PROJECT REPORT

Title: Password Strength Analyzer with Custom Wordlist Generator

INTRODUCTION

In today's digital world, securing online accounts is crucial. Many users still choose weak or predictable passwords, making their accounts vulnerable to attacks. This project aims to provide a simple Python-based tool that analyzes the strength of user passwords and generates a custom wordlist based on personal information. The tool educates users on password safety and also demonstrates how attackers might create targeted password lists using known user details.

ABSTRACT

The project consists of two primary components:
1. A password strength analyzer using the zxcvbn library, which provides a score and crack time estimation for user-entered passwords.
2. A custom wordlist generator that takes personal inputs like name, birthdate, and pet name, then creates common password variants using those values.
The combination of these features helps both users and cybersecurity learners understand how password weaknesses are exploited and how to protect against them.

TOOLS USED

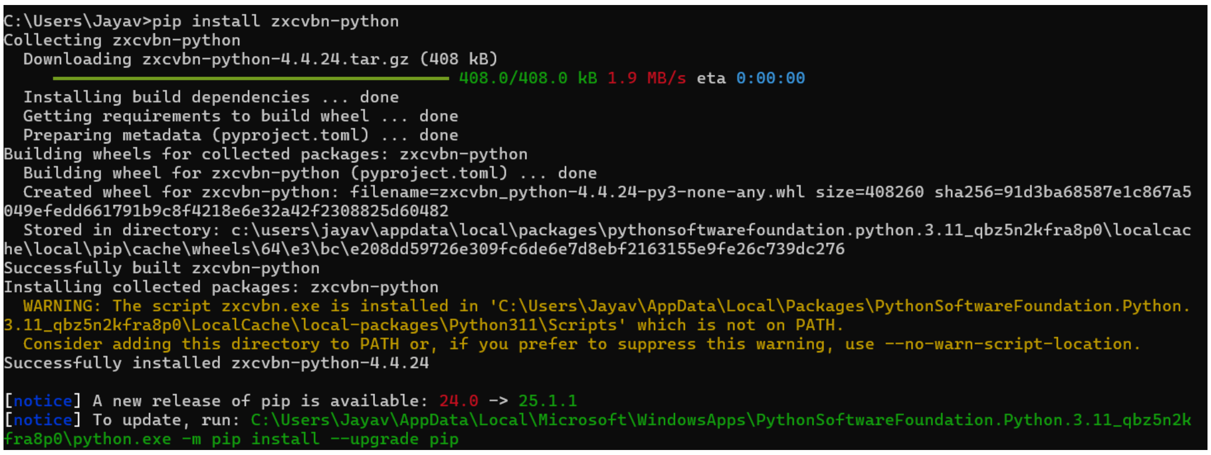
1. Python – Programming language used to develop the tool.
2. zxcvbn-python – Library to analyze password strength with realistic patterns.
3. File I/O – To save the generated wordlist as a .txt file.
4. Command-line Interface (CLI) – For user input and output display.

STEPS INVOLVED

1. Set up the environment and install the zxcvbn-python library.
2. Create a CLI tool that accepts user input for a password.
3. Analyze the password using zxcvbn() and display:
   Score (0 to 4)
   Estimated crack time
   Suggestions and warnings
4. Accept user input like name, date of birth, and pet name.
5. Generate a list of possible password combinations using patterns:
   Name + year (e.g., varshu2005)

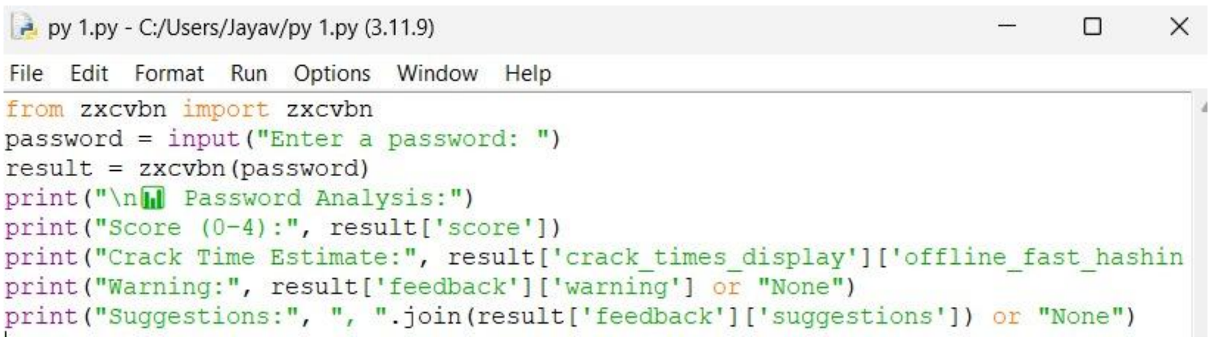
Reversed words, leetspeak, common endings like 123, @, etc.

