Task-03

“

Password Complexity Checker

BY: OMAR MAHMOOD

Table of Contents

1. INTRODUCTION

- The Password Complexity Checker is a Python script designed to evaluate the strength of a given password based on specific criteria. The tool provides feedback to users regarding the password's strength, considering factors such as length, the presence of uppercase and lowercase letters, numbers, and special characters.

2. FEATURES

1.Password Strength Assessment:

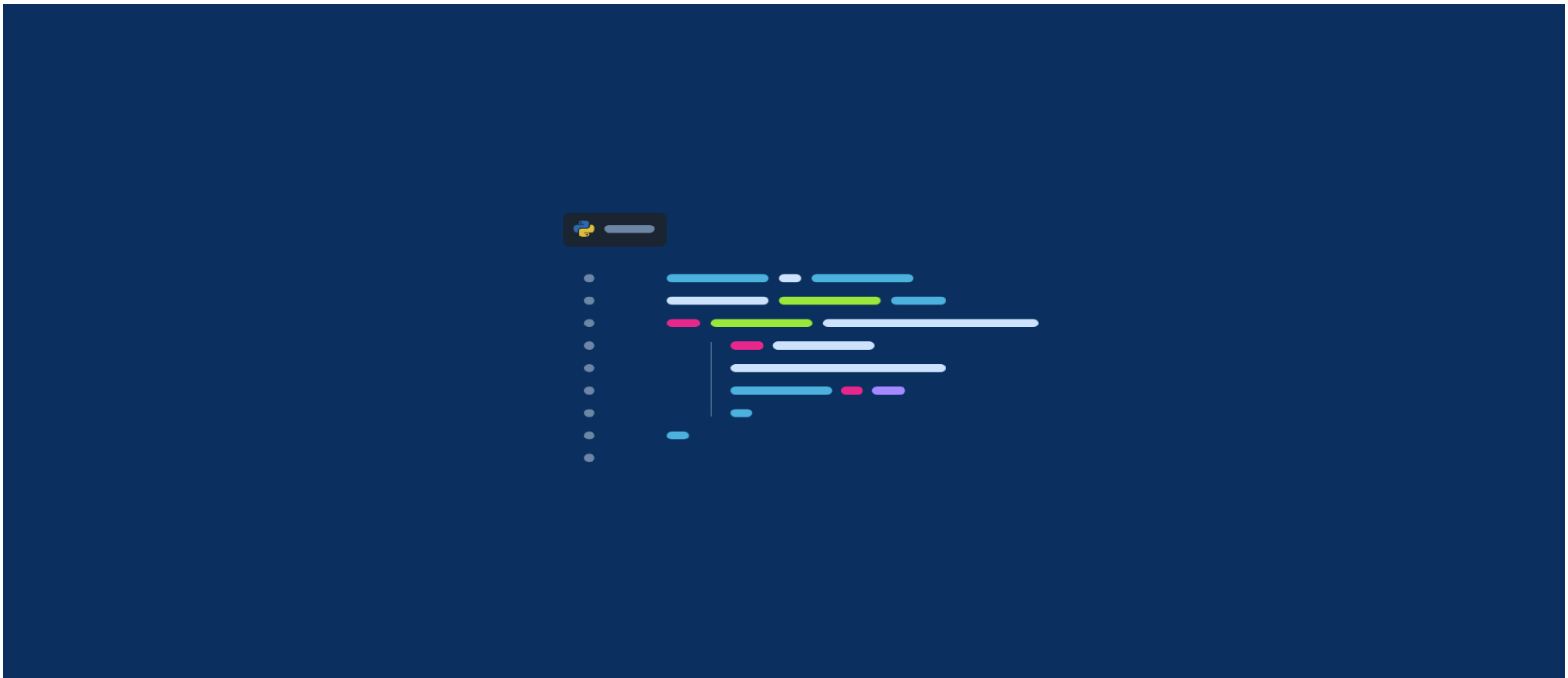
- The script analyzes the entered password against predefined criteria.
- Criteria include minimum length, at least one uppercase letter, one lowercase letter, one digit, and one special character.

2.Feedback Mechanism:

- Users receive immediate feedback on the strength of their password.
- Feedback is categorized into different levels, ranging from "Weak" to "Excellent".

3.Password Strength Meter:

- The script visually represents the password's strength through a percentage-based meter.



- The meter reflects the proportion of met criteria out of the total requirements.

4.Strong Password Examples:

- Users are provided with strong password examples based on their input.
- Examples demonstrate how to enhance the complexity of passwords.

5.User Interaction:

- The script offers a user-friendly interface, guiding users through the password complexity checking process.
- Real-time password input with asterisks ensures a secure and interactive experience.

3. USEAGE

1 Welcome Message:

- Upon execution, users are greeted with an ASCII art welcome message, providing an aesthetically pleasing introduction.

