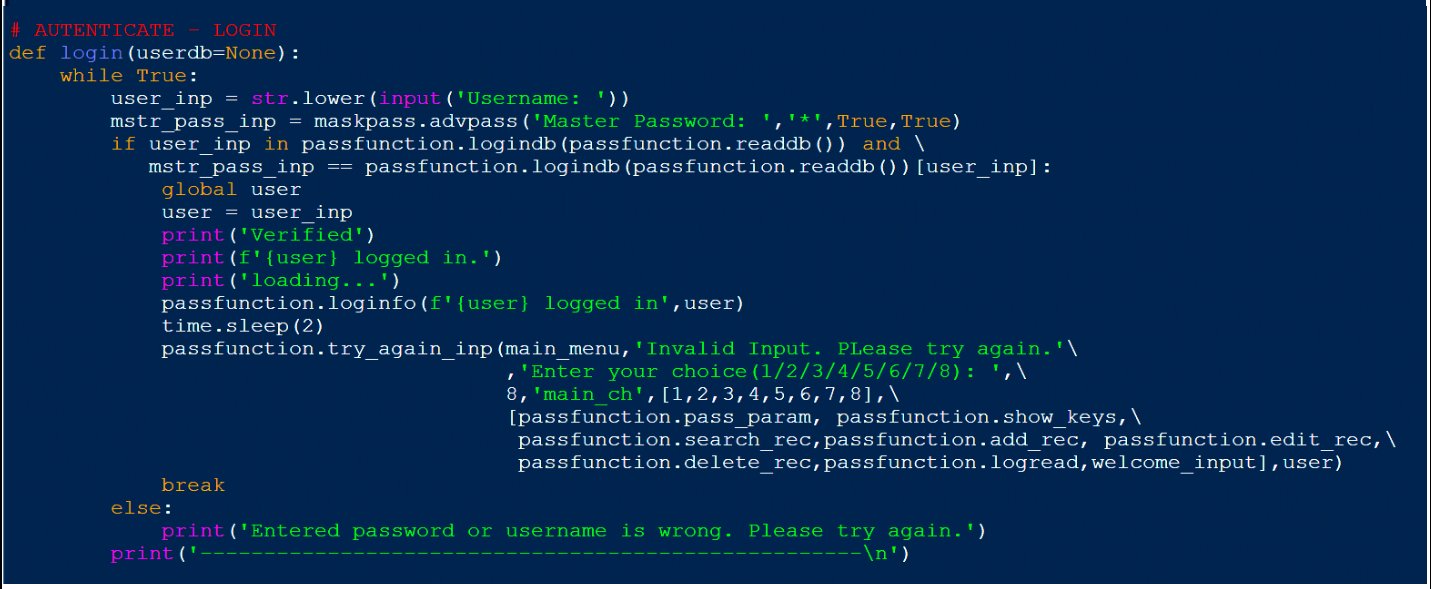
Contents of .dat (binary file) stored in non-human understandable form

SOURCE CODE

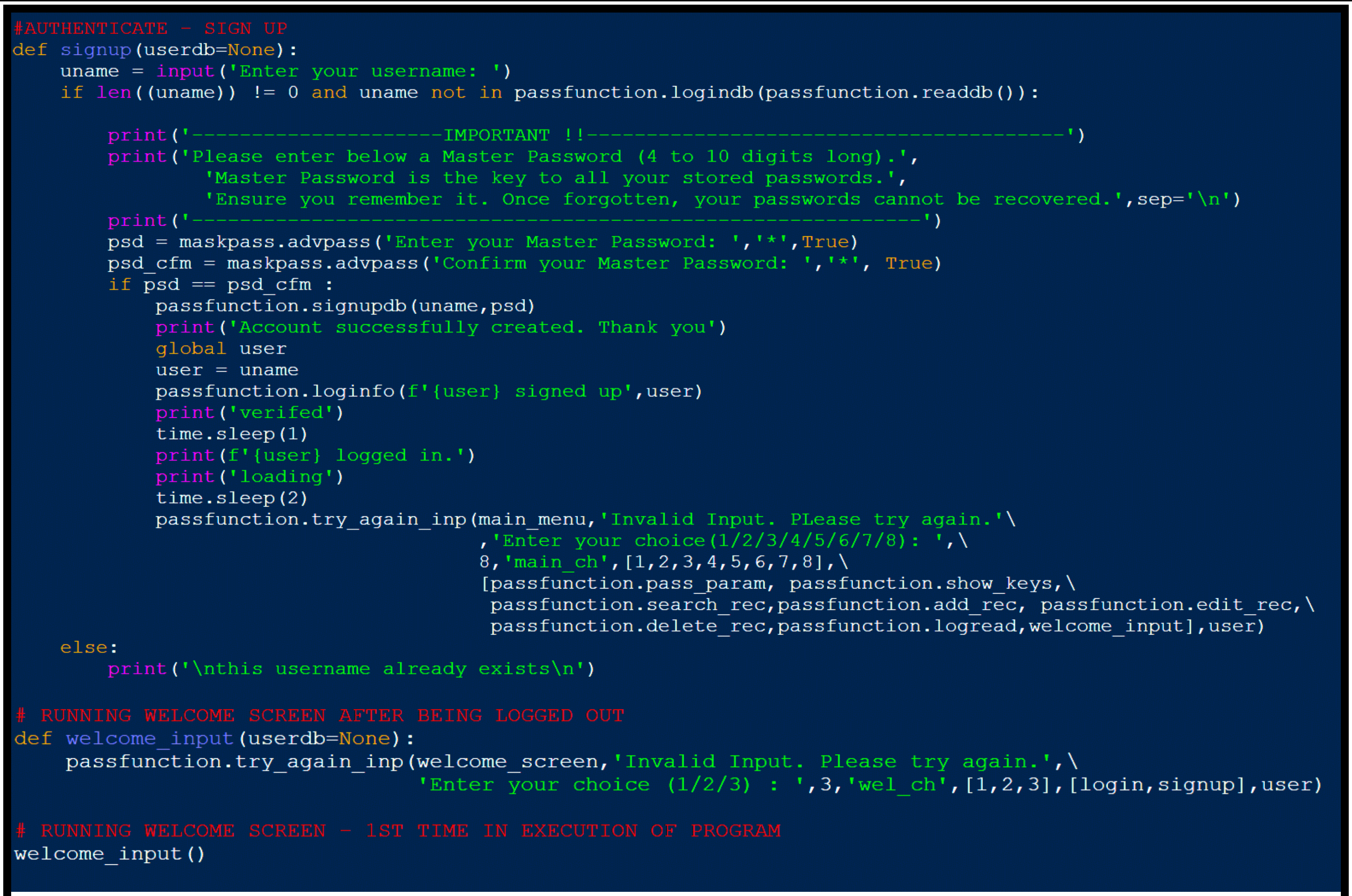
1. pass_manager_main.py



```python
#By- Abhishek Gupta 21-10-23 XII-PROJECT COPYRIGHT OCT 2023

#----------------------------MAIN MODULE---------------------------

import passfunction
import time, maskpass


#WELCOME SCREEN - AUTHENTICATION - LOGIN/SIGNUP
def welcome_screen():
    print('---------------------------Welcome to Password Manager v.1.0.0----------------------')
    print('1. Login as an existing customer')
    print('2. Sign up for the first time')
    print('3. EXIT')
    print('-------------------------------\n')

#MAIN MENU - NAVIGATIONS TO FUNCTIONS
def main_menu():
    print('\n--------------------MAIN MENU---------------------')
    print('1. Password Generator')
    print('2. Show all accounts/keys')
    print('3. Search password in your vault')
    print('4. Add account & password to your vault')
    print('5. Change accounts & password')
    print('6. Delete accounts & password')
    print('7. View Security logs')
    print('8. Logout')
    print('--------------------------------------------------\n')

user=None
```

