Password Generator App

# Workflow

1. The user provides the desired password length (minimum 4 characters).

2. The user selects character types to include: Alphabets, Numbers, Special Characters.

3. The app validates the inputs: at least one type must be selected.

4. Based on selected options, a pool of characters is created.

5. The app uses Python's `random.choice()` to generate a password of desired length from the selected pool.

6. The password is displayed in a styled code block using Streamlit's `st.code()` widget.

7. The entire app is run using Streamlit, requiring no backend or database.

8. Deployment is done via Streamlit Cloud, making the app accessible via a public URL.

# Assumptions

1. Users must enter a password length of at least 4 characters.

2. At least one character type (alphabets, numbers, special characters) must be selected.

3. The app is intended for general-purpose password generation, not for cryptographic-level security.

4. All password generation is done on the client side using Python and Streamlit.

5. No user data is stored; the app is stateless and does not require authentication.

6. The app is designed for simplicity: no external frontend or backend frameworks are used.

7. The codebase is a single file (`app.py`) for clarity and minimalism.

8. Streamlit UI is responsive and handles all interactions (input, validation, output).

9. Python’s built-in `string` module is used to access character sets.

10. The app is hosted on Streamlit Cloud or can be run locally with minimal setup.