# **Cyber-Shield: Password Strength Analyzer & Attack Simulator**

## **Overview**

Cyber-Shield is a web-based application designed to help users understand the security of their passwords. It not only assesses password strength but also simulates common cyber-attacks (like brute-force and dictionary attacks) to demonstrate potential vulnerabilities. The tool provides insights into the time it would take to crack a given password and offers practical suggestions for creating more robust and secure credentials.

## **Features**

* **Password Strength Analysis:** Get an instant rating of your password's complexity and resistance based on factors like length, character diversity, and common patterns.
* **Attack Simulation:**
  + **Brute-Force Attack Simulation:** Estimates the time it would take for a brute-force attack to crack your password.
  + **Dictionary Attack Simulation:** Identifies if your password is vulnerable to dictionary-based attacks.
* **Attack Analysis:** Provides a breakdown of the simulated attack methods and the estimated time taken for each.
* **Strong Password Suggestions:** Receive tailored recommendations to enhance your password's security.

## **Technologies Used**

* **Backend:** Python (Flask)
* **Frontend:** HTML, CSS, JavaScript
* **Password Strength Library:** zxcvbn-python

## **How to Run the Code Locally (For Teachers / Other Students)**

Follow these steps to set up and run the Cyber-Shield project on your local machine:

### **Prerequisites**

Before you begin, ensure you have the following installed:

* **Python 3.x:** (e.g., Python 3.8 or higher)
  + You can download it from [python.org](https://www.python.org/downloads/).
* **pip:** Python's package installer (usually comes with Python).

### **Step-by-Step Installation & Setup**

1. Clone the Repository:  
   Open your terminal or command prompt and run the following command to clone the project to your local machine:  
   git clone https://github.com/himanshunegi230111937/Cyber-Shield.git
2. Navigate into the Project Directory:  
   Change your current directory to the cloned project folder:  
   cd Cyber-Shield
3. Create a Virtual Environment (Recommended):  
   It's good practice to use a virtual environment to manage project dependencies. This prevents conflicts with other Python projects on your system.  
   python -m venv venv
4. **Activate the Virtual Environment:**
   * **On Windows:**  
     .\venv\Scripts\activate
   * **On macOS/Linux:**  
     source venv/bin/activate

You should see (venv) at the beginning of your terminal prompt, indicating the virtual environment is active.

1. Install Dependencies:  
   With your virtual environment activated, install all the required Python packages using pip:  
   pip install -r requirements.txt
2. Run the Flask Application:  
   Now you can start the Flask development server:  
   python app.py  
     
   You should see output similar to this, indicating the server is running:  
    \* Serving Flask app 'app'  
    \* Debug mode: on  
   WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.  
    \* Running on http://127.0.0.1:5000  
   Press CTRL+C to quit  
    \* Restarting with stat  
    \* Debugger is active!  
    \* Debugger PIN: XXX-XXX-XXX
3. Access the Web Interface:  
   Open your web browser and navigate to the address shown in the terminal (usually http://127.0.0.1:5000).  
   You should now see the Cyber-Shield application running!

### **Deactivating the Virtual Environment**

When you're done working on the project, you can deactivate the virtual environment by simply typing:

deactivate

## **Future Enhancements**

* **More Sophisticated Attack Simulations:** Implement more advanced attack vectors like rainbow table attacks, credential stuffing (simulated), or social engineering indicators.
* **User Authentication & History:** Allow users to create accounts and save their password analysis history.
* **Interactive Visualizations:** Enhance the attack analysis with dynamic charts or graphs to visualize crack times and attack probabilities.
* **Improved UI/UX:** Refine the user interface for a more intuitive and visually appealing experience.
* **Dockerization:** Provide a Dockerfile for easier deployment and environment setup.