VIDHI ROHIRA
S.Y B.TECH

COMPUTER ENGINEERING

231071052

PD LAB 8

BATCH-C

LABORATORY 8

AIM:- Write an application to connect to a database in python.

THEORY:-

Q] WHAT IS TKINTER?

Tkinter is a standard GUI (Graphical User Interface) toolkit for Python, providing a simple way to create desktop applications. It is built on the Tcl/Tk framework and is included with most Python installations. With Tkinter, developers can create windows, buttons, menus, and other widgets to build interactive applications. Its ease of use and integration with Python make it popular for beginners and experienced programmers alike. Additionally, Tkinter supports various platform-specific features, ensuring that applications run smoothly on Windows, macOS, and Linux.

Q] WHAT IS A DATABASE?

A database is an organized collection of data that is stored and accessed electronically. It allows for efficient data management, retrieval, and manipulation. Databases can store various types of information, such as text, numbers, images, and more, in a structured format, typically using tables. They use a database management system (DBMS) to facilitate operations like querying, updating, and managing the data. Databases are essential for applications that require persistent data storage, such as websites, business applications, and enterprise systems, enabling users to easily retrieve and analyze information.

Q] HOW TO WRITE A PROGRAM TO CONNECT TO DATABASE?

To write a program that connects to a database, you typically start by importing a database-specific library or module, such as `sqlite3` for SQLite or `mysql.connector` for MySQL. Next, you'll establish a connection using a connection string that specifies the database location and credentials, if necessary. After establishing the connection, you can create a cursor object to execute SQL commands for querying, inserting, updating, or deleting data. It's important to handle exceptions to manage potential connection errors gracefully. Finally, ensure that you close the connection properly to release resources.

