Macabodbod, Shanrick Lance R.

BSCS-1B

**A Documentation of my Graphical User Interface**

**Description of the Study**

In my recent studies, I've delved into creating a login form using GUI code in Python, focusing on understanding both user authentication and interface design principles. My journey began with choosing the right GUI library, and I opted for Tkinter due to its simplicity and comprehensive documentation. This library provided me with the necessary tools to construct an interactive and user-friendly form. My primary objective was to design an interface that allows users to input their credentials, such as a username and password, and validate these credentials effectively. This required a blend of visual design skills and a deep understanding of the underlying logic necessary for input validation, error handling, and secure data transmission.

Working on this project, I learned to create a clean and intuitive layout using Tkinter widgets like labels, entry fields, and buttons. I paid special attention to usability principles to ensure that the form was easy to navigate and user-friendly. On the technical side, I implemented secure data handling practices to protect user information. This included encrypting passwords and securely managing user data, which I stored in an SQLite database. Integrating the database required me to use SQLAlchemy for efficient data manipulation and retrieval. Additionally, I developed mechanisms for exception handling and user feedback, such as displaying error messages for incorrect login attempts. This hands-on experience significantly enhanced my understanding of creating secure and efficient authentication systems, blending both front-end design and back-end security practices.

**Benefits of the Code:**

Implementing a login form using GUI code in Python offers several significant benefits, both from a user and developer perspective. One of the primary advantages is enhanced security. By requiring users to authenticate before accessing sensitive information or functionalities, the login form acts as a gatekeeper, protecting the system from unauthorized access. This is crucial in any application that handles personal data, financial transactions, or other confidential information. The process of user authentication ensures that only legitimate users can access specific resources, which helps in maintaining data integrity and privacy. Moreover, with proper encryption and secure handling of credentials, developers can mitigate risks associated with data breaches and cyber-attacks.

Another benefit is the improvement in user experience. A well-designed login form provides a smooth and intuitive process for users to access their accounts. Features such as error messages for incorrect inputs, options for password recovery, and responsive design elements contribute to a seamless user journey. This is particularly important in retaining users and fostering trust in the application. From a developer's viewpoint, creating a login form with a GUI framework like Tkinter streamlines the development process, allowing for rapid prototyping and customization. It also provides the flexibility to integrate additional features such as two-factor authentication, social media logins, and more, enhancing the overall functionality and security of the application.

**Objectives of the Code:**

**key objectives:**

1. Enhance Security: The primary objective is to protect the application from unauthorized access. This involves implementing robust authentication mechanisms to ensure that only legitimate users can access sensitive information and functionalities. By encrypting user credentials and securely storing them, the login form aims to prevent data breaches and unauthorized access attempts.

2. User Authentication: Another objective is to accurately verify the identity of users. The login form must ensure that the credentials entered by the user match the stored data. This process involves validating the username and password against a database or predefined criteria, thereby establishing the authenticity of the user.

3. Improve User Experience: The login form should provide a seamless and intuitive experience for users. This includes designing a user-friendly interface with clear instructions, responsive elements, and helpful feedback messages. Ensuring that users can easily navigate the form and resolve any issues, such as forgotten passwords, is a critical objective.

4. Data Management: Efficiently handling and storing user data is a key objective. The login form should be integrated with a database system to manage user credentials securely. This involves not only storing data but also implementing features for data retrieval, updates, and deletion in a secure manner.

5. Accessibility and Usability: Ensuring that the login form is accessible to all users, including those with disabilities, is an important objective. This includes following best practices for accessibility, such as providing alternative text for screen readers and ensuring keyboard navigability.

6. Scalability and Flexibility: The login form should be designed to accommodate future enhancements and increased user load. This involves writing modular and maintainable code that can be easily updated to include additional security features, such as two-factor authentication, or to handle a larger number of users as the application grows.

**Purpose of the Code:**

The purpose of the login form code in a Python GUI application is to ensure secure, reliable, and user-friendly authentication of users. It serves as a critical security measure by verifying user identities and restricting access to authorized individuals only, thus protecting sensitive data and functionalities from unauthorized access. Additionally, the login form facilitates access control, allowing the application to differentiate between various user roles and provide appropriate levels of access. By implementing secure practices like password encryption, it safeguards user data against potential breaches.

**Significance of the Code:**

The significance of the login form code in a Python GUI application lies in its role as a cornerstone for both security and user management within the software. Firstly, it provides a critical security layer that ensures only authorized users can access sensitive information and functionalities. This is essential for protecting personal data, financial information, and any other confidential content from unauthorized access and potential breaches. By verifying user identities through secure authentication mechanisms, the login form helps maintain the integrity and confidentiality of the application’s data.

**Features of the Code:**

This code which is a login form using Tkinter in Python incorporates several key features that enhance its functionality, usability, and security.

1. Graphical User Interface (GUI): The code uses Tkinter to create a graphical user interface, which makes the login form visually appealing and user-friendly.

2. Image Integration: The login form includes images, such as a header image and an icon for the custom message box, enhancing the visual appeal. The code uses the PIL (Pillow) library to handle image resizing and conversion.