2 Password Input:

- Users are prompted to enter a password securely, with real-time asterisk masking.
- Password criteria are displayed for user reference.

3 Password Strength Feedback:

- The script evaluates the entered password against the criteria and displays a percentage-based strength meter.
- Users receive immediate feedback on the strength of

4 Strong Password Examples:

- Strong password examples are generated based on the user's input, offering practical suggestions.

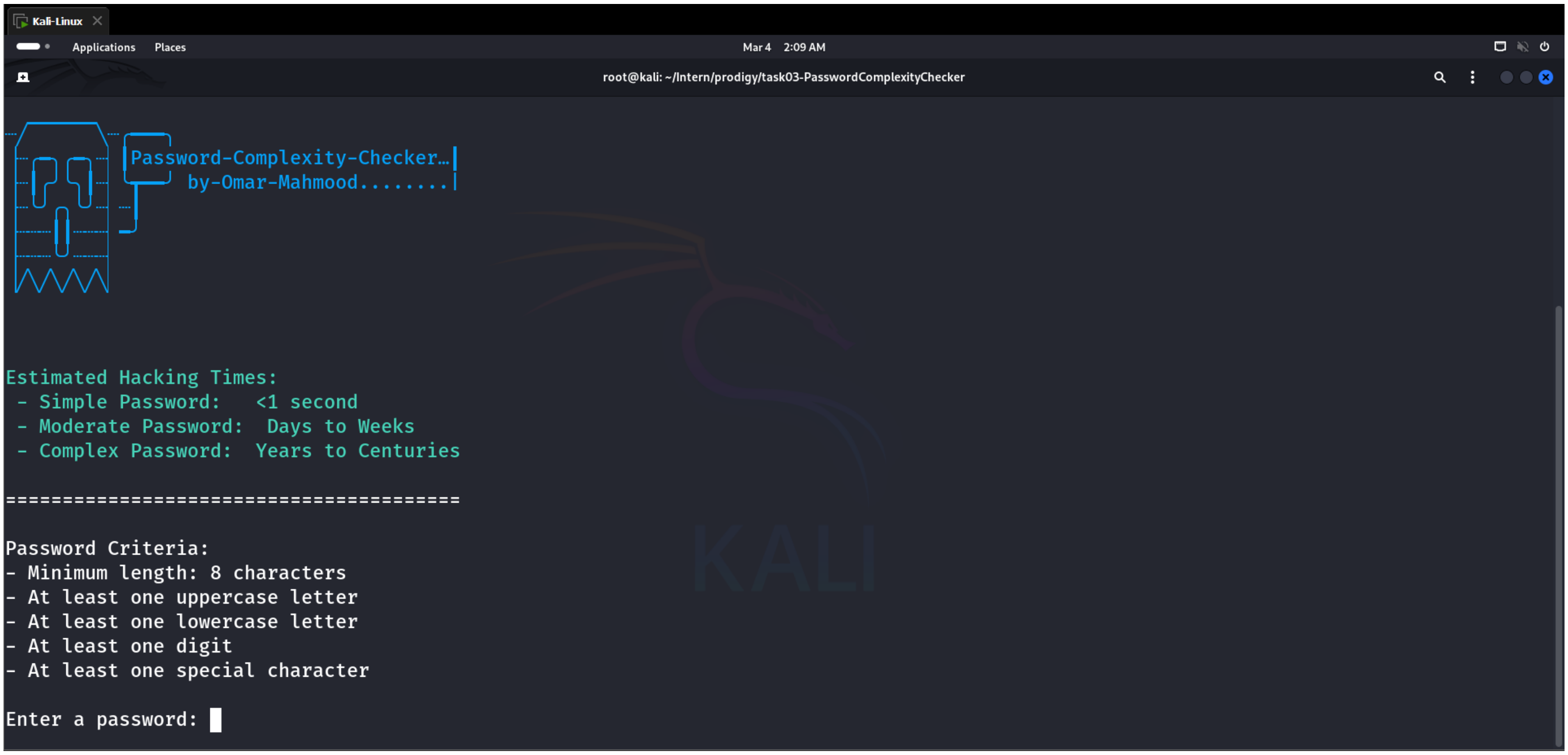
5 Interactive Experience:

- Users can choose to check another password for complexity or exit the script.
- The script handles user interactions gracefully.