CODE:-

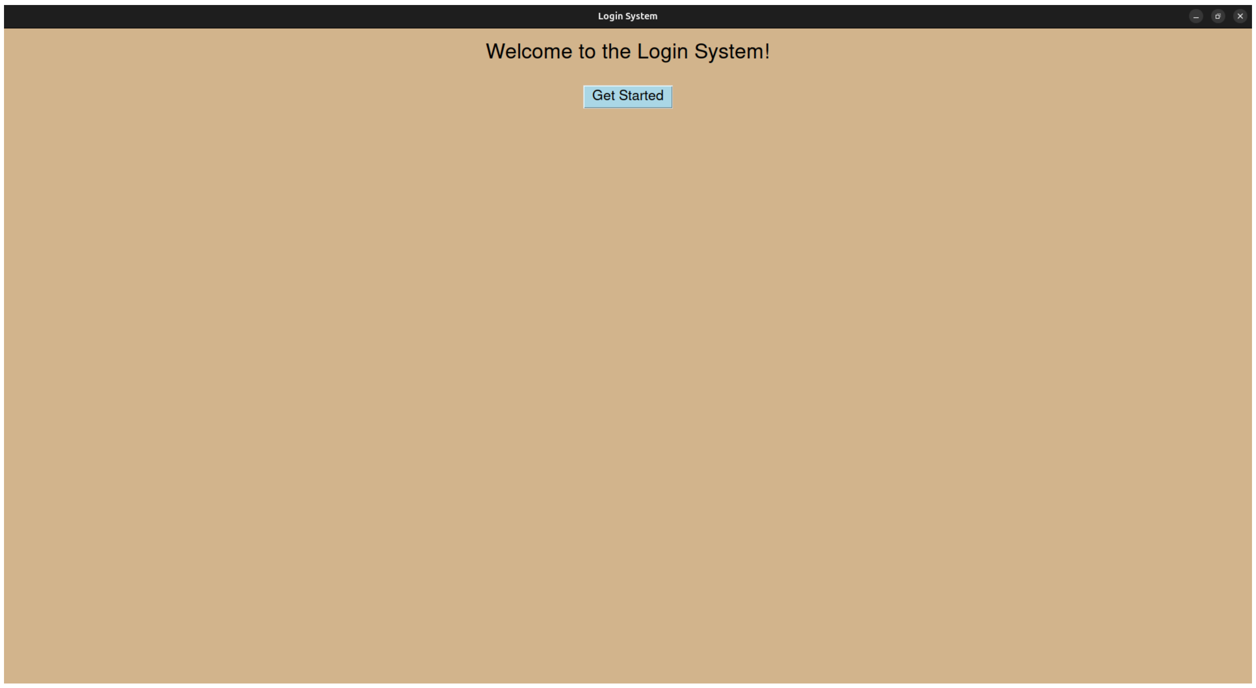
```
import tkinter as tk
from tkinter import messagebox
import sqlite3
import hashlib

# Create a database or connect to one
conn = sqlite3.connect('user_data.db')
c = conn.cursor()

# Create a table for storing user information
c.execute('''CREATE TABLE IF NOT EXISTS users (
               username TEXT PRIMARY KEY,
               password TEXT NOT NULL)''')
conn.commit()

# Variable to track password visibility
password_visible = False

# Function to show the welcome page
def show_welcome_page():
   for widget in root.winfo_children():
       widget.destroy()  # Clear the main window

   welcome_label = tk.Label(root, text="Welcome to the Login
System!", font=('Helvetica', 24), bg='#D2B48C')
   welcome_label.pack(pady=20)

   get_started_button = tk.Button(root, text="Get Started",
command=show_login, font=('Helvetica', 16), bg='lightblue',
activebackground='lightgreen')
   get_started_button.pack(pady=10)

# Function to display the login page
def show_login():
   for widget in root.winfo_children():
       widget.destroy()  # Clear the main window

   username_label = tk.Label(root, text="Username",
font=('Helvetica', 16), bg='#D2B48C')
   username_label.pack(pady=5)

   global username_entry
   username_entry = tk.Entry(root, font=('Helvetica', 16), width=30)
   username_entry.pack(pady=5)

   password_label = tk.Label(root, text="Password",
font=('Helvetica', 16), bg='#D2B48C')
   password_label.pack(pady=5)
```

```
    # Create a frame for password entry and eye button
    password_frame = tk.Frame(root, bg='#D2B48C')
    password_frame.pack(pady=5)

    global password_entry
    password_entry = tk.Entry(password_frame, show='*',
font=('Helvetica', 16), width=25)
    password_entry.pack(side=tk.LEFT)

    # Create the eye button to toggle password visibility
    global eye_button
    eye_button = tk.Button(password_frame, text='👁️',
command=toggle_password_visibility, font=('Helvetica', 12),
bg='white')
    eye_button.pack(side=tk.RIGHT)

    login_button = tk.Button(root, text="Login", command=login,
font=('Helvetica', 16), bg='lightblue',
activebackground='lightgreen')
    login_button.pack(pady=10)

    register_button = tk.Button(root, text="Register",
command=register, font=('Helvetica', 16), bg='lightblue',
activebackground='lightgreen')
    register_button.pack(pady=10)

    recovery_button = tk.Button(root, text="Forgot Password?",
command=show_recovery, font=('Helvetica', 12), bg='lightyellow',
activebackground='lightgreen')
    recovery_button.pack(pady=5)

# Function to toggle password visibility
def toggle_password_visibility():
    global password_visible
    if password_visible:
        password_entry.config(show='*')  # Hide password
        eye_button.config(text='👁️')  # Change icon to closed eye
    else:
        password_entry.config(show='')  # Show password
        eye_button.config(text='👁‍🗨')  # Change icon to open eye
    password_visible = not password_visible

# Function to register a new user
def register():
    username = username_entry.get()
    password = password_entry.get()

    if username == "" or password == "":
        messagebox.showerror("Error", "All fields are required!")
        return

    if len(username) < 3:
        messagebox.showerror("Error", "Username must be at least 3 
characters long!")
        return

    hashed_password = 
hashlib.sha256(password.encode('utf-8')).hexdigest()
    try:
        # Insert user into the database
        c.execute("INSERT INTO users (username, password) VALUES (?, 
?)", (username, hashed_password))
        conn.commit()
        messagebox.showinfo("Success", "Registration Successful!")
        username_entry.delete(0, tk.END)
        password_entry.delete(0, tk.END)
    except sqlite3.IntegrityError:
        messagebox.showerror("Error", "Username already exists!")

# Function to log in the user
def login():
    username = username_entry.get()
    password = password_entry.get()

    if username == "" or password == "":
        messagebox.showerror("Error", "Please fill in all fields.")
        return

    # Check credentials in the database
    c.execute("SELECT password FROM users WHERE username = ?", 
(username,))
    user = c.fetchone()

    if user:
        hashed_password = 
hashlib.sha256(password.encode('utf-8')).hexdigest()

        if hashed_password == user[0]:
            show_welcome(username)
            username_entry.delete(0, tk.END)
            password_entry.delete(0, tk.END)
        else:
            messagebox.showerror("Error", "Invalid username or 
password!")
    else:
        messagebox.showerror("Error", "Invalid username or password!")

# Function to show the password recovery page
def show_recovery():
    for widget in root.winfo_children():
        widget.destroy()  # Clear the main window

    recovery_label = tk.Label(root, text="Reset Your Password", 
font=('Helvetica', 16), bg='#D2B48C')
```