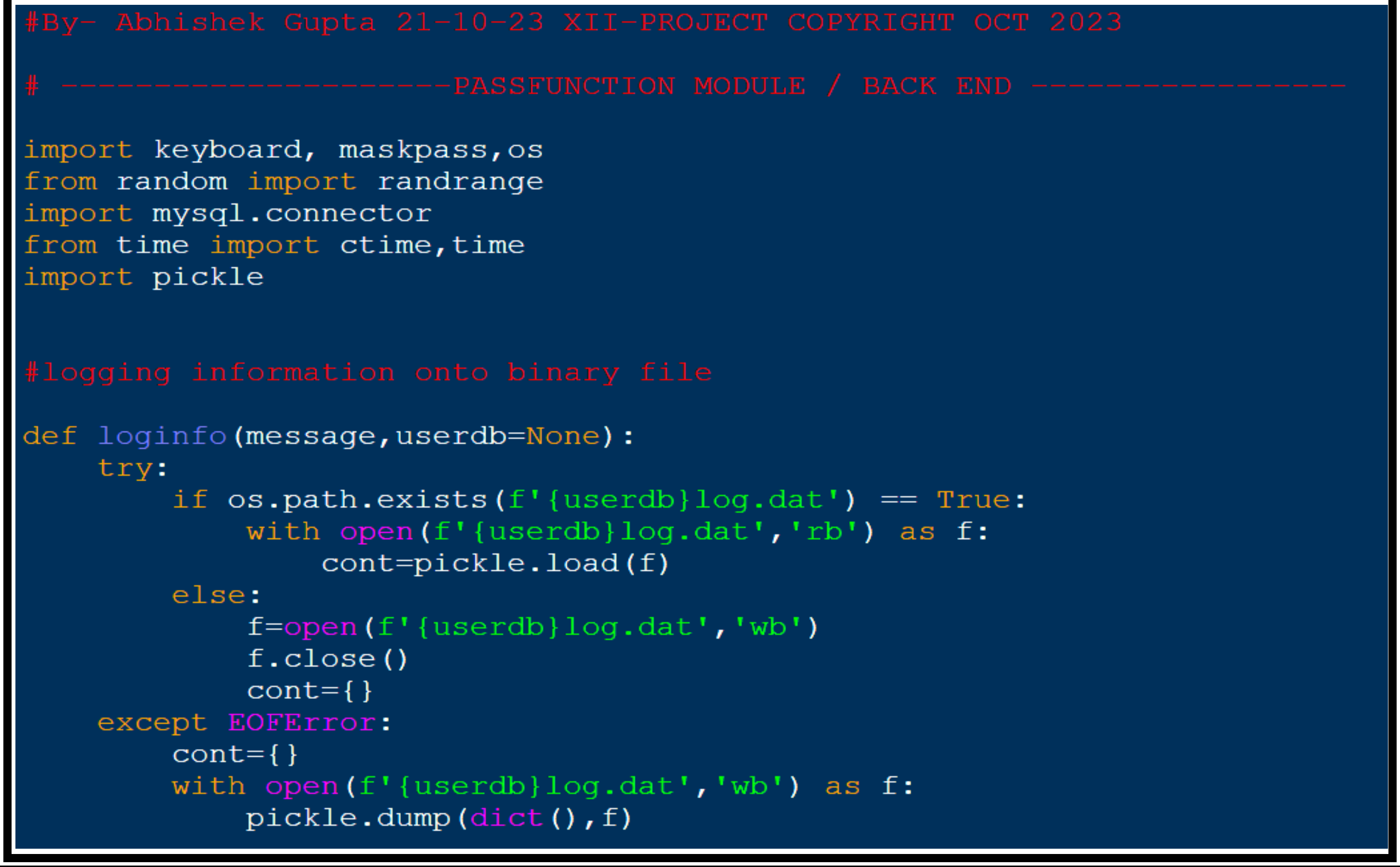


```python
# AUTENTICATE - LOGIN
def login(userdb=None):
    while True:
        user_inp = str.lower(input('Username: '))
        mstr_pass_inp = maskpass.advpass('Master Password: ','*',True,True)
        if user_inp in passfunction.logindb(passfunction.readdb()) and \
           mstr_pass_inp == passfunction.logindb(passfunction.readdb())[user_inp]:
            global user
            user = user_inp
            print('Verified')
            print(f'{user} logged in.')
            print('loading...')
            passfunction.loginfo(f'{user} logged in',user)
            time.sleep(2)
            passfunction.try_again_inp(main_menu,'Invalid Input. PLease try again.'\
                                       ,'Enter your choice(1/2/3/4/5/6/7/8): ',\
                                       8,'main_ch',[1,2,3,4,5,6,7,8],\
                                       [passfunction.pass_param, passfunction.show_keys,\
                                        passfunction.search_rec,passfunction.add_rec, passfunction.edit_rec,\
                                        passfunction.delete_rec,passfunction.logread,welcome_input],user)
            break
        else:
            print('Entered password or username is wrong. Please try again.')
        print('----------------------------------------------------\n')
```



```
#AUTHENTICATE - SIGN UP
def signup(userdb=None):
    uname = input('Enter your username: ')
    if len((uname)) != 0 and uname not in passfunction.logindb(passfunction.readdb()):

        print('--------------------IMPORTANT !!---------------------------------------')
        print('Please enter below a Master Password (4 to 10 digits long).',
                'Master Password is the key to all your stored passwords.',
                'Ensure you remember it. Once forgotten, your passwords cannot be recovered.',sep='\n')
        print('---------------------------------------------------------------------------')
        psd = maskpass.advpass('Enter your Master Password: ','*',True)
        psd_cfm = maskpass.advpass('Confirm your Master Password: ','*', True)
        if psd == psd_cfm :
            passfunction.signupdb(uname,psd)
            print('Account successfully created. Thank you')
            global user
            user = uname
            passfunction.loginfo(f'{user} signed up',user)
            print('verifed')
            time.sleep(1)
            print(f'{user} logged in.')
            print('loading')
            time.sleep(2)
            passfunction.try_again_inp(main_menu,'Invalid Input. PLease try again.'\
                                       ,'Enter your choice(1/2/3/4/5/6/7/8): ',\
                                       8,'main_ch',[1,2,3,4,5,6,7,8],\
                                       [passfunction.pass_param, passfunction.show_keys,\
                                        passfunction.search_rec,passfunction.add_rec, passfunction.edit_rec,\
                                        passfunction.delete_rec,passfunction.logread,welcome_input],user)
    else:
        print('\nthis username already exists\n')

# RUNNING WELCOME SCREEN AFTER BEING LOGGED OUT
def welcome_input(userdb=None):
    passfunction.try_again_inp(welcome_screen,'Invalid Input. Please try again.',\
                               'Enter your choice (1/2/3) : ',3,'wel_ch',[1,2,3],[login,signup],user)

# RUNNING WELCOME SCREEN - 1ST TIME IN EXECUTION OF PROGRAM
welcome_input()
```

