VIMALA CENTRAL SCHOOL

KARAMCODU P.O, CHATHANNOOR, KOLLAM, KERALA-691579

(AFFILIATED TO CENTRAL BOARD OF SECONDARY EDUCATION NEW DELHI CODE.NO. 930331)

In partial fulfilment of the requirement for the second year of all Indian Senior Secondary Certificate Examination, 2023-24

DEPARTMENT OF COMPUTER SCIENCE

"PASSWORD MANAGER"

DONE BY :

Name	: AAROMAL A
Class	: XII-B
Register no	:
Admission no.	: 797
Year of study	: 2023-24

Teacher in charge : Mrs Bincy BS

Examiners : **Principal** :

ACKNOWLEDGEMENT

First of all, I would like to express my sincere concern and regards to our most respected **Rev.Fr Samuel Pazhavoor Padickal** and principal **Tom Mathew** for giving me ample support and encouragement for completing this project. This project was done under the guidance of **Bincy** teacher, department of computer science Vimala Central School.

I have great indebtedness to them and I express my wholehearted gratitude for their valuable guidance. I would also extend my faithful thanks to my friends and my family for their valuable help and advice during the course of the project.

At last, not the least I am thanking his guidance,

"THE ALMIGHTY GOD"

AAROMAL A

ABSTRACT

The **Python Password Manager** project presents a user friendly and efficient solution for securely managing passwords and sensitive information.The project developed with python programming language and sqlite database.

The program intended to store all your online credentials securely in one place under a strong master password i.e the only password you need to remember. It provides a simple console interface to manage/retrieve your credentials.

The functionality of this project is

- Secure storage of credentials
- Strong password generation
- Updation and deletion of stored credentials
- Copying required credentials to clipboard
- Exporting data to a backup file

The program demonstrates python-sql connectivity, database & file management, user authentication and programming methodologies.

CONTENTS

INTRODUCTION

In the modern digital age, password managers have become indispensable tools by promoting good password hygiene. With the proliferation of online accounts and cyber attacks,password managers offer a secure repository for storing and managing complex,unique passwords for various services in oneplace. They can generate and store strong passwords. Also,they reduce the burden of memorizing multiple strong credentials, insead only one strong master password is all you need.

This python password manager provides a safe vault to store all your online credentials securely in one place under a strong master password. It provides a simple console interface to manage/retrieve your credentials. The project offers features such as Secure storage of credentials, Strong password generation, updation and deletion of stored credentials, copying required credentials to clipboard, Exporting data to a backup file

In order to run the project, you must have Python and sqlite3 database installed on your PC. And the required external python modules also needed to be installed from pip for the clean execution of the program.

Modules

The project is developed on **python 3.11**, lightweight sqlite3 database, and some external python modules. The external python modules used for the development of this project are as follows :

1. ***Sqlite3*** : It is used to establish connection between python and the database
2. ***Rich 13.6.0*** : It is used to customize the console terminal interface
3. ***Pyperclip 1.8.2*** : It is used to copy passwords to clipboard.

Also some built-in modules like random and String are used for enhancing the functionality of the program.

We can download the following modules using pip and import it to the program using the import statement.

SURVEY OF TECHNOLOGIES

The following technologies are used in the project for the development of password manager.

1. ***PYTHON*** : Python is an interpreted, high-level, general purpose programming language developed by Guido van Rossum in 1991. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms. It has a comprehensive standard library and a huge community.

2. ***SQLITE3 :*** SQLite is a database engine written in the C programming language. It is not a standalone app; rather, it is a library that software developers embed in their apps. As such, it belongs to the family of embedded databases. It is the most widely deployed database engine, as it is used by several of the top web browsers, operating systems, mobile phones, and other embedded systems.Many programming languages have bindings to the SQLite library like python have sqlite3 library.

Some features of python :

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable.

1. *Simple and Expressive*: Python has few keywords, simple structure, and a clearly defined syntax. This allows us to maintain clear and compact code.