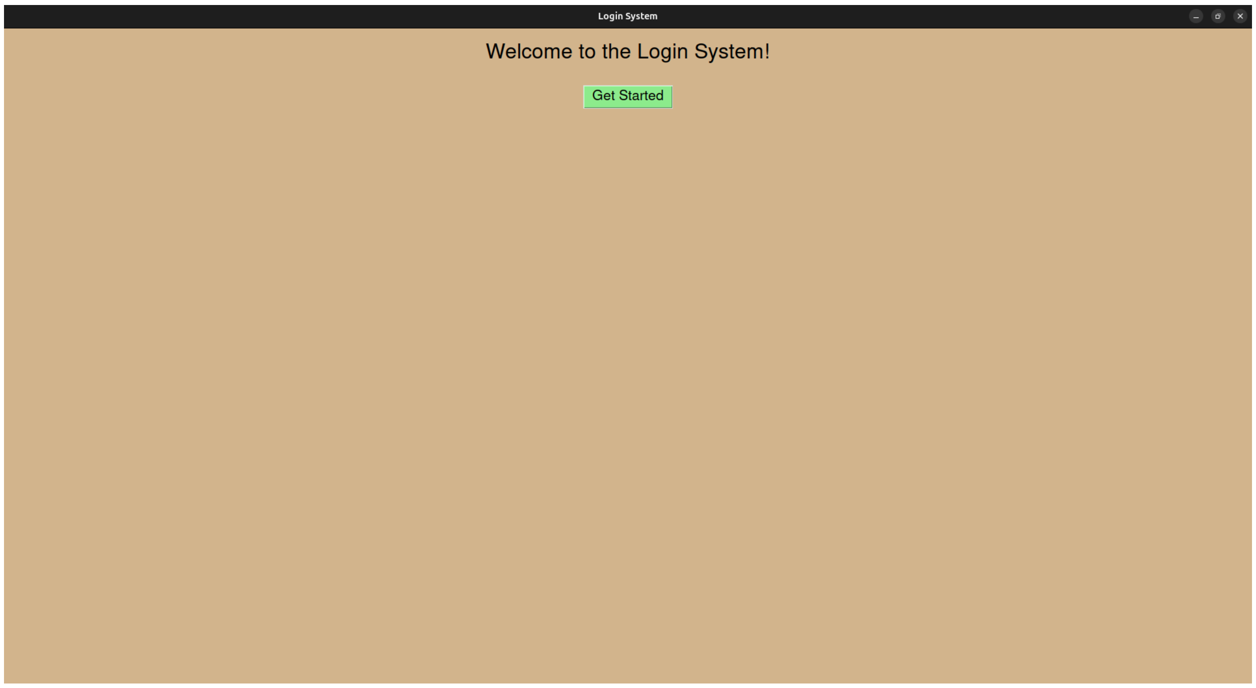
```
    recovery_label.pack(pady=10)

    global recovery_username_entry
    recovery_username_label = tk.Label(root, text="Username",
font=('Helvetica', 12), bg='#D2B48C')
    recovery_username_label.pack(pady=5)
    recovery_username_entry = tk.Entry(root, font=('Helvetica', 16),
width=30)
    recovery_username_entry.pack(pady=5)

    new_password_label = tk.Label(root, text="New Password",
font=('Helvetica', 12), bg='#D2B48C')
    new_password_label.pack(pady=5)
    global new_password_entry
    new_password_entry = tk.Entry(root, show='*', font=('Helvetica',
16), width=30)
    new_password_entry.pack(pady=5)

    recover_button = tk.Button(root, text="Reset Password",
command=reset_password, font=('Helvetica', 16), bg='lightblue',
activebackground='lightgreen')
    recover_button.pack(pady=10)

    back_button = tk.Button(root, text="Back to Login",
command=show_login, font=('Helvetica', 12), bg='lightyellow',
activebackground='lightgreen')
    back_button.pack(pady=5)

# Function to handle password reset
def reset_password():
    username = recovery_username_entry.get()
    new_password = new_password_entry.get()

    if username == "" or new_password == "":
        messagebox.showerror("Error", "Please fill in all fields.")
        return

    # Hash the new password
    hashed_password =
hashlib.sha256(new_password.encode('utf-8')).hexdigest()

    try:
        # Update the password in the database
        c.execute("UPDATE users SET password = ? WHERE username = ?",
(hashed_password, username))
        conn.commit()
        messagebox.showinfo("Success", "Password reset successful!")
        recovery_username_entry.delete(0, tk.END)
        new_password_entry.delete(0, tk.END)
        show_login()
    except sqlite3.Error:
        messagebox.showerror("Error", "Username not found!")
```

```
# Function to show the welcome message after login
def show_welcome(username):
    for widget in root.winfo_children():
        widget.destroy()  # Clear the main window

    welcome_label = tk.Label(root, text=f"Welcome, {username}!",
font=('Helvetica', 24), bg='#D2B48C')
    welcome_label.pack(pady=20)

    logout_button = tk.Button(root, text="Logout", command=show_login,
font=('Helvetica', 16), bg='lightblue',
activebackground='lightgreen')
    logout_button.pack(pady=10)

# Create the main window
root = tk.Tk()
root.title("Login System")
root.geometry("400x400")  # Set the size of the window
root.configure(bg='#D2B48C')  # Set the background color
show_welcome_page()  # Display the welcome page initially

# Run the application
root.mainloop()

# Close the database connection
conn.close()
```