2. passfunction.py

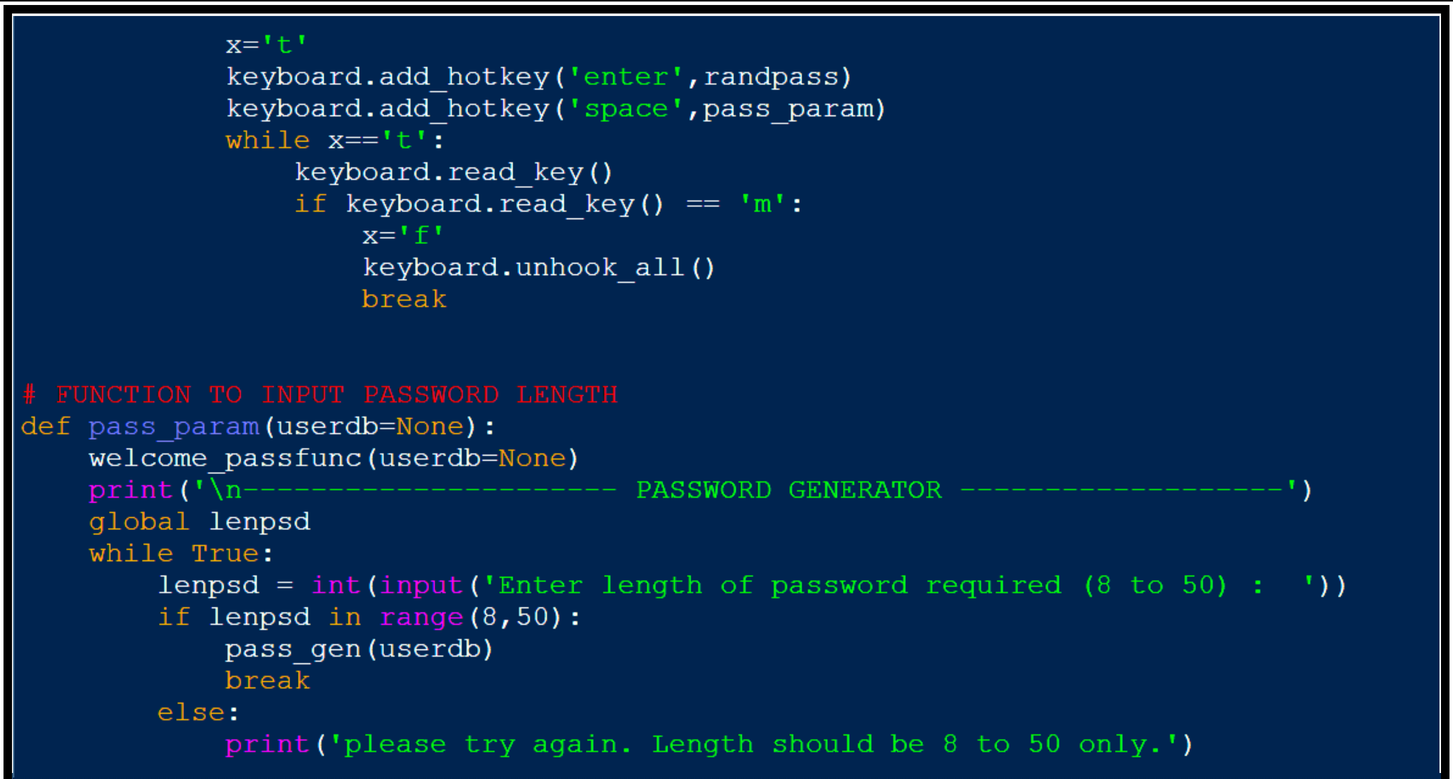


```
#By- Abhishek Gupta 21-10-23 XII-PROJECT COPYRIGHT OCT 2023

# ---------------------PASSFUNCTION MODULE / BACK END -----------------

import keyboard, maskpass,os
from random import randrange
import mysql.connector
from time import ctime,time
import pickle


#logging information onto binary file

def loginfo(message,userdb=None):
    try:
        if os.path.exists(f'{userdb}log.dat') == True:
            with open(f'{userdb}log.dat','rb') as f:
                cont=pickle.load(f)
        else:
            f=open(f'{userdb}log.dat','wb')
            f.close()
            cont={}
    except EOFError:
        cont={}
        with open(f'{userdb}log.dat','wb') as f:
            pickle.dump(dict(),f)
    f=open(f'{userdb}log.dat','wb')
    updated={}
    if cont == {} and type(cont) == 'NoneType':
        updated = {message : f' ( {ctime(time())} )'}
    else:
        updated = cont | {message : f'{ctime(time())}'}
    pickle.dump(updated,f)
    f.close()
```