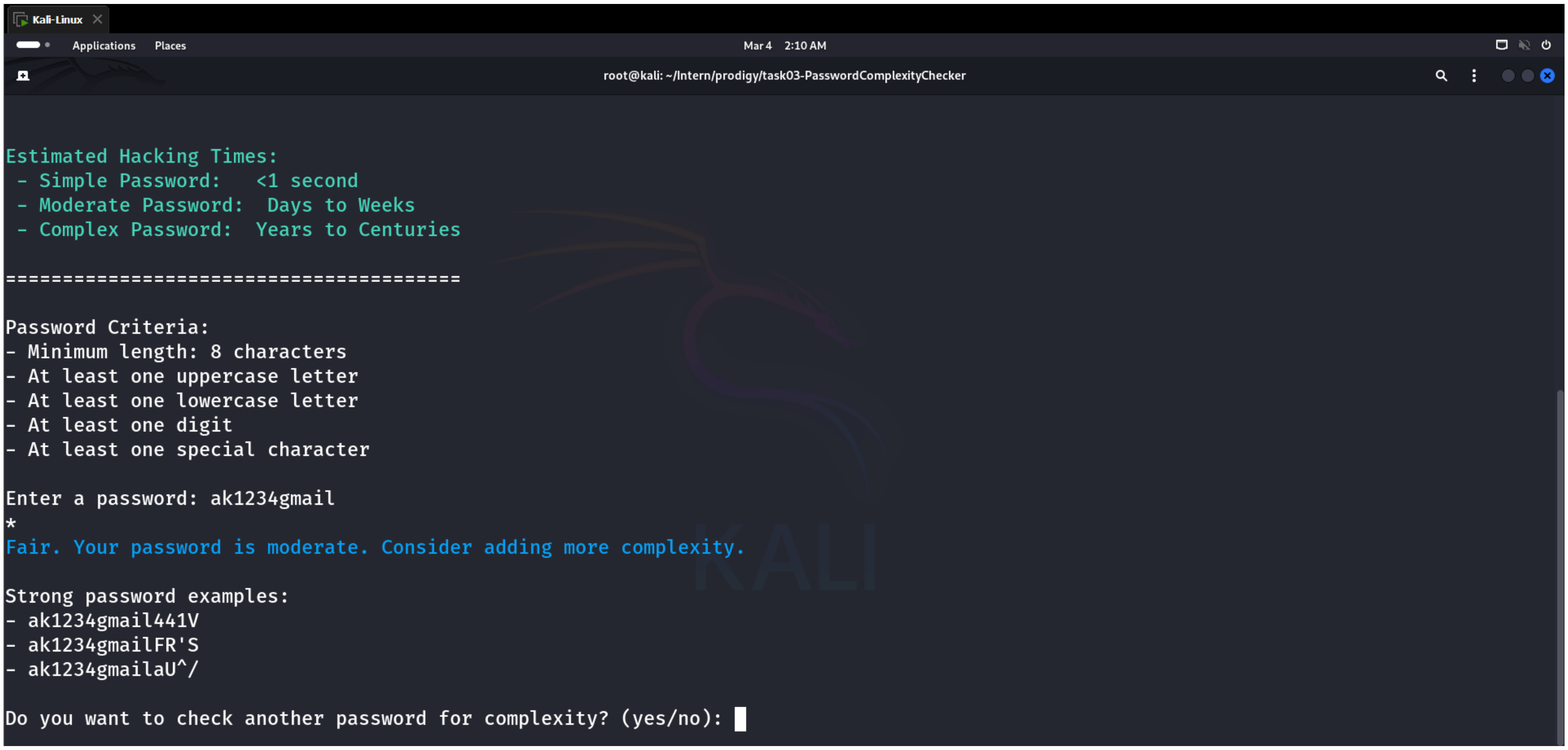
6 Thank You Message:

- Upon completion, users receive an ASCII art thank you message, enhancing the overall user experience.

5. SAMPLE OUTPUT


Kali-Linux
Applications
Places
Mar 4 2:09 AM
root@kali: ~/Intern/prodigy/task03-PasswordComplexityChecker

```
Password-Complexity-Checker…
     by-Omar-Mahmood........



Estimated Hacking Times:
 - Simple Password:   <1 second
 - Moderate Password:  Days to Weeks
 - Complex Password:  Years to Centuries

========================================

Password Criteria:
- Minimum length: 8 characters
- At least one uppercase letter
- At least one lowercase letter
- At least one digit
- At least one special character

Enter a password: 
```


Kali-Linux
Applications
Places
Mar 4 2:10 AM
root@kali: ~/Intern/prodigy/task03-PasswordComplexityChecker

```
Estimated Hacking Times:
 - Simple Password:   <1 second
 - Moderate Password:  Days to Weeks
 - Complex Password:  Years to Centuries

========================================

Password Criteria:
- Minimum length: 8 characters
- At least one uppercase letter
- At least one lowercase letter
- At least one digit
- At least one special character

Enter a password: ak1234gmail
*
Fair. Your password is moderate. Consider adding more complexity.

Strong password examples:
- ak1234gmail441V
- ak1234gmailFR'S
- ak1234gmailaU^/

Do you want to check another password for complexity? (yes/no):
```

6. CONCLUSION

The Password Complexity Checker script provides a comprehensive solution for assessing and improving password strength. Its user-friendly interface, real-time feedback, and engaging elements make it an effective tool for users seeking to enhance their password security.