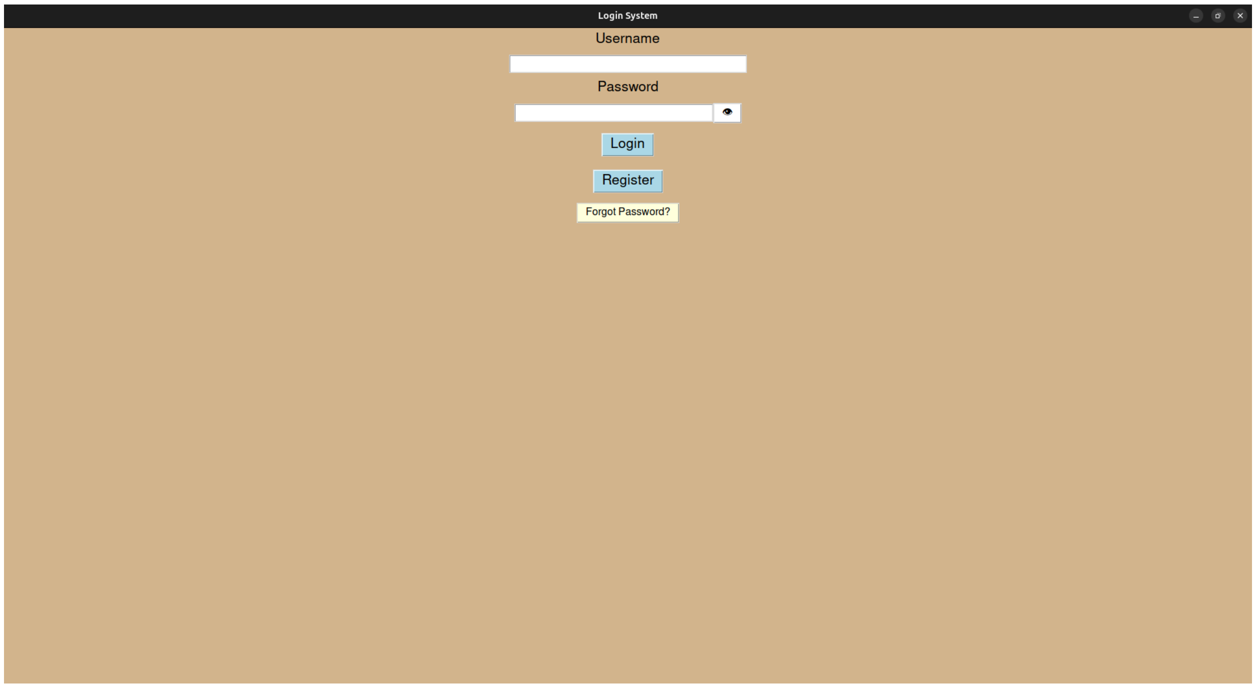
OUTPUT:-



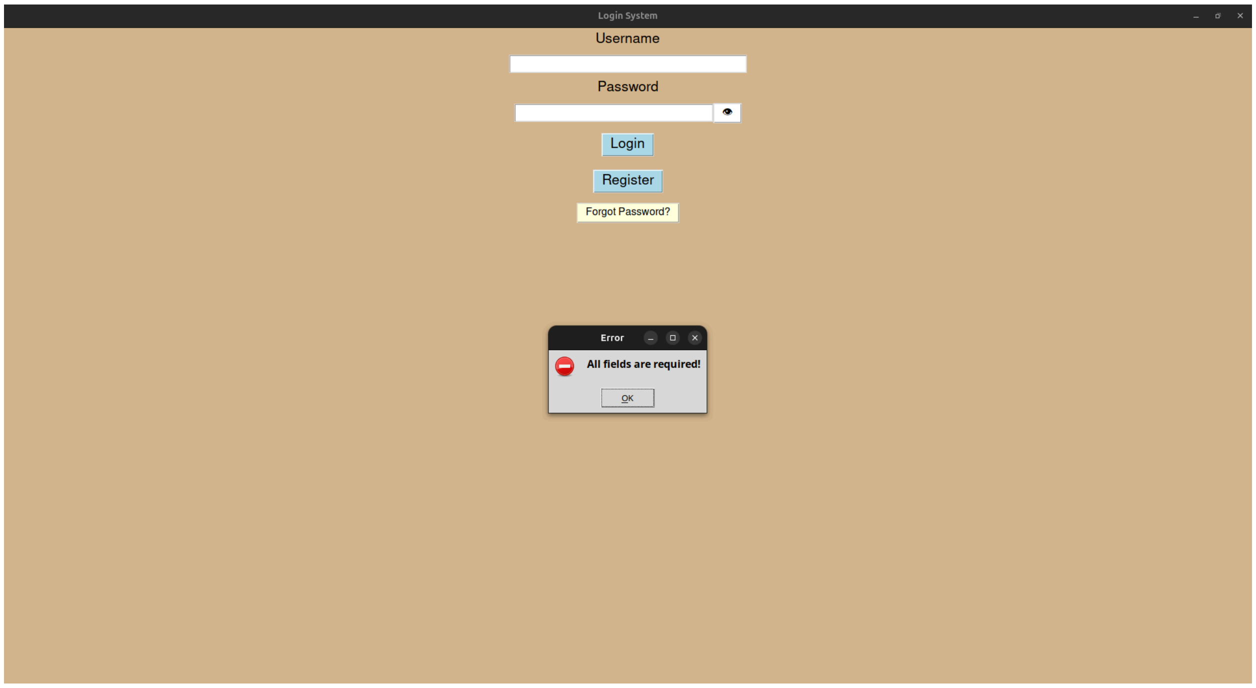
Front page



Button colour change on hovering.