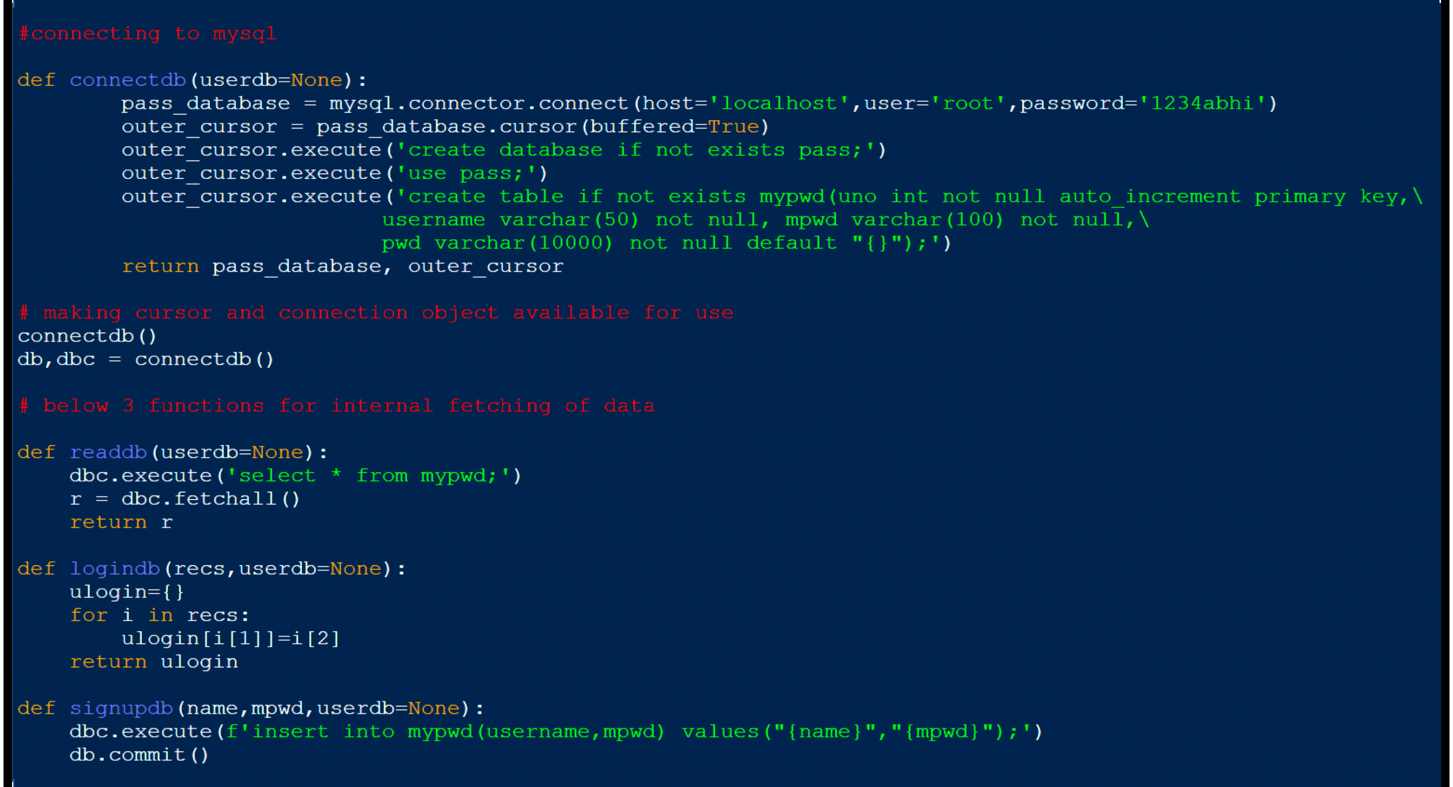
```
#reading from binary file

def logread(userdb=None):

    f=open(f'{userdb}log.dat','rb')
    r=pickle.load(f)
    sort_r = dict(sorted(list(r.items()), key= lambda x:x[1]))

    print('\n-------------------- SECURITY LOG --------------------------')
    for rec in sort_r:
        print(rec,'\t\t',sort_r[rec])
    print('--------------------------------------------------------------')
    f.close()



# EXECUTED WITH PASS_GEN - PASSWORD GENERATOR
def welcome_passfunc(userdb=None):
    print('--------------------IMPORTANT!--------------------------')
    print('Press \'ENTER\' key to regenerate password')
    print('Press \'m\' key to return to main menu')
```

```
def try_again_inp(func,wrong_msg,inp_msg,no_of_ifs,ifs_var,ifs_values,func_exe,username):
    while True:
        func()
        try:
            ifs_var = int(input(inp_msg).strip())
        except ValueError:
            print('Should enter positive values only')

        if no_of_ifs == 3 and ifs_var == no_of_ifs:
            print('\n---------------------- Thankyou for using our services ----------------------------')
            break
        if ifs_var == no_of_ifs:
            print('Logged Out.')
            loginfo(f'{username} Exited.',userdb = username)
            break
        else:
            for i in range(no_of_ifs):
                if ifs_var == ifs_values[i]:
                    func_exe[i](userdb=username)
                    break
            else:
                print(wrong_msg)
```

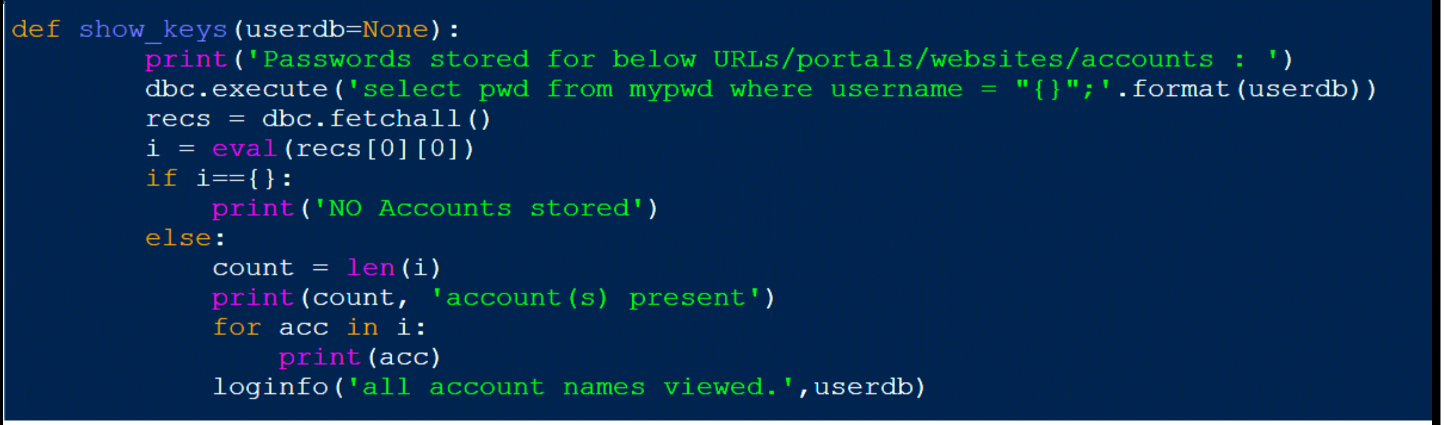
```
#PASSWORD GENERATOR
def pass_gen(userdb=None):
            def randpass(userdb=None):
                gen_psd_orig = str()
                for i in range(lenpsd):
                    ch = chr(randrange(33,127))
                    gen_psd_orig += ch
                print('\n',gen_psd_orig,'-----------------------------',sep='\n')
                loginfo('password generated.',userdb)

            randpass()
```