2. *Comprehensive standard library and huge community:* Python has a comprehensive standard library which has all the features out of the box. Its huge community has contributed a bundle of advanced functionalities.Thus it has a wide range of applications.

3. *Cross-platform language*: Python can run equally well on a variety of platforms like windows, linux/unix and smartphones which make it a portable language.

4. *Free and opensource*: By the support of a huge community, python is available free and opensource along with its source code

5. *Interpreted language*: This means the code is processed line by line by the interpreter at runtime. It makes python an easy to debug language.

6. *Object-Oriented language*: Python supports Object-Oriented style or technique of programming that encapsulates code within objects. It helps to manage heavy code bases.

SOFTWARE & HARDWARE REQUIREMENTS

HARDWARE SPECIFICATIONS :

- *Processor* : PENTIUM DUAL CORE or Above
- *RAM* : 2GB
- *Hard Disk* : 1TB
- *Monitor* : 14'' LED or Above
- *Keyboard* : 104 keys
- *Mouse* : Optical mouse
- *Printer* : 24 Pin DOT MATRIX or Above

SOFTWARE SPECIFICATIONS :

- *Operating System* : Windows 7 or Above, Linux/Unix
- *Front End* : Terminal, python3
- *Middleware* : python-sqlite connector (sqlite3)
- *Back End* : sqlite3 database

SOURCE CODE

#main.py

```
from rich.table import Table
from rich import print
from rich.panel import Panel
import pyperclip
from getpass import getpass

import utils #other important fns

panel = Panel("[bold blue]PASSWORD MANAGER[/bold blue]")
print(panel)

utils.initialisedb()

if utils.currentuser() == None:
    username = input("Enter username  : ")
    while True:
        masterpass = getpass("Enter masterpassword :")
        if masterpass ==  getpass("Re-Type :"):
            break
        else:
            print("[yellow]try again[/yellow]")

    authenticate = utils.registeruser(username, masterpass)
    print("[green]user registered successfully.[/green]")

else:
    masterpass = getpass("Enter masterpassword : ")
    authenticate = utils.authenticateuser(masterpass)
    print("[green]user logged in successfully.[/green]")

if authenticate:
    print("""\n [bold]OPTIONS[/bold]:
          1. ADD ENTRY
          2. SHOW/SEARCH ENTRIES
```

```
          3. UPDATE ENTRIES
          4. DELETE ENTRY
          5. RESTORE/EXPORT
          6. EXIT""")
else:
    print("[red]OOPS! something went wrong.[/red]")

while authenticate:
    dbcur = utils.dbwriter()
    op = input("> ")

    if op == "1":
        sitename = input("Enter sitename : ")
        if sitename != "":
            password = input("Enter password (default generate) : ")
            if password == "":
                password = utils.generatepass(10)
            dbcur.execute("INSERT INTO ENTRIES (sitename,password) VALUES
(?,?)",(sitename,password))
            print("[green]added new entry[/green]")
        else:
            print("[red]invalid site name[/red]")
            pass

    elif op == "2":
        table = Table(title="VAULT")
        table.add_column("No", justify="left", style="bold")
        table.add_column("site", justify="center")
        table.add_column("password", justify="right")
        dbcur.execute("SELECT no,sitename,password FROM ENTRIES")
        records =  list(dbcur.fetchall())
        for row in records:
            table.add_row(str(row[0]), row[1], row[2])
        print(table)
        try:
            no = int(input("COPY password (0/entryno):"))
            if utils.checkentry(no):
                for i in records:
                    if i[0] == no:
                        pyperclip.copy(i[2])
                        break
                print("[green]sucessfully copied to clipboard[/green]")
```

```
            elif no == 0:
                pass
            else:
                print("[red]no entry found[/red]")
        except:
            print("[red]something went wrong[/red]")
            pass


    elif op == "3":
        try:
            no = int(input("Enter the entry no. to update : "))
            if utils.checkentry(no):
                sitename =  input("Enter sitename (default old) : ")
                password = input("Enter password (default old): ")
                if sitename:
                    dbcur.execute("UPDATE ENTRIES SET sitename = ? WHERE no =
?",(sitename,no))
                if password:
                    dbcur.execute("UPDATE ENTRIES SET password = ? WHERE no =
?",(password,no))
                print("[green]updated entry[/green]")
            else:
                print("[red]no entry found[/red]")
        except:
            print("[red]something went wrong[/red]")
            pass

    elif op == "4":
        try:
            no = int(input("Enter entry no. to delete (no / 404 to delete all): "))
            if utils.checkentry(no):
                dbcur.execute("DELETE FROM ENTRIES WHERE no=?",(no,))
                print("[green]deleted entry[/green]")
            elif no == 404:
                dbcur.execute("DELETE FROM ENTRIES")
                print("[green]wiped all the entries[/green]")
            else:
                print("[red]no entry found[/red]")
        except:
            print("[red]something went wrong[/red]")
            pass
```

```
    elif op == "5":
        dbcur.execute("SELECT * FROM ENTRIES")
           with open("./entries.txt","w") as fp:
            lines = [str(i) + "\n" for i in dbcur.fetchall()]
            fp.writelines(lines)
        print("[green]exported to ./entries.txt[/green]")

    elif op == "6":
        authenticate = False

    else:
        pass

utils.closevault()
```