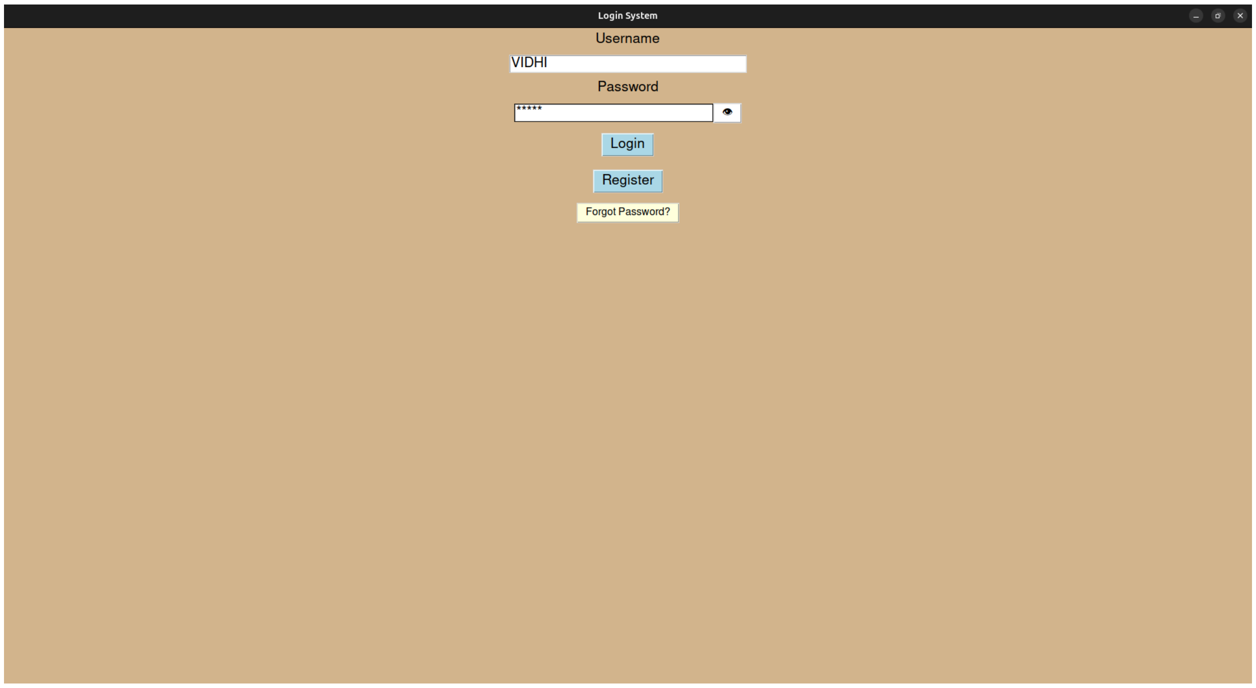


Login page

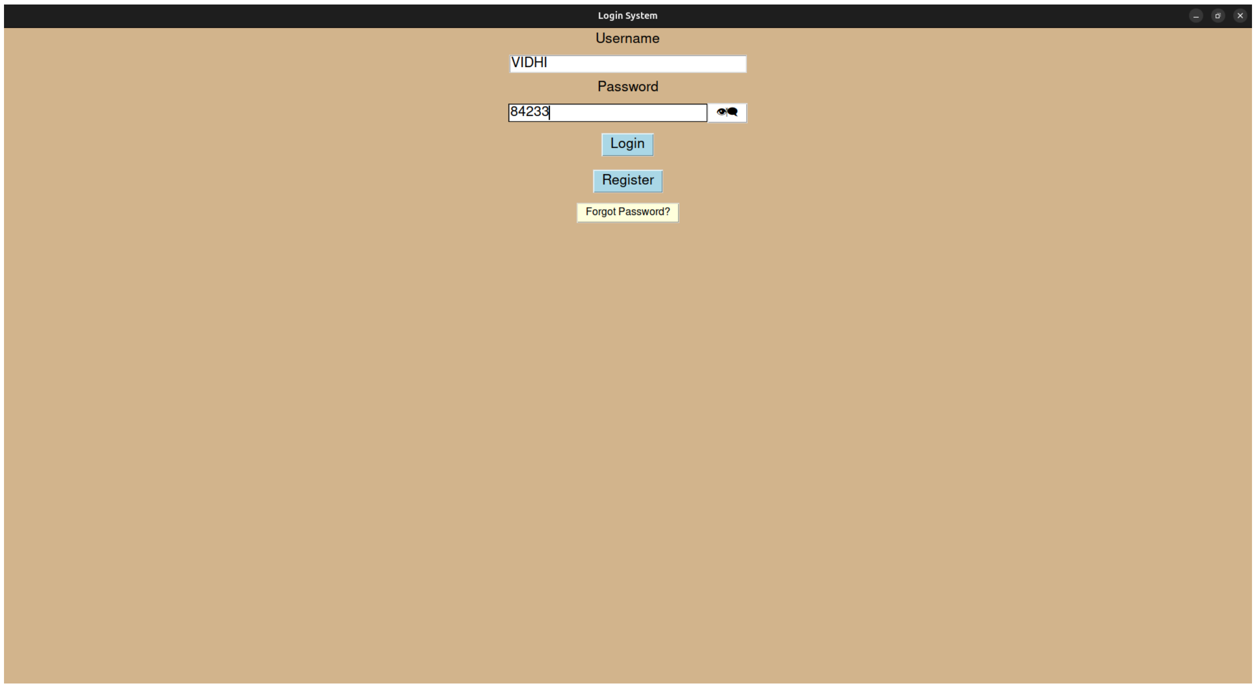


All fields are required!

