**Password Analyzer and Wordlist Generator**

# Introduction

# In today’s digital-first world, users rely on passwords to secure personal, academic, and professional data. Despite their importance, passwords remain one of the weakest links in cybersecurity because many individuals continue to reuse or create predictable patterns. Weak choices such as "123456" or using birthdays make systems highly vulnerable to brute-force or dictionary-based attacks. This project was designed to bring awareness to password security by implementing a system that analyzes password robustness and generates targeted wordlists for security testing. Beyond technical functionality, the project also serves as an educational tool, showing how attackers think and why better password practices are essential for long-term digital safety.

# Abstract

The project introduces a two-part solution. The first part is the **Password Analyzer**, which evaluates a password using entropy calculations and the open-source zxcvbn library. This approach provides a numerical score, an estimate of cracking time, and direct suggestions for improvement. It ensures that users receive practical feedback instead of abstract technical terms.  
The second part is the **Wordlist Generator**, which transforms hints such as names, nicknames, dates, or locations into a structured dictionary of password candidates. By applying variations—like capitalization, leetspeak substitutions, reversing words, appending years, and adding common symbols—the generator creates realistic sets of potential passwords. This mirrors how attackers prepare their wordlists during penetration testing. Together, both modules provide a comprehensive learning experience by combining the perspective of defenders and attackers.

# Tools Used

# Python: Core programming language to implement the logic for both modules.

# zxcvbn Library: Provides realistic password strength evaluations by modeling real-world attacks.

# Regular Expressions (Regex): Used to detect numbers, years, and split input hints efficiently.

# JavaScript, HTML, CSS: Front-end technologies for building an interactive, browser-based GUI.

# Tkinter: Optional desktop GUI for users who prefer a standalone application.

# File Handling Modules: For exporting the generated wordlists into .txt format that can be used with password-cracking tools.

# Steps Involved in Building the Project

1. Collect input: accept either a password to be analyzed or hints (like names or birthdays) for wordlist generation.
2. Implement entropy calculations to mathematically evaluate unpredictability.
3. Integrate zxcvbn to simulate real-world attack scenarios and provide feedback in plain language.
4. Tokenize and process hints to build seed words. Apply multiple transformations such as case variation, leetspeak substitution, reversing, and attaching prefixes, suffixes, or years.
5. Filter and sort the generated candidates to remove duplicates and maintain reasonable wordlist size.
6. Create a preview window for users to inspect the generated wordlist before saving.
7. Add functionality to export the wordlist as a .txt file, ready for ethical hacking tools.
8. Build user interfaces: a command-line version for developers and a GUI/web version for general users.

**Conclusion**

This project demonstrates the importance of strong password practices in the simplest yet most practical way. By showing how quickly weak passwords can be analyzed and mimicked, it helps users understand the dangers of reusing predictable credentials. For cybersecurity learners, the project provides hands-on experience with concepts like entropy, dictionary attacks, and human password behavior. For professionals, it can serve as a lightweight tool for ethical penetration testing.  
The key lesson is that length and unpredictability matter more than complexity alone—passphrases and password managers offer stronger protection than short, clever-looking words. In the future, this project could be enhanced by integrating artificial intelligence to predict likely password variants, supporting multilingual wordlists, or extending the system to work as a browser plugin for real-time password guidance.