1. Importing Required Libraries

```
import tkinter as tk
from tkinter import messagebox, colorchooser
import smtplib
import random
import string
from email.message import EmailMessage
```

1. **tkinter**: Provides the GUI functionality.
2. **messagebox**: Used for showing pop-up messages (error or info).
3. **colorchooser**: Allows the user to select a color for the background.
4. **smtplib**: Used for sending emails via SMTP (Simple Mail Transfer Protocol).
5. **random & string**: Used for generating the OTP.
6. **EmailMessage**: Used to construct and send the email message.

2. Email Configuration & Sample User Data

```
SMTP_SERVER = "smtp.gmail.com"
SMTP_PORT = 587
EMAIL_ADDRESS = "smayowa689@gmail.com"  # Sender email
EMAIL_PASSWORD = "Kgqaizzngrcgtnjm"    # Email app password

users = {
   "user1": {"password": "mypassword", "email": "user1@example.com", "security_answer":
"Group14"},
   # Other users...
}
```

- **SMTP Server Configuration**: Defines the SMTP server and port for sending emails via Gmail.
- **Users Dictionary**: Stores usernames, passwords, emails, and security answers for different users.

3. Global Variables

```
current_user = None
otp_code = None
selected_color = "#ffffff"  # Default background color
```

1. **current_user**: Holds the username of the currently logged-in user.
2. **otp_code**: Holds the generated OTP.
3. **selected_color**: Holds the background color for the GUI, initialized to white.

4. OTP Generation Function

```
def generate_otp():
    return ''.join(random.choices(string.digits, k=6))
```

- **generate_otp**: This function generates a 6-digit random OTP using digits (0-9).

5. Sending OTP Email Function

```
def send_otp_email(email, otp_code):
    msg = EmailMessage()
    msg['Subject'] = 'Your OTP Code'
    msg['From'] = EMAIL_ADDRESS
    msg['To'] = email
    msg.set_content(f"Your OTP code is: {otp_code}")

    try:
        with smtplib.SMTP(SMTP_SERVER, SMTP_PORT) as server:
            server.starttls()
            server.login(EMAIL_ADDRESS, EMAIL_PASSWORD)
            server.send_message(msg)
        messagebox.showinfo("OTP Sent", f"An OTP has been sent to {email}.")
    except Exception as e:
        messagebox.showerror("Email Error", f"Failed to send OTP email. Error: {e}")
```

- **send_otp_email**: This function creates and sends the OTP to the user's email. It uses the smtplib to connect to Gmail's SMTP server, sends the OTP, and shows a confirmation pop-up if successful or an error message if it fails.

6. OTP Validation Function

```
def validate_otp(entered_otp):
    if entered_otp == otp_code:
        otp_window.destroy()
        show_security_question()
    else:
        messagebox.showerror("Invalid OTP", "The OTP you entered is incorrect.")
```

- **validate_otp**: This function checks if the entered OTP matches the generated OTP. If correct, it proceeds to the next step (security question); if incorrect, an error message is shown.

7. Security Answer Validation Function

```
def validate_security_answer(answer):
    if answer.lower() == users[current_user]["security_answer"].lower():
        security_window.destroy()
        show_welcome_screen()
```

```
    else:
        messagebox.showerror("Invalid Answer", "Incorrect answer to the security question.")
```

- **validate_security_answer**: This function checks the user's answer to the security question. If correct, it proceeds to the welcome screen; if incorrect, an error message is shown.

8. Show OTP Screen

```
def show_otp_screen():
    global otp_code, otp_window
    otp_code = generate_otp()
    user_email = users[current_user]["email"]
    send_otp_email(user_email, otp_code)

    otp_window = tk.Toplevel()
    otp_window.title("Enter OTP")
    otp_window.config(bg=selected_color)
    otp_window.geometry("300x300")
    tk.Label(otp_window, text="Enter OTP sent to your email:",
bg=selected_color).pack(pady=10)
    otp_entry = tk.Entry(otp_window)
    otp_entry.pack(pady=5)

    def submit_otp():
        validate_otp(otp_entry.get().strip())

    tk.Button(otp_window, text="Submit", command=submit_otp).pack(pady=10)
```

- **show_otp_screen**: This function generates an OTP, sends it to the user's email, and opens a window where the user can enter the OTP to proceed. If the OTP is correct, it shows the security question.