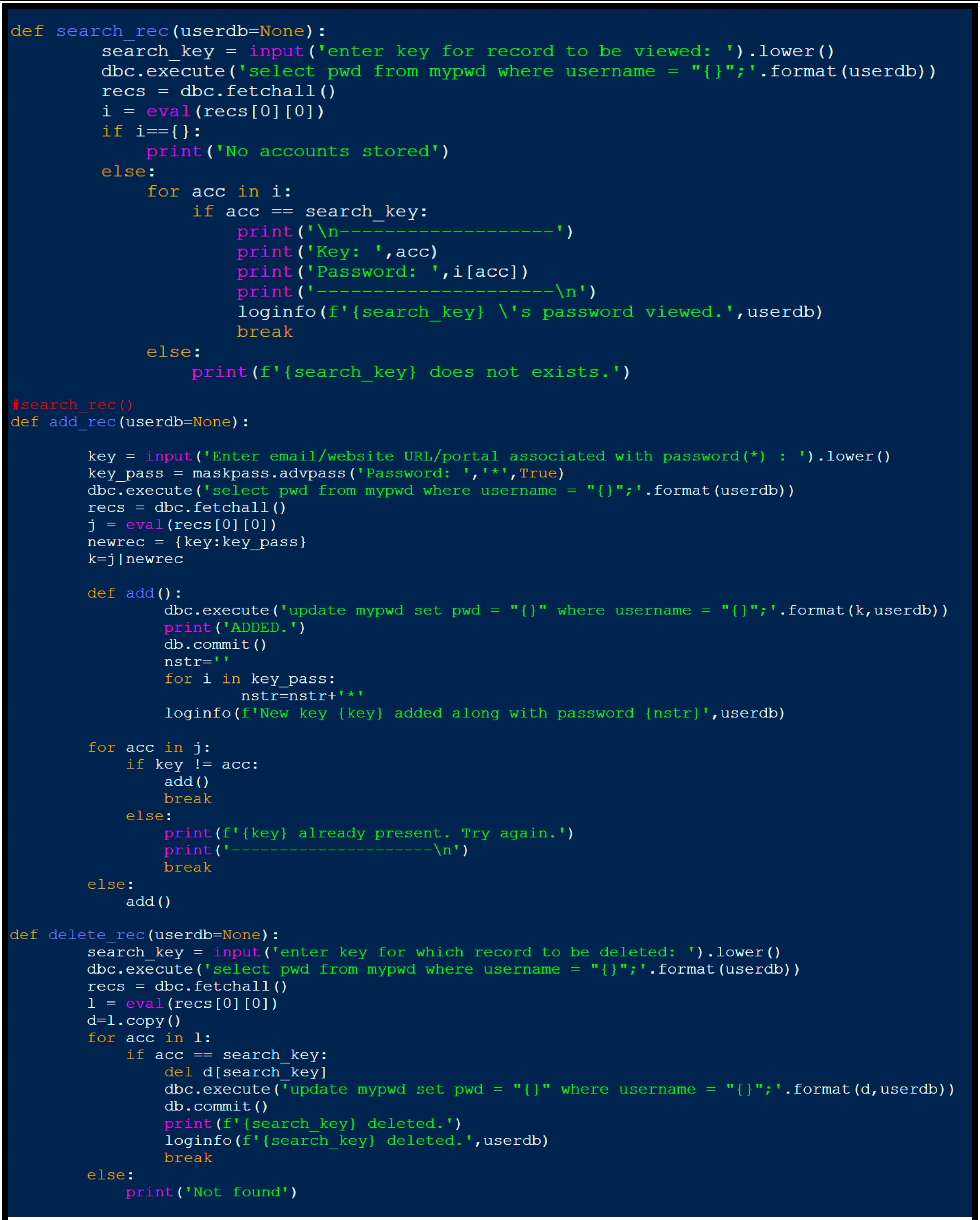
```
            x='t'
            keyboard.add_hotkey('enter',randpass)
            keyboard.add_hotkey('space',pass_param)
            while x=='t':
                keyboard.read_key()
                if keyboard.read_key() == 'm':
                    x='f'
                    keyboard.unhook_all()
                    break


# FUNCTION TO INPUT PASSWORD LENGTH
def pass_param(userdb=None):
    welcome_passfunc(userdb=None)
    print('\n---------------------- PASSWORD GENERATOR -------------------')
    global lenpsd
    while True:
        lenpsd = int(input('Enter length of password required (8 to 50) :  '))
        if lenpsd in range(8,50):
            pass_gen(userdb)
            break
        else:
            print('please try again. Length should be 8 to 50 only.')
```

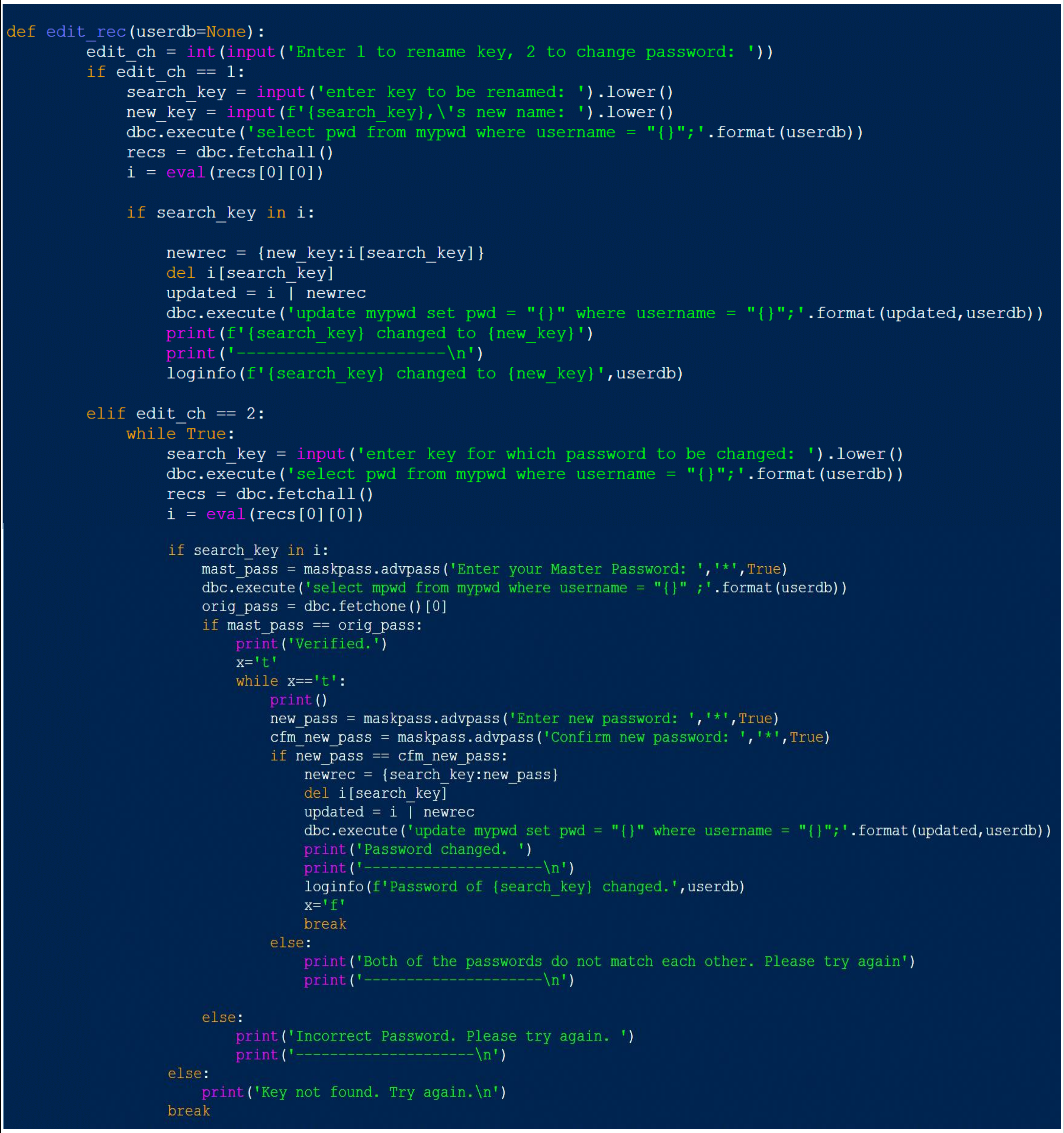


```
#connecting to mysql

def connectdb(userdb=None):
        pass_database = mysql.connector.connect(host='localhost',user='root',password='1234abhi')
        outer_cursor = pass_database.cursor(buffered=True)
        outer_cursor.execute('create database if not exists pass;')
        outer_cursor.execute('use pass;')
        outer_cursor.execute('create table if not exists mypwd(uno int not null auto_increment primary key,\
                            username varchar(50) not null, mpwd varchar(100) not null,\
                            pwd varchar(10000) not null default "{}");')
        return pass_database, outer_cursor

# making cursor and connection object available for use
connectdb()
db,dbc = connectdb()

# below 3 functions for internal fetching of data

def readdb(userdb=None):
    dbc.execute('select * from mypwd;')
    r = dbc.fetchall()
    return r

def logindb(recs,userdb=None):
    ulogin={}
    for i in recs:
        ulogin[i[1]]=i[2]
    return ulogin

def signupdb(name,mpwd,userdb=None):
    dbc.execute(f'insert into mypwd(username,mpwd) values("{name}","{mpwd}");')
    db.commit()
```



```
def show_keys(userdb=None):
        print('Passwords stored for below URLs/portals/websites/accounts : ')
        dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
        recs = dbc.fetchall()
        i = eval(recs[0][0])
        if i=={}:
            print('NO Accounts stored')
        else:
            count = len(i)
            print(count, 'account(s) present')
            for acc in i:
                print(acc)
            loginfo('all account names viewed.',userdb)
```