#utils.py

```
import sqlite3
import random,string


dbconn = sqlite3.connect("./vault.db")
dbcur = dbconn.cursor()

def dbwriter():
    return dbcur

def initialisedb():
    dbcur.execute("CREATE TABLE IF NOT EXISTS ENTRIES (no INTEGER PRIMARY KEY
AUTOINCREMENT, sitename TEXT NOT NULL, password TEXT)")
    dbcur.execute("CREATE TABLE IF NOT EXISTS USERS (user TEXT, masterpass TEXT)")

def currentuser():
    dbcur.execute("SELECT * FROM USERS")
    return dbcur.fetchone()

def registeruser(username, masterpass):
    dbcur.execute("INSERT INTO USERS(user,masterpass) VALUES
(?,?)",(username,masterpass))
    return True
```

```
def authenticateuser(masterpass):
    user = currentuser()
    if masterpass == user[1]:
        return True
    else:
        return False

def checkentry(entryno):
    dbcur.execute("SELECT no FROM ENTRIES")
    entrynos = [item[0] for item in list(dbcur.fetchall())]
    return entryno in entrynos

def generatepass(length):
    return ''.join(random.choices(string.ascii_uppercase + string.digits, k = length))

def closevault():
    dbconn.commit()
    dbconn.close()
```