9. Show Security Question Screen

```
def show_security_question():
    global security_window
    security_window = tk.Toplevel()
    security_window.title("Security Question")
    security_window.geometry("400x400")
    security_window.config(bg=selected_color)

    tk.Label(security_window, text="What is the name of Your swep group?",
bg=selected_color).pack(pady=10)
    answer_entry = tk.Entry(security_window)
    answer_entry.pack(pady=10)
```

```
    def submit_answer():
        validate_security_answer(answer_entry.get().strip())

    tk.Button(security_window, text="Submit", command=submit_answer).pack(pady=10)
```

- **show_security_question**: This function displays the security question and allows the user to enter the answer. If the answer is correct, the user is redirected to the welcome screen.

10. Show Welcome Screen

```
def show_welcome_screen():
    welcome_window = tk.Toplevel()
    welcome_window.title("Welcome")
    welcome_window.geometry("400x400")
    welcome_window.config(bg=selected_color)

    tk.Label(welcome_window, text="Welcome to the platform!", font=("Helvetica", 16),
bg=selected_color).pack(pady=20)
    tk.Button(welcome_window, text="Logout", command=
create_login_window).pack(pady=90)
```

- **show_welcome_screen**: This function shows a welcome message and a "Logout" button to return to the login screen.

11. Login Function

```
def login():
    global current_user
    username = username_entry.get().strip()
    password = password_entry.get().strip()

    if username in users and users[username]["password"] == password:
        current_user = username
        login_window.destroy()
        show_otp_screen()
    else:
        messagebox.showerror("Login Failed", "Incorrect username or password.")
```

- **login**: This function checks if the entered username and password match any in the users dictionary. If they do, it proceeds to the OTP screen; if not, an error message is displayed.

12. Color Selection Function

```
def choose_color():
    global selected_color
    color_code = colorchooser.askcolor(title="Choose Background Color")
    if color_code[1]:
        selected_color = color_code[1]
        login_window.config(bg=selected_color)
```

- **choose_color**: This function allows the user to pick a color for the background using the colorchooser module.

13. Logout Function

```
def logout():
    global login_window
    current_user = None
    for widget in login_window.winfo_children():
        widget.destroy()
    login_window.deiconify()
```

- **logout**: This function logs out the current user, resets the login window, and brings back the login screen.

14. Login Window Creation

```
def create_login_window():
    global login_window, username_entry, password_entry
    login_window = tk.Tk()
    login_window.title("2FA Login System")
    login_window.geometry("400x400")
    login_window.config(bg=selected_color)

    tk.Label(login_window, text="Username:", bg=selected_color, font=("Helvetica",
12)).pack(pady=10)
    username_entry = tk.Entry(login_window, font=("Helvetica", 12))
    username_entry.pack(pady=5)

    tk.Label(login_window, text="Password:", bg=selected_color, font=("Helvetica",
12)).pack(pady=10)
    password_entry = tk.Entry(login_window, show="*", font=("Helvetica", 12))
    password_entry.pack(pady=5)

    tk.Button(login_window, text="Choose Background Color", command=choose_color,
bg="#4CAF50", fg="white").pack(pady=10)
```

```
    tk.Button(login_window, text="Login", command=login, font=("Helvetica", 12),
bg="#4CAF50", fg="white").pack(pady=20)

    login_window.mainloop()
```

- **create_login_window**: This creates the main login window where the user can enter their username, password, and choose a background color.

15. Starting the Application

```
create_login_window()
```

- This line starts the program by displaying the login window.

Output:

1. **Login Window**: A window appears with fields to enter the username and password, and a button to select the background color.
2. **OTP Window**: After successful login, an OTP is sent to the user's email, and a window prompts the user to enter it.
3. **Security Question**: After entering the correct OTP, the user is asked a security question.
4. **Welcome Screen**: If the security question is answered correctly, the user is shown a welcome screen with a logout option.