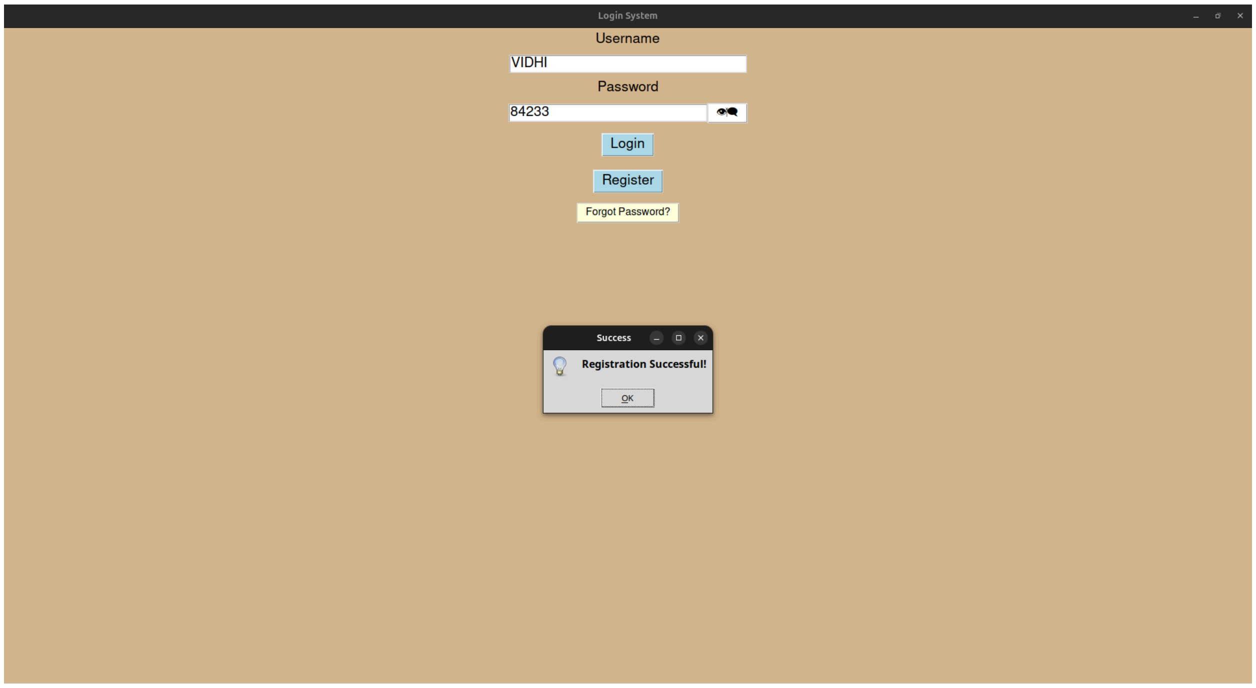
Shows invalid when you try to login/register without entering the required details.



Password not visible



Password visible


Login System

Username

VIDHI

Password

84233

Login

Register

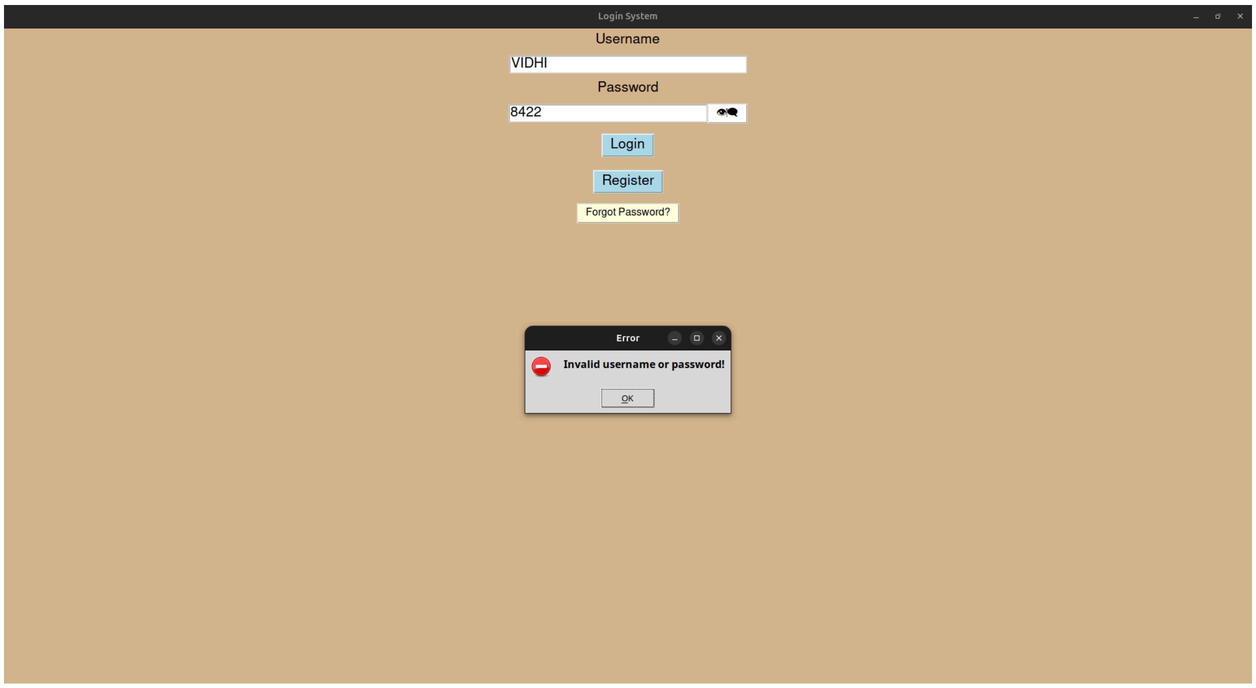
Forgot Password?