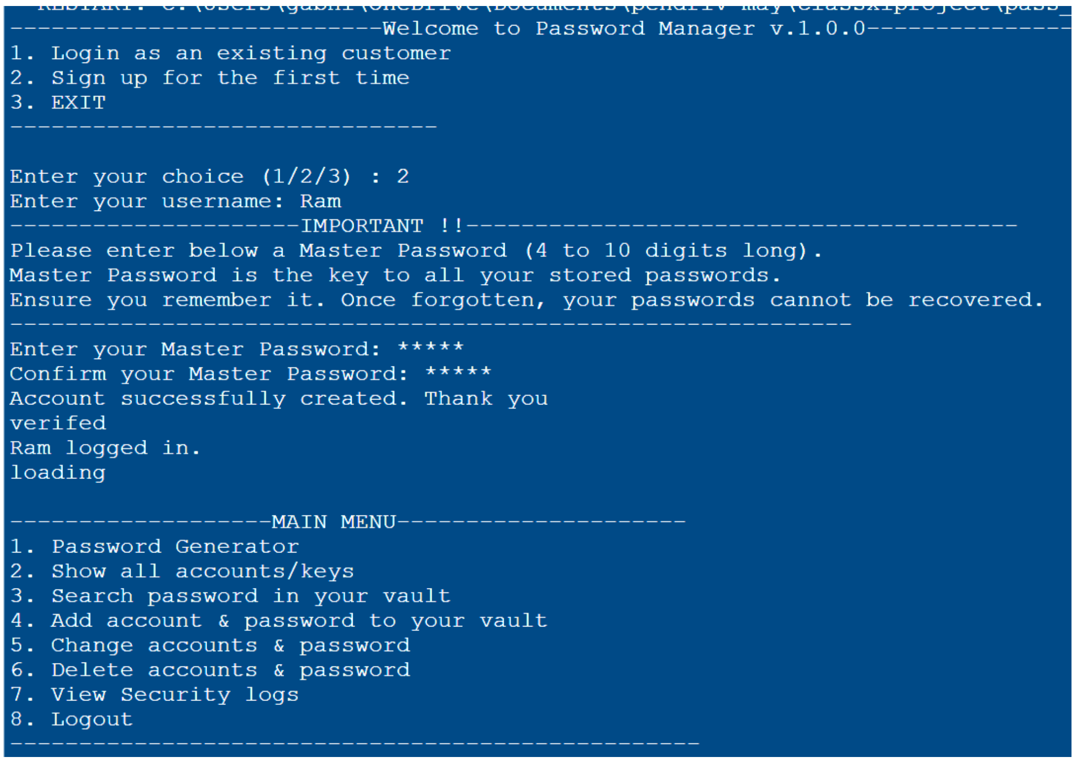

```
def search_rec(userdb=None):
        search_key = input('enter key for record to be viewed: ').lower()
        dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
        recs = dbc.fetchall()
        i = eval(recs[0][0])
        if i=={}:
            print('No accounts stored')
        else:
            for acc in i:
                if acc == search_key:
                    print('\n--------------------')
                    print('Key: ',acc)
                    print('Password: ',i[acc])
                    print('---------------------\n')
                    loginfo(f'{search_key} \'s password viewed.',userdb)
                    break
            else:
                print(f'{search_key} does not exists.')

#search_rec()
def add_rec(userdb=None):

        key = input('Enter email/website URL/portal associated with password(*) : ').lower()
        key_pass = maskpass.advpass('Password: ','*',True)
        dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
        recs = dbc.fetchall()
        j = eval(recs[0][0])
        newrec = {key:key_pass}
        k=j|newrec

        def add():
                dbc.execute('update mypwd set pwd = "{}" where username = "{}";'.format(k,userdb))
                print('ADDED.')
                db.commit()
                nstr=''
                for i in key_pass:
                        nstr=nstr+'*'
                loginfo(f'New key {key} added along with password {nstr}',userdb)

        for acc in j:
            if key != acc:
                add()
                break
            else:
                print(f'{key} already present. Try again.')
                print('---------------------\n')
                break
        else:
            add()

def delete_rec(userdb=None):
        search_key = input('enter key for which record to be deleted: ').lower()
        dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
        recs = dbc.fetchall()
        l = eval(recs[0][0])
        d=l.copy()
        for acc in l:
            if acc == search_key:
                del d[search_key]
                dbc.execute('update mypwd set pwd = "{}" where username = "{}";'.format(d,userdb))
                db.commit()
                print(f'{search_key} deleted.')
                loginfo(f'{search_key} deleted.',userdb)
                break
        else:
            print('Not found')
```



```python
def edit_rec(userdb=None):
        edit_ch = int(input('Enter 1 to rename key, 2 to change password: '))
        if edit_ch == 1:
            search_key = input('enter key to be renamed: ').lower()
            new_key = input(f'{search_key},\'s new name: ').lower()
            dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
            recs = dbc.fetchall()
            i = eval(recs[0][0])

            if search_key in i:

                newrec = {new_key:i[search_key]}
                del i[search_key]
                updated = i | newrec
                dbc.execute('update mypwd set pwd = "{}" where username = "{}";'.format(updated,userdb))
                print(f'{search_key} changed to {new_key}')
                print('---------------------\n')
                loginfo(f'{search_key} changed to {new_key}',userdb)

        elif edit_ch == 2:
            while True:
                search_key = input('enter key for which password to be changed: ').lower()
                dbc.execute('select pwd from mypwd where username = "{}";'.format(userdb))
                recs = dbc.fetchall()
                i = eval(recs[0][0])

                if search_key in i:
                    mast_pass = maskpass.advpass('Enter your Master Password: ','*',True)
                    dbc.execute('select mpwd from mypwd where username = "{}" ;'.format(userdb))
                    orig_pass = dbc.fetchone()[0]
                    if mast_pass == orig_pass:
                        print('Verified.')
                        x='t'
                        while x=='t':
                            print()
                            new_pass = maskpass.advpass('Enter new password: ','*',True)
                            cfm_new_pass = maskpass.advpass('Confirm new password: ','*',True)
                            if new_pass == cfm_new_pass:
                                newrec = {search_key:new_pass}
                                del i[search_key]
                                updated = i | newrec
                                dbc.execute('update mypwd set pwd = "{}" where username = "{}";'.format(updated,userdb))
                                print('Password changed. ')
                                print('---------------------\n')
                                loginfo(f'Password of {search_key} changed.',userdb)
                                x='f'
                                break
                            else:
                                print('Both of the passwords do not match each other. Please try again')
                                print('---------------------\n')

                    else:
                        print('Incorrect Password. Please try again. ')
                        print('---------------------\n')
                else:
                    print('Key not found. Try again.\n')
                break
```

