Password Strength Analyzer & Custom Wordlist Generator

□ Introduction

This project combines a modern password strength analyzer with an intelligent wordlist generator in a single Python tool. Designed with both a **GUI interface** (using Tkinter) and **CLI support**, it is perfect for users ranging from casual learners to penetration testers on platforms like **Kali Linux**.

Unlike basic tools, this version integrates **zxcvbn**, Dropbox's password strength estimator, to simulate real-world crack times and offer smart feedback.

🎯 Abstract

The tool evaluates the strength of a given password using:

- **Entropy-based calculations** (predictability estimation)
- **zxcvbn scoring** (pattern-aware strength detection)

It also allows users to enter personal details such as names, pet names, or dates, which are used to generate a **customized wordlist**. These are intelligently modified using:

- **Leetspeak variations**
- **Case combinations**
- **Special symbols and year patterns**

Such a feature set makes it extremely useful for:

- Red-teaming simulations
- Targeted password guessing
- CTF challenges
- Brute-force dictionary preparation

🛠 Tools & Technologies Used

- **Python 3**
- **GUI:** Tkinter (for interactive password checks)
- **CLI:** argparse (for command-line automation)
- **Password Analysis:** zxcvbn (DropBox's password strength estimator)
- **Regex, Math, Threading:** Built-in modules for analysis and generation

---

📌 Project Steps

1. **Built a base structure** using Python and argparse for CLI support.
2. **Integrated zxcvbn** to provide score, crack time, and intelligent suggestions.
3. **Created a GUI interface** with real-time password strength meter using Tkinter.
4. **Added password heuristics** (length, upper/lowercase, symbols, digits).
5. **Developed a wordlist generator** based on personal inputs:
    - Leetspeak (a → @, e → 3, etc.)
    - Case transformations
    - Symbol & year-based combinations
6. **Implemented export functionality** to save the generated wordlist as .txt.
7. **Used threading** to keep the GUI responsive during long wordlist generations.

□ Conclusion

This tool provides a practical solution for evaluating password strength while also enabling ethical hackers to simulate personalized dictionary attacks. By blending simplicity with power, and providing both visual and terminal-based outputs, it stands as a valuable addition to any cybersecurity toolkit—especially for learners, trainers, or professionals working in constrained environments like **Kali Linux**.