**Facets internship assignment**

**Password Generator (using python)**

**Bhuvanashree H J (UVCE)**

**Code explanation:**

**1. Importing Modules & Global Variables**

First, we import the required modules – tkinter for GUI, messagebox for pop-ups, random for generating passwords, and string for character sets.

We also have two global variables:

* password\_history → Stores the last few generated passwords
* dark\_mode → Keeps track of whether dark mode is on or not

**2. Password Generation Function (generate\_password)**

This function does the main work – generating a strong password.

* First, it takes the length from the input field and makes sure it’s a valid positive number. If not, it throws an error.
* Then, it checks if the user has entered a custom word (optional). If the word itself is longer than the password length, it gives an error.
* After that, it builds a character set based on checkboxes (alphabets, numbers, special characters). If nothing is selected, it throws another error.
* Finally, it generates a random password (excluding the custom word length) and adds the custom word at the start.

It also updates the strength indicator and password history.

**3. Password Animation (animate\_password)**

Instead of just showing the password instantly, this function types it out letter by letter with a cool effect. Basically, it updates the label text with one character at a time using root.after(50, type\_out, index + 1).

**4. Checking Password Strength (update\_strength\_indicator)**

This function checks if the password has:

* Lowercase letters
* Uppercase letters
* Numbers
* Special characters

Based on the count of these, it assigns:

* **Weak (Red)** → If password length is too short
* **Medium (Orange)** → If password has less variety
* **Strong (Green)** → If it has everything

**5. Copying Password to Clipboard (copy\_to\_clipboard)**

This function clears the clipboard, copies the generated password, and shows a pop-up saying "Copied to Clipboard". Useful so you don’t have to manually copy it.

**6. Password History (update\_password\_history)**

Stores the last 5 generated passwords. Updates the history text box so you can check past passwords without generating again.

**7. Clearing Password History (clear\_history)**

Just wipes the history list and updates the text box to show nothing.

**8. Theme Toggle (toggle\_theme)**

This function flips between light and dark mode.

* **Light mode:** White background, black text
* **Dark mode:** Dark grey background, golden text, and dark blue buttons

It also calls set\_widget\_colors() to apply the theme to all widgets.

**9. Setting Widget Colors (set\_widget\_colors)**

Loops through all GUI elements (buttons, labels, entries) and updates their colors based on the selected theme.

**10. Setting Up the GUI (root = tk.Tk())**

Here, we set up the main window:

* **Fullscreen mode** so it covers the whole screen
* **Beige (changed to white) background**
* **Entry fields for password length & custom word**
* **Checkboxes** for including alphabets, numbers, and special characters
* **Buttons for generating password, copying it, clearing history, and toggling dark mode**
* **Password history displayed** with last 5 generated passwords

At the end, root.mainloop() starts the Tkinter GUI loop to keep the app running.

**Final Thoughts**

This is a well-structured password generator with a clean UI, animations, and dark mode. The code is modular, meaning each function does one specific task, making it easy to modify. We can further improve it by adding:  
✔ An option to set password complexity levels (Easy, Medium, Hard)  
✔ A way to save passwords securely (like hashing them)  
✔ A feature to exclude similar-looking characters (like O and 0, l and 1)