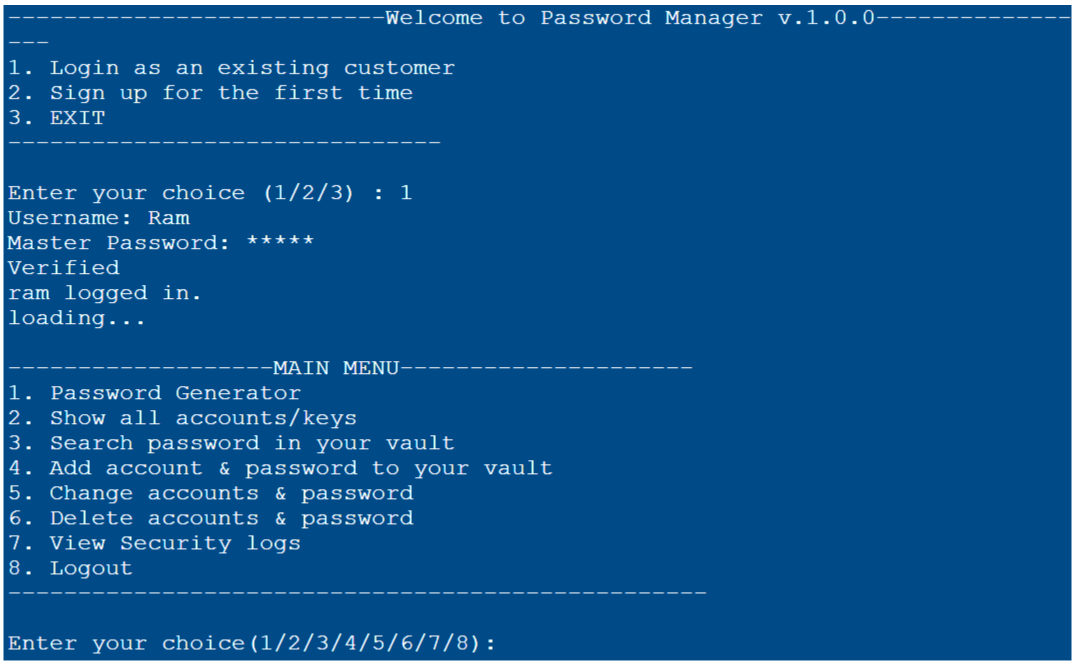
OUTPUT SCREENS

- **Sign up**



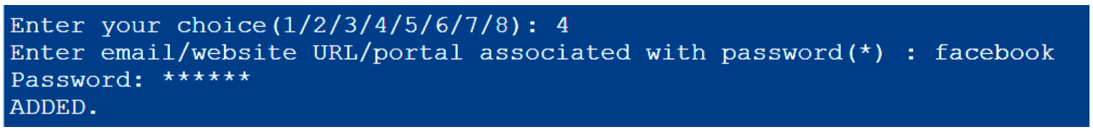
```
 RESTART: C:\Users\gauri\OneDrive\Documents\pendriv may\classxiproject\pass_
----------------------------Welcome to Password Manager v.1.0.0---------------
1. Login as an existing customer
2. Sign up for the first time
3. EXIT
--------------------------------

Enter your choice (1/2/3) : 2
Enter your username: Ram
---------------------IMPORTANT !!---------------------------------------
Please enter below a Master Password (4 to 10 digits long).
Master Password is the key to all your stored passwords.
Ensure you remember it. Once forgotten, your passwords cannot be recovered.
---------------------------------------------------------------
Enter your Master Password: *****
Confirm your Master Password: *****
Account successfully created. Thank you
verifed
Ram logged in.
loading

-------------------MAIN MENU---------------------
1. Password Generator
2. Show all accounts/keys
3. Search password in your vault
4. Add account & password to your vault
5. Change accounts & password
6. Delete accounts & password
7. View Security logs
8. Logout
---------------------------------------------------
```

- **Log in**



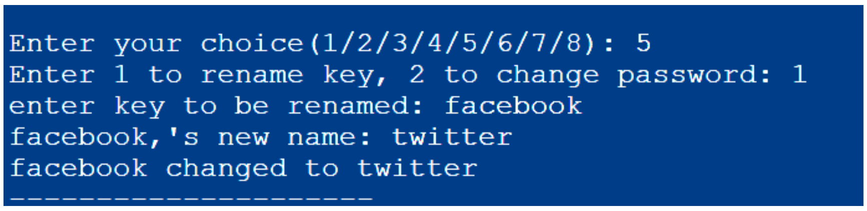
```
----------------------------Welcome to Password Manager v.1.0.0--------------
---
1. Login as an existing customer
2. Sign up for the first time
3. EXIT
--------------------------------

Enter your choice (1/2/3) : 1
Username: Ram
Master Password: *****
Verified
ram logged in.
loading...

-------------------MAIN MENU---------------------
1. Password Generator
2. Show all accounts/keys
3. Search password in your vault
4. Add account & password to your vault
5. Change accounts & password
6. Delete accounts & password
7. View Security logs
8. Logout
---------------------------------------------------

Enter your choice(1/2/3/4/5/6/7/8):
```

- **Adding account and password**



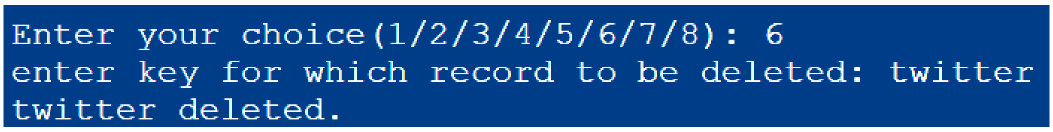
```
Enter your choice(1/2/3/4/5/6/7/8): 4
Enter email/website URL/portal associated with password(*) : facebook
Password: ******
ADDED.
```

- **Changing account name**



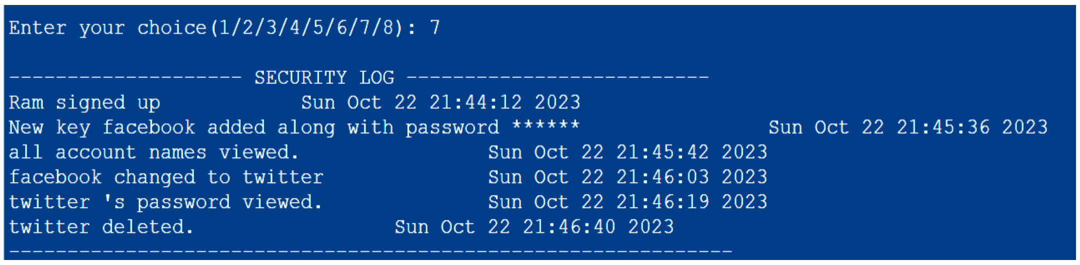
```
Enter your choice(1/2/3/4/5/6/7/8): 5
Enter 1 to rename key, 2 to change password: 1
enter key to be renamed: facebook
facebook,'s new name: twitter
facebook changed to twitter
---------------------
```