3. User Authentication: The `validate\_login` function checks the entered username and password against predefined credentials ("Shanrick" and "password"). This basic authentication mechanism ensures that only authorized users can log in.

4. Custom Message Box: The `custom\_messagebox` function creates a custom message box that displays a welcome message with an icon when the login is successful. This adds a personalized touch to the user experience.

5. Error Handling: The code uses Tkinter's `messagebox.showerror` to display an error message when the login credentials are invalid, providing immediate feedback to the user.

6. Password Masking: The password entry field uses the `show="\*"` attribute to mask the entered password, enhancing security by preventing onlookers from seeing the password.

7. Button and Labels: The code includes labels and an entry field for both the username and password, making the form intuitive and easy to use. The login button triggers the validation process.

8. Window Management: The code manages the main window and additional top-level windows (custom message boxes) effectively, ensuring a smooth user experience.

9. Image Handling: The code resizes images using the `Image.Resampling.LANCZOS` method from the Pillow library, which maintains high quality while resizing.

10. Grid Layout: The form uses the grid layout manager to organize widgets in a structured manner, making the interface clean and easy to navigate.

**The Code:**

import tkinter as tk

from tkinter import messagebox

from PIL import Image, ImageTk

def validate\_login():

username = username\_entry.get()

password = password\_entry.get()

if username == "Shanrick" and password == "password":

custom\_messagebox("Login Successful", "Welcome, Mr. Lance!",

"padlock.png") # Change "padlock.png" to your padlock icon file path

else:

messagebox.showerror("Login Failed", "Invalid username or password")

def custom\_messagebox(title, message, icon\_path):

top = tk.Toplevel()

top.title(title)

top.geometry("250x100")

icon = Image.open("C:\\Users\\creen\\Downloads\\fcb428cc-cdf5-4b76-a26d-26a3ac100d10.png")

icon = icon.resize((20, 20), Image.Resampling.LANCZOS)

icon = ImageTk.PhotoImage(icon)

label = tk.Label(top, text=message)

label.pack(side=tk.LEFT, padx=5)

icon\_label = tk.Label(top, image=icon)

icon\_label.image = icon # Keep a reference to avoid garbage collection

icon\_label.pack(side=tk.LEFT, padx=5)

button = tk.Button(top, text="OK", command=top.destroy)

button.pack(side=tk.LEFT, pady=5)

root = tk.Tk()

root.title("Login Form")

# Open and resize the image

image = Image.open("C:\\Users\\creen\\Downloads\\login-button-png-member-login-button-png-photos-574.png") # Change "your\_image.jpg" to your image file path

image = image.resize((400, 100), Image.Resampling.LANCZOS)

photo = ImageTk.PhotoImage(image)

# Convert the image to a format compatible with tkinter

header\_image = ImageTk.PhotoImage(image)

# Create a label to display the image

header\_label = tk.Label(root, image=header\_image)

header\_label.image = header\_image # Keep a reference to avoid garbage collection

header\_label.grid(row=0, columnspan=2, pady=10)

username\_label = tk.Label(root, text="Username:")

username\_label.grid(row=1, column=0, padx=10, pady=5)

username\_entry = tk.Entry(root)

username\_entry.grid(row=1, column=1, padx=15, pady=5)

password\_label = tk.Label(root, text="Password:")

password\_label.grid(row=2, column=0, padx=15, pady=5)

password\_entry = tk.Entry(root, show="\*")

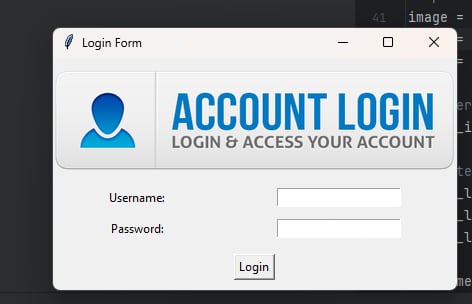
password\_entry.grid(row=2, column=1, padx=10, pady=5)

login\_button = tk.Button(root, text="Login", command=validate\_login)

login\_button.grid(row=3, column=0, columnspan=2, pady=10)

root.mainloop()

**Output:**

****

# 

**RESUME**

**Shanrick Lance R. Macabodbod**

Purok Magtatanim - Sitio Rizal, Matin-ao, Mainit, Surigao del Norte

09309664705

[smacabodbod@ssct.edu.ph](mailto:smacabodbod@ssct.edu.ph)

[2004shanrick@gmail.com](mailto:2004shanrick@gmail.com)

**Career Objectives:**

Self-motivated and adaptable developer with applicable database knowledge and coding skills. Organized programmer looking for an opportunity to grow as a developer and support an ambitious startup technology company. Collaborative communicator with relationship building skills.

**Education History:**

Tertiary - Surigao del Norte State University (SNSU) 2023-present

Surigao City, Surigao del Norte

Bachelor of Science in Computer Science

Secondary - Matin-ao National High School 2017-2023

Matin-ao, Mainit, Surigao del Norte

Primary - Matin-ao Central Elementary School 2011-2017

Matin-ao, Mainit, Surigao del Norte