Success

Registration Successful!

OK

Registration successful


Login System

Username

VIDHI

Password

8422

Login

Register

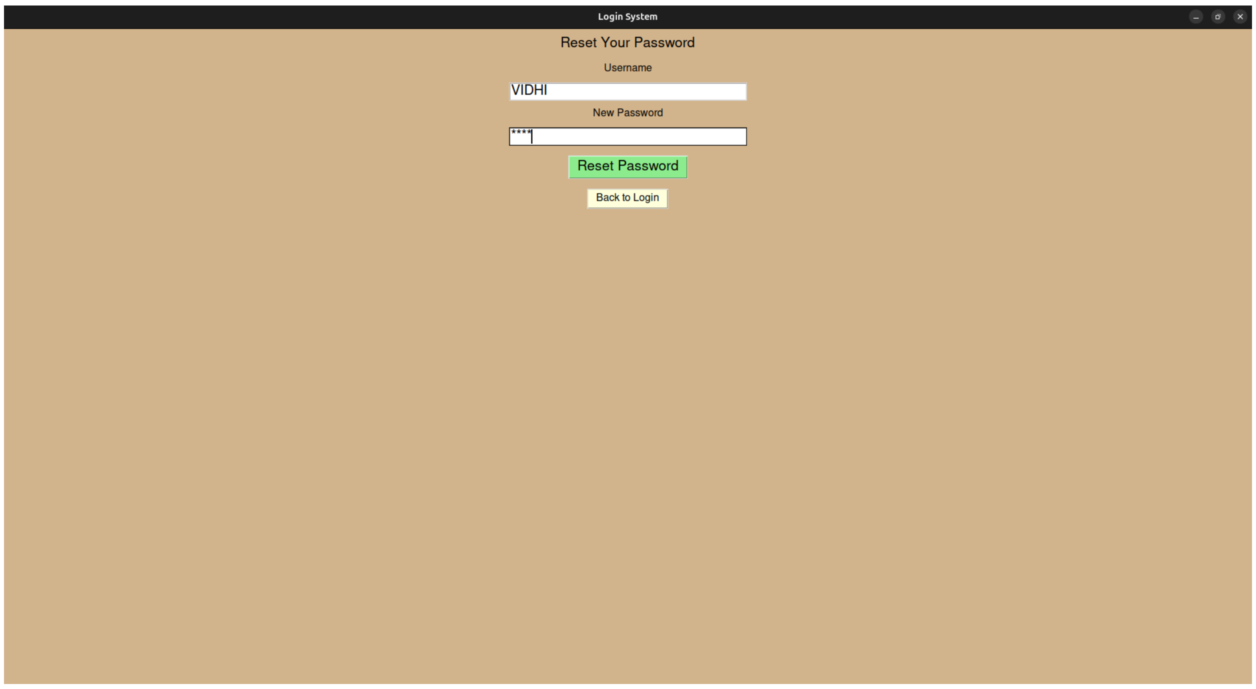
Forgot Password?