SQL COMMANDS

1. *Creating Tables :*

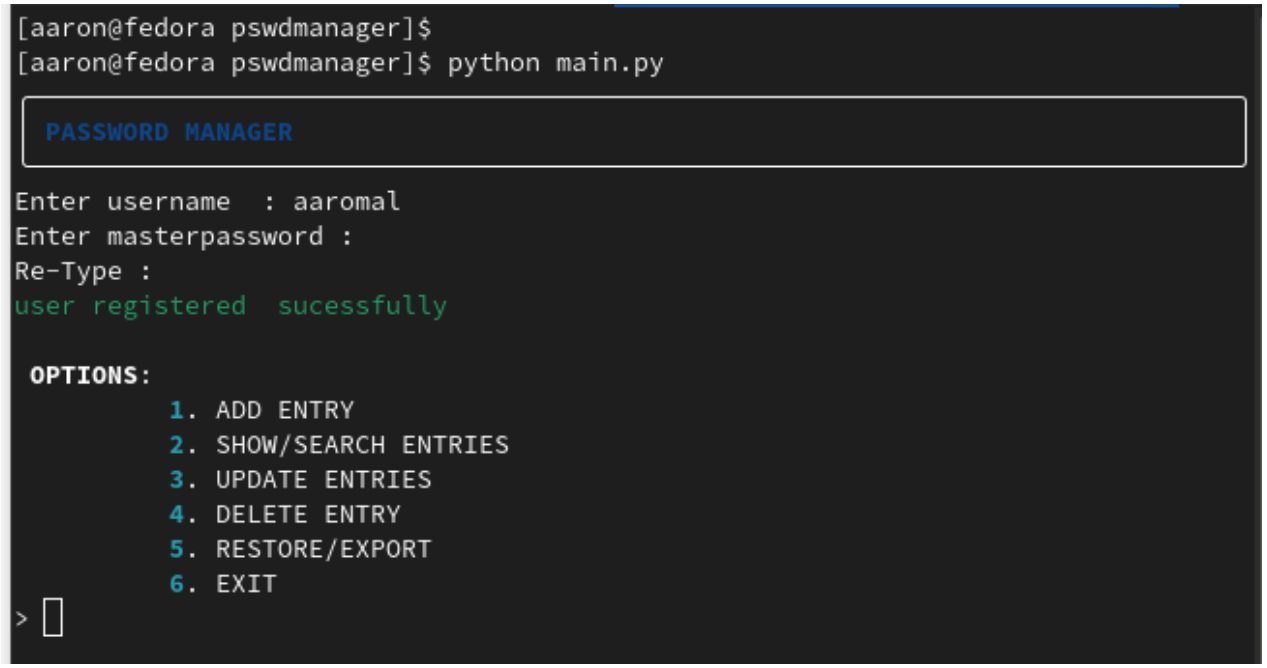
```
def initialisedb():
    dbcur.execute("CREATE TABLE IF NOT EXISTS ENTRIES (no INTEGER PRIMARY KEY
AUTOINCREMENT, sitename TEXT NOT NULL, password TEXT)")
    dbcur.execute("CREATE TABLE IF NOT EXISTS USERS (user TEXT, masterpass TEXT)")
```