6. Save the generated list into a file called **custom_wordlist.txt.**
7. Print a success message with number of words created.



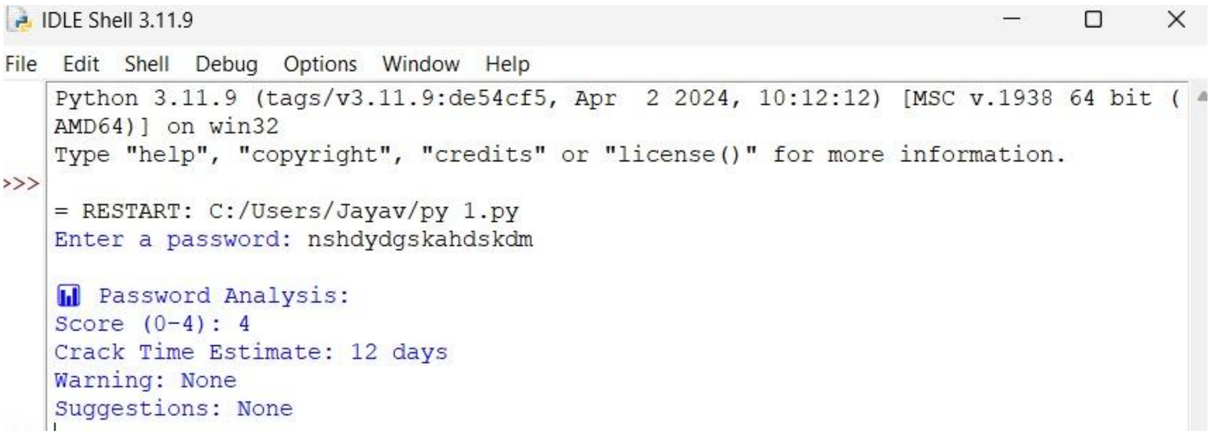
```
C:\Users\Jayav>pip install zxcvbn-python
Collecting zxcvbn-python
  Downloading zxcvbn-python-4.4.24.tar.gz (408 kB)
     ---------------------------------------- 408.0/408.0 kB 1.9 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: zxcvbn-python
  Building wheel for zxcvbn-python (pyproject.toml) ... done
  Created wheel for zxcvbn-python: filename=zxcvbn_python-4.4.24-py3-none-any.whl size=408260 sha256=91d3ba68587e1c867a5
049efedd661791b9c8f4218e6e32a42f2308825d60482
  Stored in directory: c:\users\jayav\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcac
he\local\pip\cache\wheels\64\e3\bc\e208dd59726e309fc6de6e7d8ebf2163155e9fe26c739dc276
Successfully built zxcvbn-python
Installing collected packages: zxcvbn-python
  WARNING: The script zxcvbn.exe is installed in 'C:\Users\Jayav\AppData\Local\Packages\PythonSoftwareFoundation.Python.
3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed zxcvbn-python-4.4.24

[notice] A new release of pip is available: 24.0 -> 25.1.1
[notice] To update, run: C:\Users\Jayav\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.11_qbz5n2k
fra8p0\python.exe -m pip install --upgrade pip
```

STEP 01 Install the zxcvbn-python library



py 1.py - C:/Users/Jayav/py 1.py (3.11.9)