Error

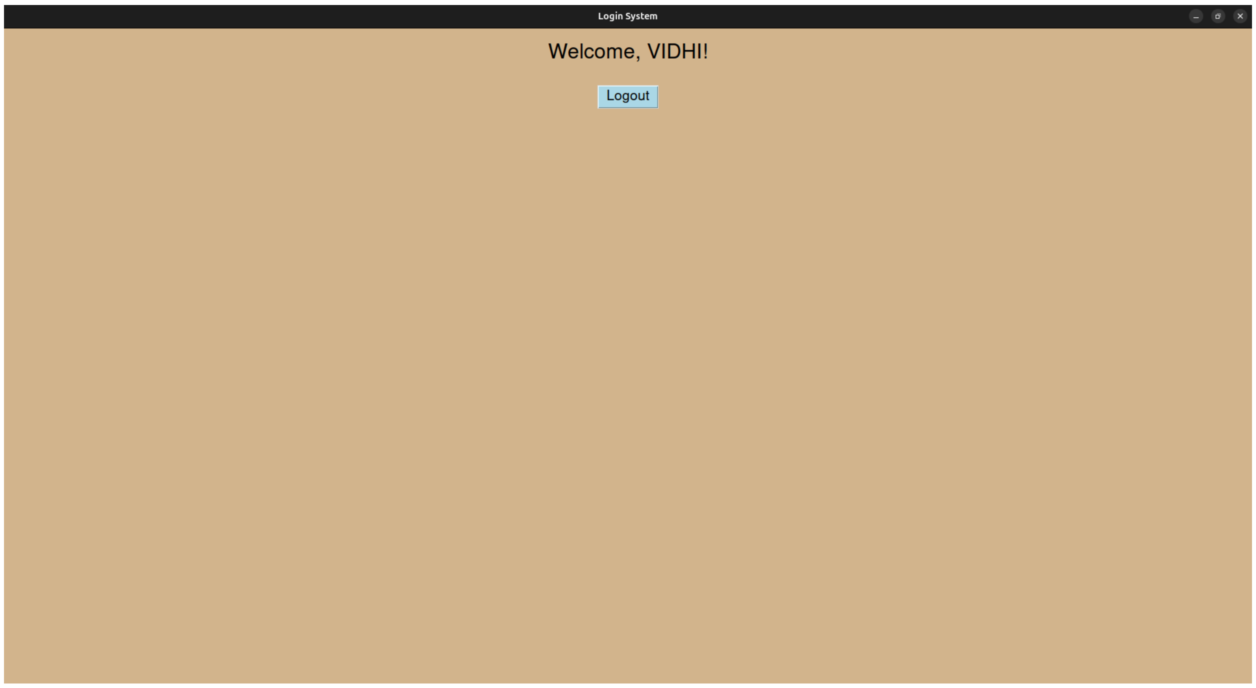
Invalid username or password!

OK

Invalid password/username shown incase of invalid password

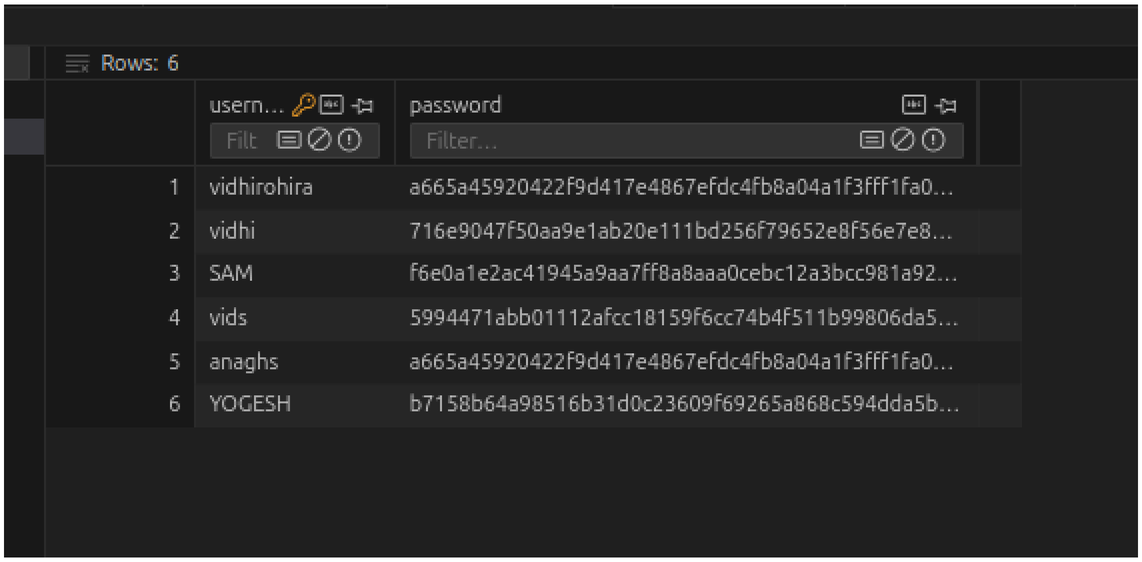


Feature to change password when you forget



Welcome, VIDHI!

Login page shown once you have logged in successfully



Rows: 6

	usern...	password
1	vidhirohira	a665a45920422f9d417e4867efdc4fb8a04a1f3fff1fa0...
2	vidhi	716e9047f50aa9e1ab20e111bd256f79652e8f56e7e8...
3	SAM	f6e0a1e2ac41945a9aa7ff8a8aaa0cebc12a3bcc981a92...
4	vids	5994471abb01112afcc18159f6cc74b4f511b99806da5...
5	anaghs	a665a45920422f9d417e4867efdc4fb8a04a1f3fff1fa0...
6	YOGESH	b7158b64a98516b31d0c23609f69265a868c594dda5b...

Database file

CONCLUSION:-

In this lab, we developed a simple login system using Tkinter for the graphical user interface and SQLite for data storage. We created a user registration and login process, allowing users to securely store their credentials with password hashing. The application also features password visibility toggling and a password recovery option. Through this exercise, we gained hands-on experience in integrating a database with a GUI, managing user input, and implementing basic security practices. Overall, this project provided valuable insights into building interactive applications with Python.