2. *Describing tables :*

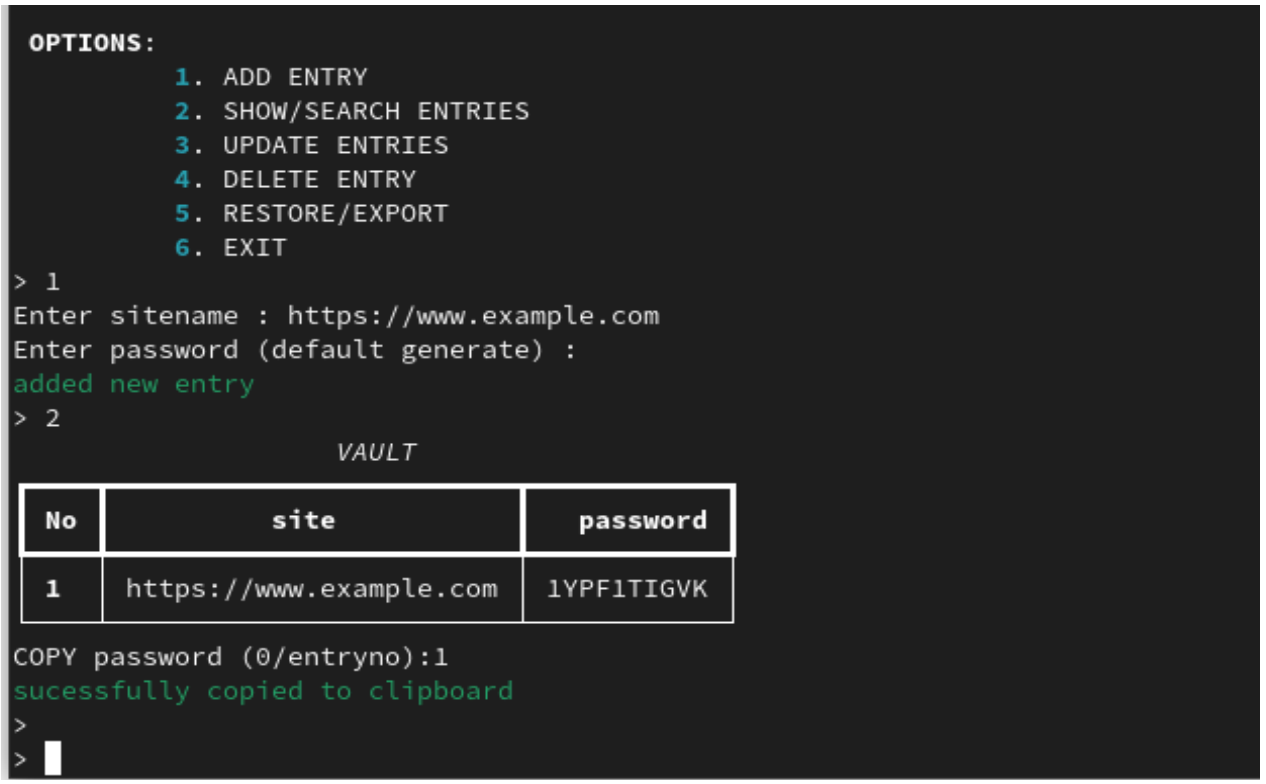
```
sql> describe ENTRIES;
+----------+------+------+-----+---------+----------------+
| Field    | Type | Null | Key | Default | Extra          |
+----------+------+------+-----+---------+----------------+
| no       | int  | NO   | PRI | NULL    | auto_increment |
| sitename | text | NO   |     | NULL    |                |
| password | text | YES  |     | NULL    |                |
+----------+------+------+-----+---------+----------------+
3 rows in set (0.19 sec)

sql> describe USERS;
+------------+------+------+-----+---------+-------+
| Field      | Type | Null | Key | Default | Extra |
+------------+------+------+-----+---------+-------+
| user       | text | YES  |     | NULL    |       |
| masterpass | text | YES  |     | NULL    |       |
+------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

OUTPUT SCREEN

1. *User register/login :*



2. *Adding & retrieving entry :*



3. *Updating entry :*

```
 OPTIONS:
          1. ADD ENTRY
          2. SHOW/SEARCH ENTRIES
          3. UPDATE ENTRIES
          4. DELETE ENTRY
          5. RESTORE/EXPORT
          6. EXIT
> 3
Enter the entry no. to update : 1
Enter sitename (default old) :
Enter password (default old): mynewpassword
updated entry
> 2
                    VAULT
```

No	site	password
1	https://www.example.com	mynewpassword