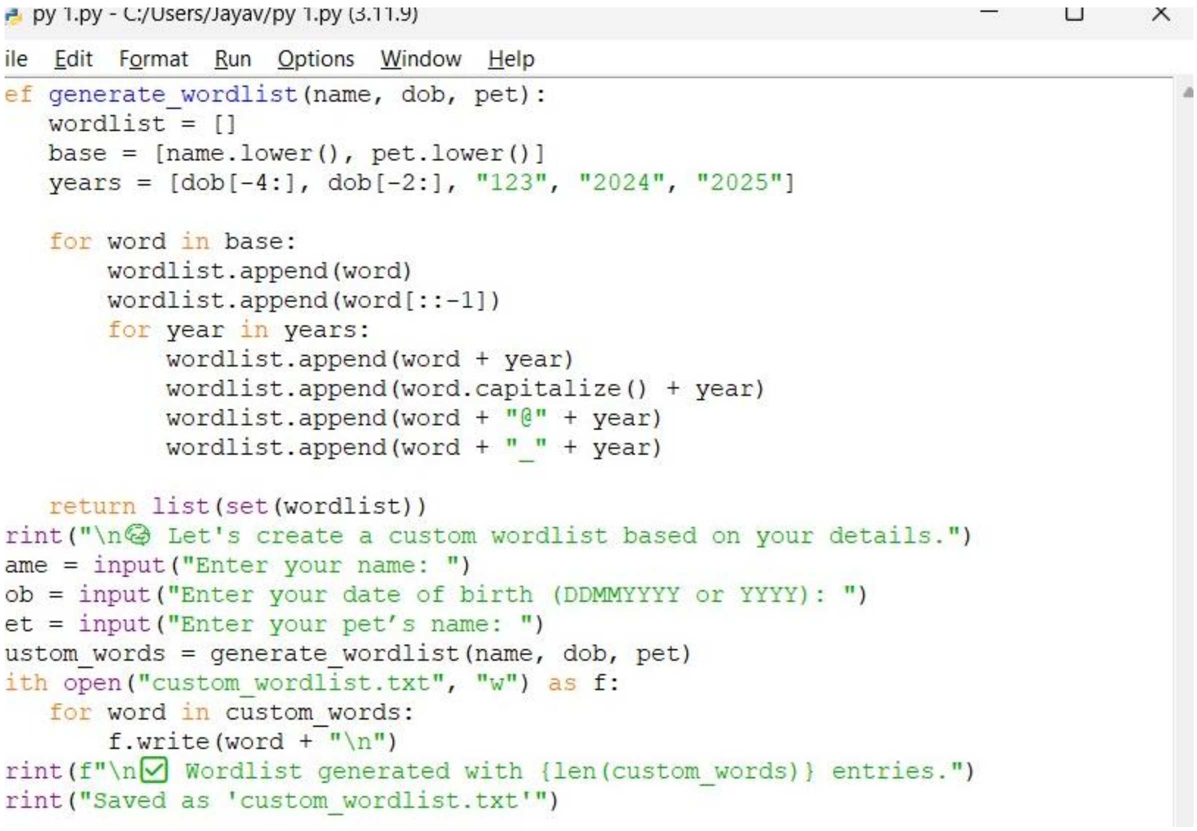
```python
from zxcvbn import zxcvbn
password = input("Enter a password: ")
result = zxcvbn(password)
print("\n📊 Password Analysis:")
print("Score (0-4):", result['score'])
print("Crack Time Estimate:", result['crack_times_display']['offline_fast_hashin
print("Warning:", result['feedback']['warning'] or "None")
print("Suggestions:", ", ".join(result['feedback']['suggestions']) or "None")
```



IDLE Shell 3.11.9

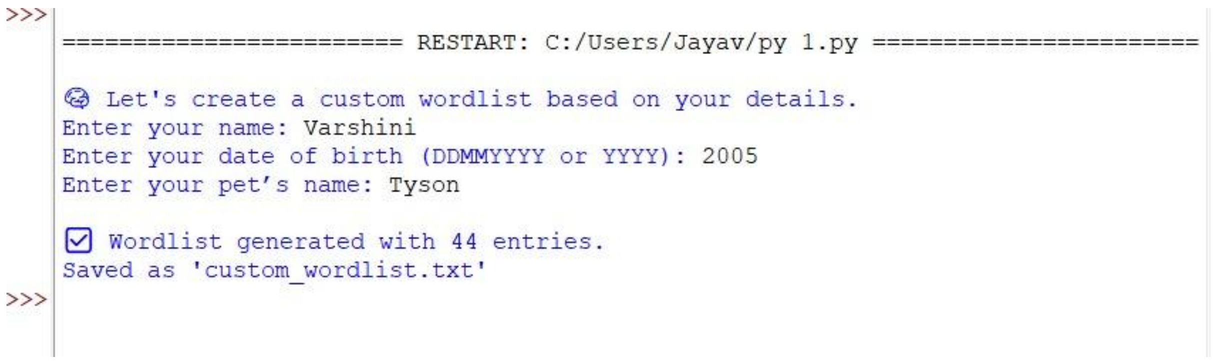
```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024, 10:12:12) [MSC v.1938 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Jayav/py 1.py
Enter a password: nshdydgskahdskdm

📊 Password Analysis:
Score (0-4): 4
Crack Time Estimate: 12 days
Warning: None
Suggestions: None
```