- **Deleting account and password**



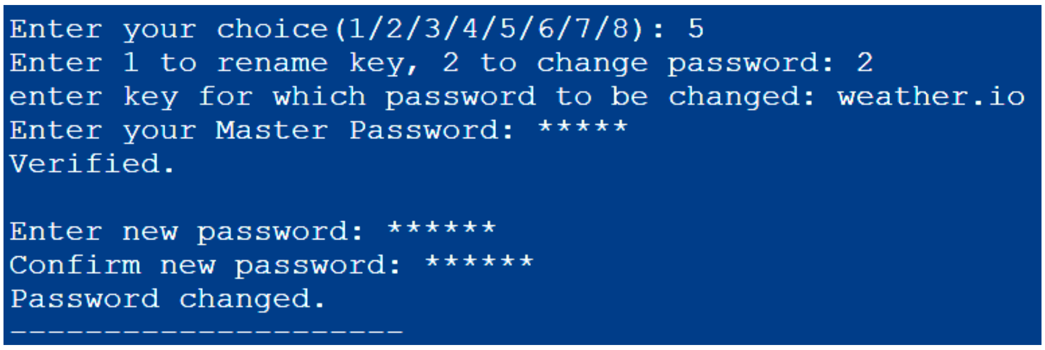
```
Enter your choice(1/2/3/4/5/6/7/8): 6
enter key for which record to be deleted: twitter
twitter deleted.
```

- **Viewing security logs**



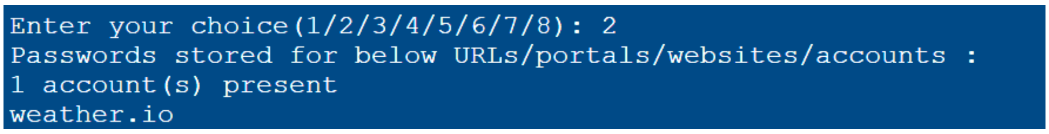
```
Enter your choice(1/2/3/4/5/6/7/8): 7

------------------ SECURITY LOG ------------------------
Ram signed up           Sun Oct 22 21:44:12 2023
New key facebook added along with password ******               Sun Oct 22 21:45:36 2023
all account names viewed.               Sun Oct 22 21:45:42 2023
facebook changed to twitter             Sun Oct 22 21:46:03 2023
twitter 's password viewed.             Sun Oct 22 21:46:19 2023
twitter deleted.                Sun Oct 22 21:46:40 2023
--------------------------------------------------------
```

- **Changing password of a particular account**



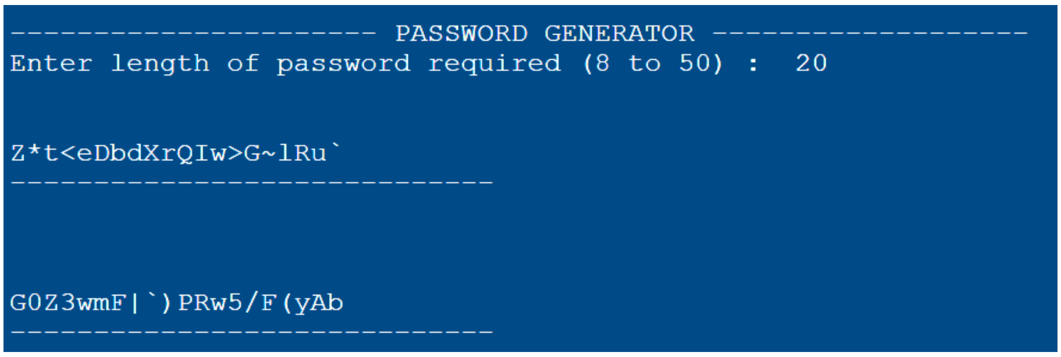
```
Enter your choice(1/2/3/4/5/6/7/8): 5
Enter 1 to rename key, 2 to change password: 2
enter key for which password to be changed: weather.io
Enter your Master Password: *****
Verified.

Enter new password: ******
Confirm new password: ******
Password changed.
---------------------
```

- **Viewing all accounts stored in vault**



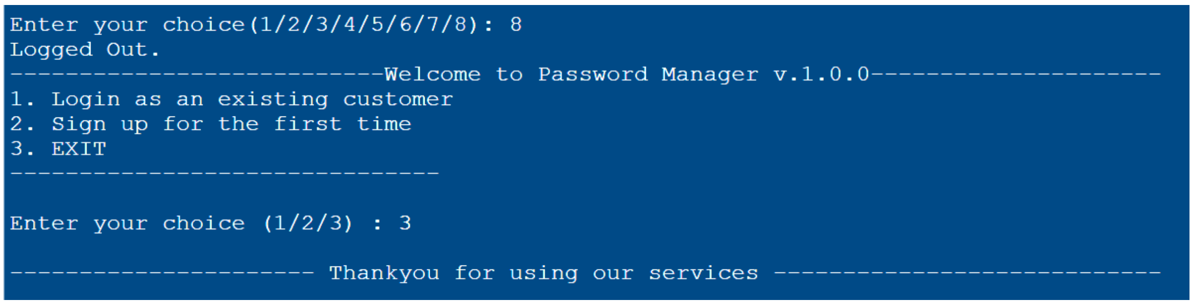
```
Enter your choice(1/2/3/4/5/6/7/8): 2
Passwords stored for below URLs/portals/websites/accounts :
1 account(s) present
weather.io
```

- **Generating Password**



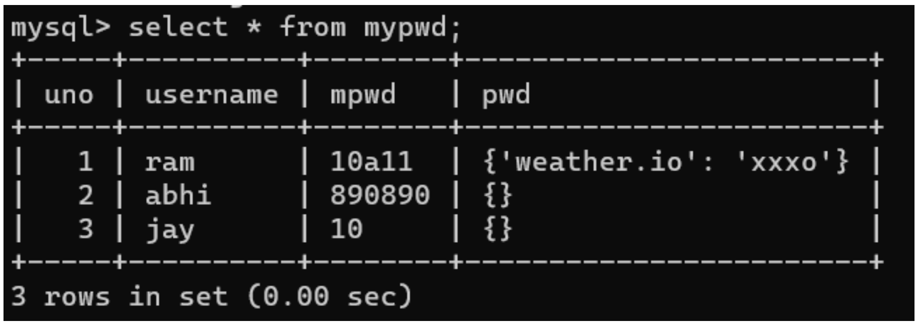
```
---------------------- PASSWORD GENERATOR --------------------
Enter length of password required (8 to 50) :  20


Z*t<eDbdXrQIw>G~lRu`
-----------------------------



G0Z3wmF|`)PRw5/F(yAb
-----------------------------
```

- **Exiting**



```
Enter your choice(1/2/3/4/5/6/7/8): 8
Logged Out.
---------------------------Welcome to Password Manager v.1.0.0---------------------
1. Login as an existing customer
2. Sign up for the first time
3. EXIT
-------------------------------

Enter your choice (1/2/3) : 3

---------------------- Thankyou for using our services ----------------------------
```

- **Changes made in mypwd table:**



```
mysql> select * from mypwd;
+-----+----------+---------+------------------------+
| uno | username | mpwd    | pwd                    |
+-----+----------+---------+------------------------+
|   1 | ram      | 10a11   | {'weather.io': 'xxxo'} |
|   2 | abhi     | 890890  | {}                     |
|   3 | jay      | 10      | {}                     |
+-----+----------+---------+------------------------+
3 rows in set (0.00 sec)
```

BIBLIOGRAPHY

Following resource/articles/online content have been used to prepare the project:-

1. Class XII Computer Science with Python by Preeti Arora
2. Stackoverflow.com
3. geeksforgeeks.org
4. https://www.techopedia.com/definition/31435/password-manager
5. https://its.ucsc.edu/security/passwords.html
6. Google Search results