```
COPY password (0/entryno):0
>
```

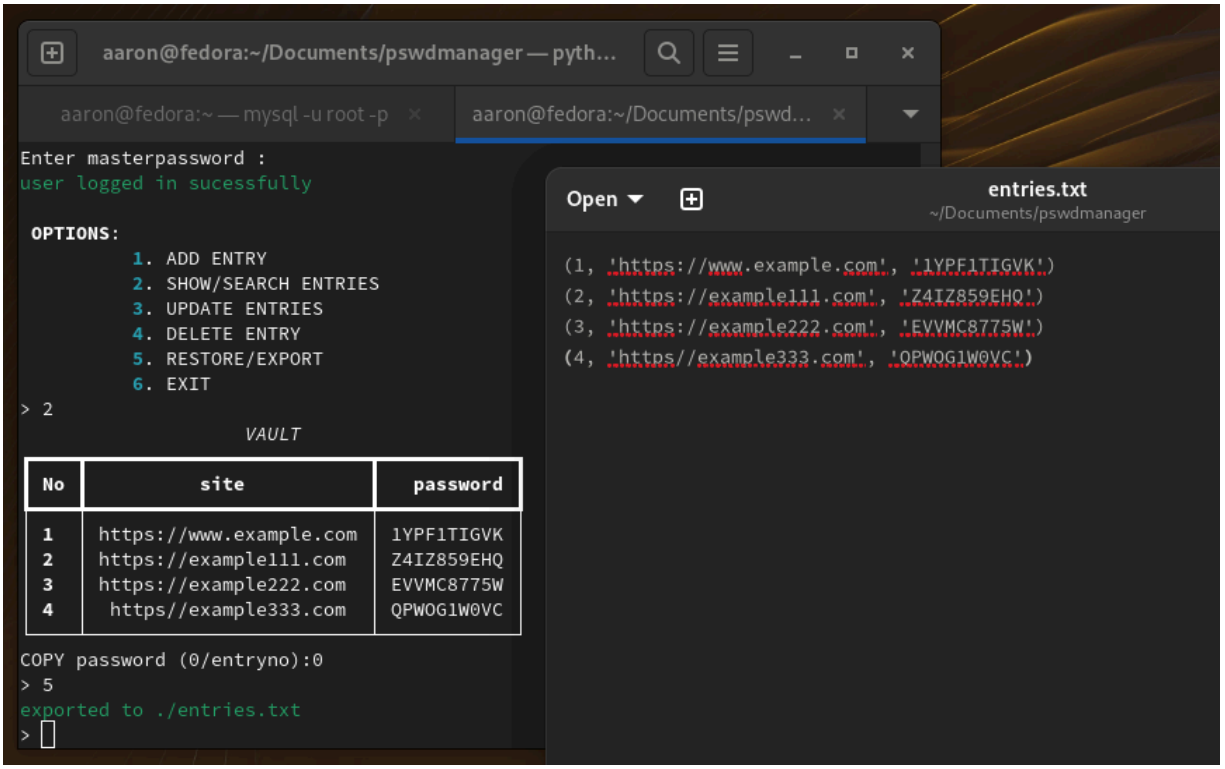
4. *Delete entries :*

```
 OPTIONS:
          1. ADD ENTRY
          2. SHOW/SEARCH ENTRIES
          3. UPDATE ENTRIES
          4. DELETE ENTRY
          5. RESTORE/EXPORT
          6. EXIT
> 4
Enter entry no. to delete (no / 404 to delete all): 1
deleted entry
> 2
          VAULT
```

No	site	password

```
COPY password (0/entryno):0
>
```

5. *Restore/Export entries into a file :*



6. *Effect on database :*

```
sql> SELECT * FROM USERS;
+---------+------------+
| user    | masterpass |
+---------+------------+
| aaromal | tiger      |
+---------+------------+
1 row in set (0.00 sec)

sql> SELECT * FROM ENTRIES;
+----+------------------------+------------+
| no | sitename               | password   |
+----+------------------------+------------+
|  1 | https://example.com    | 1YPF1TIGVK |
|  2 | https://example111.com | Z4IZ859EHQ |
|  3 | https://example222.com | EVVMC8775W |
|  4 | https://example333.com | QPWOG1W0VC |
+----+------------------------+------------+
4 rows in set (0.00 sec)
```

CONCLUSION

We are very delighted and relieved with the successful completion of our project. We have made a small attempt in developing a model of password manager. The program intended to demonstrate python-sql connectivity, database & file management, user authentication and programming methodologies. This project helped us immensely in understanding the basic principles behind the operations involved within the Python and SQL project environment. This has also enabled us to learn deeply in Python and PiPy Community.

We tested the application and have found it working satisfactorily on Windows/Linux platforms and no serious errors are spotted. Hence this software project has been developed to fulfill the academic learning in CBSE Class XII, Computer Science Curriculum.

BIBLIOGRAPHY

Various books and websites are being referred for the successful development of this project. Some of them are :

- *CBSE Class XII Preeti Arora CS Textbook (2023 edition)*
- *https://docs.python.org/3*
- *https://github.com/*
- *https://chat.openai.com/*
- *https://www.wikipedia.org/*