STEP 02 Analyzing the password

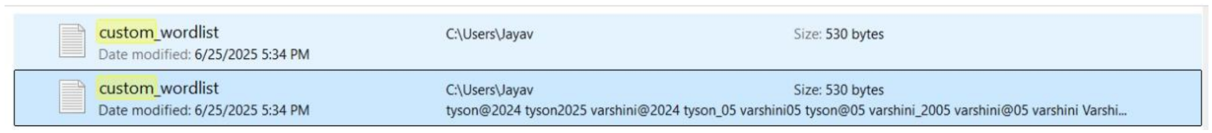

py 1.py - C:/Users/Jayav/py 1.py (3.11.9)
ile Edit Format Run Options Window Help

```
ef generate_wordlist(name, dob, pet):
    wordlist = []
    base = [name.lower(), pet.lower()]
    years = [dob[-4:], dob[-2:], "123", "2024", "2025"]

    for word in base:
        wordlist.append(word)
        wordlist.append(word[::-1])
        for year in years:
            wordlist.append(word + year)
            wordlist.append(word.capitalize() + year)
            wordlist.append(word + "@" + year)
            wordlist.append(word + "_" + year)

    return list(set(wordlist))
rint("\n🧠 Let's create a custom wordlist based on your details.")
ame = input("Enter your name: ")
ob = input("Enter your date of birth (DDMMYYYY or YYYY): ")
et = input("Enter your pet's name: ")
ustom_words = generate_wordlist(name, dob, pet)
ith open("custom_wordlist.txt", "w") as f:
    for word in custom_words:
        f.write(word + "\n")
rint(f"\n✅ Wordlist generated with {len(custom_words)} entries.")
rint("Saved as 'custom_wordlist.txt'")
```

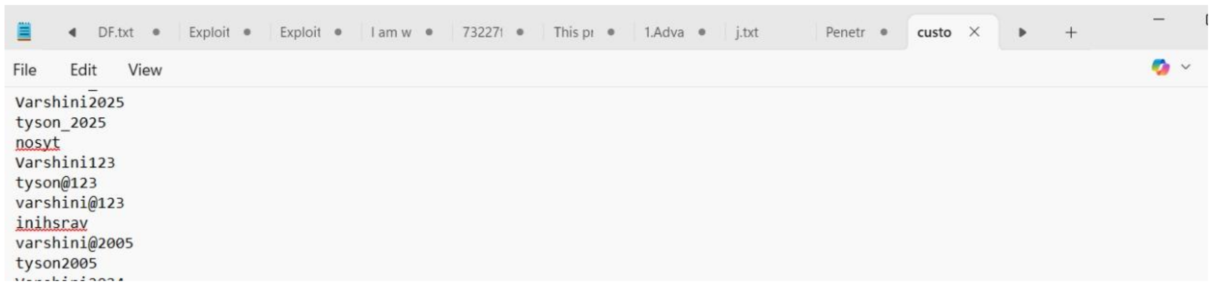
RESULT



```
>>>
    ======================== RESTART: C:/Users/Jayav/py 1.py ========================

    🧠 Let's create a custom wordlist based on your details.
    Enter your name: Varshini
    Enter your date of birth (DDMMYYYY or YYYY): 2005
    Enter your pet's name: Tyson

    ✅ Wordlist generated with 44 entries.
    Saved as 'custom_wordlist.txt'
>>>
```



```
custom_wordlist                      C:\Users\Jayav                Size: 530 bytes
Date modified: 6/25/2025 5:34 PM

custom_wordlist                      C:\Users\Jayav                Size: 530 bytes
Date modified: 6/25/2025 5:34 PM     tyson@2024 tyson2025 varshini@2024 tyson_05 varshini05 tyson@05 varshini_2005 varshini@05 varshini Varshi...
```



```
Varshini2025
tyson_2025
nosyt
Varshini123
tyson@123
varshini@123
inihsrav
varshini@2005
tyson2005
```

WORLIST CREATED USING THE PYTHON CODE

CONCLUSION

This project demonstrates the importance of strong, unpredictable passwords and the risks of using personal information in password creation. It also highlights how attackers can use basic data to generate probable password guesses. The tool is helpful for users who want to improve their password hygiene and for cybersecurity students learning how password cracking works. It is lightweight, educational, and adaptable for further development (e.g., adding GUI or integration into penetration testing tools).