Password Strength Checker Project Report

Name:
Priyanka Kolli

Hall Ticket No:
2406CYS115

Under the Guidance of:
Prof. Valli Kumari Vatsavayi
Professor of Computer Science and Systems Engineering, AU College of Engineering Andhra University, Visakhapatnam, Andhra Pradesh, India.

Table of Contents

1. Abstract

In the digital era, where cybersecurity threats have great influence, the project offers users a simple yet potent tool to fortify their online defences. This project uses the Python programming language and leverages the Flask web framework to deliver a seamless user experience. At its core lies a sophisticated algorithm designed to evaluate passwords against a comprehensive set of criteria. These criteria encompass not only the length of the password but also the diversity of character types, including uppercase and lowercase letters, numerals, and special characters. Through meticulous analysis, the algorithm assigns each password a strength score. The user interface of the project will be designed with simplicity and accessibility in mind. Users will be greeted with an intuitive web interface that lets them input their passwords for assessment. With a single click, users receive immediate feedback on the strength of their passwords. This seamless interaction empowers users to make informed decisions about their password security, fostering a culture of proactive cybersecurity awareness. Through this project, users will be equipped with the knowledge and resources they need to strengthen their passwords, increasing the overall resilience of online communities against cyberattacks. With the creative methodology and user-focused design, it opens the door to a more secure and safe digital environment.

2. Introduction

2.1 Problem Overview

In today's digital era, passwords serve as the primary method of securing access to personal and sensitive information. Despite their importance, weak passwords remain a common security vulnerability exploited by cyber attackers. Simple passwords, often chosen for convenience, are easily guessable and can lead to significant security breaches. Notable incidents, such as the Yahoo data breach and the LinkedIn breach, underscore the critical need for strong password practices.

Cyber attackers frequently employ techniques such as brute force attacks, dictionary attacks, and social engineering to crack weak passwords. The consequences of such breaches can be devastating, leading to identity theft, financial loss, and unauthorized access to confidential information. Therefore, educating users on the importance of strong passwords and providing tools to assess password strength is crucial in enhancing cybersecurity.

2.2 Importance of Strong Passwords

Strong passwords are essential in preventing unauthorized access to accounts and protecting sensitive information. A strong password typically combines length, complexity, and unpredictability. It includes a mix of uppercase and lowercase letters, numbers, and special characters, making it difficult for attackers to guess or crack.

Despite the availability of guidelines and tools, many users continue to create weak passwords. This behaviour is often driven by a lack of awareness or the perceived inconvenience of remembering complex passwords. The Password Strength Checker project addresses this issue by providing an easy-to-use tool that evaluates password strength and offers actionable feedback, helping users create stronger, more secure passwords.

3. Existing/Proposed System

3.1 Existing System

Currently, many users rely on simple, weak passwords due to convenience or lack of knowledge. Existing tools for password strength assessment often provide basic feedback, such as indicating whether a password meets minimum length requirements, but lack comprehensive analysis and user-friendly interfaces. These limitations leave users vulnerable to cyber-attacks.

Several online services and software applications include basic password strength meters. However, these tools often fall short in providing detailed feedback and education on creating strong passwords. Additionally, many existing tools are not integrated into a broader educational framework, limiting their effectiveness in promoting better password practices.

3.2 Proposed System

The Password Strength Checker offers an improved solution by using a detailed algorithm to evaluate passwords against multiple criteria, including length and character diversity. Implemented with Python and Flask, it provides an intuitive web interface that delivers immediate, actionable feedback, helping users create stronger, more secure passwords.

The proposed system includes several key features:

- Comprehensive Evaluation: The tool evaluates passwords based on multiple criteria, including length, use of uppercase and lowercase letters, numbers, and special characters.
- User-Friendly Interface: The web interface, developed using Flask, allows users to easily input their passwords and receive feedback.
- Actionable Feedback: The tool provides specific suggestions for improving weak passwords, helping users understand how to create stronger passwords.
- Educational Value: By offering detailed feedback and explanations, the tool educates users on the importance of strong passwords and encourages better password practices.

4. Objectives

The primary objectives of the Password Strength Checker project are as follows:

- Develop a User-Friendly Tool: Create a tool that accurately assesses password strength and provides an intuitive web interface for user interaction.
- Educate Users: Increase awareness of the importance of strong passwords and provide educational feedback to help users create more secure passwords.
- Provide Actionable Feedback: Offer specific suggestions for improving weak passwords, helping users understand how to enhance their password security.
- Enhance Cybersecurity Practices: Promote better password practices among users to improve overall cybersecurity and reduce the risk of unauthorized access.

These objectives are designed to address the current limitations of existing password strength assessment tools and provide a more comprehensive solution for enhancing password security.

5. Methodology/Framework

5.1 Criteria Definition

The first step in developing the Password Strength Checker was to define the criteria for evaluating password strength. These criteria include:

- Length: Passwords should be at least 8 characters long to be considered secure. Longer passwords are generally more secure, as they increase the complexity and time required for brute force attacks.
- Uppercase Letters: Inclusion of at least one uppercase letter (A-Z) to add complexity and make the password harder to guess.
- Lowercase Letters: Inclusion of at least one lowercase letter (a-z) to ensure a mix of character types.
- Numbers: Presence of at least one numerical digit (0-9) to further increase complexity.
- Special Characters: Use of at least one special character (e.g., !, @, #, $, etc.) to add an additional layer of complexity.

5.2 Algorithm Development

With the criteria defined, the next step was to develop an algorithm to evaluate passwords based on these criteria. The algorithm checks for the presence of each criterion and assigns a score to each password based on how well it meets these criteria. Passwords are categorized into "Weak," "Medium," and "Strong" based on their scores.

The algorithm uses regular expressions (regex) to check for different character types within the password. The steps involved in the algorithm are as follows:

- Check Length: Verify that the password meets the minimum length requirement (e.g., 8 characters).
- Check for Uppercase Letters: Use a regex pattern to check for the presence of uppercase letters.
- Check for Lowercase Letters: Use a regex pattern to check for the presence of lowercase letters.

- Check for Numbers: Use a regex pattern to check for the presence of numerical digits.
- Check for Special Characters: Use a regex pattern to check for the presence of special characters.
- Assign Score: Assign a score based on the number of criteria met.
- Categorize Password: Categorize the password as "Weak," "Medium," or "Strong" based on the score.

5.3 Web Interface Design

To make the tool accessible and user-friendly, a web interface was developed using Flask, a lightweight web framework for Python. The interface allows users to input their passwords and receive feedback on their strength.

The web interface includes the following components:

- Input Field: A text input field where users can enter their passwords.
- Submit Button: A button to trigger the password strength evaluation.
- Feedback Display: A section to display the strength of the password and provide specific suggestions for improvement.

5.4 Flowchart of the Process

A flowchart was created to visually represent the process of evaluating a password:

1. User Inputs Password: The user enters a password into the input field.
2. Trigger Evaluation: The user clicks the submit button to trigger the evaluation.
3. Evaluate Criteria: The algorithm checks the password against each defined criterion.
4. Calculate Score: The algorithm calculates a score based on the number of criteria met.
5. Categorize Password: The password is categorized as "Weak," "Medium," or "Strong."
6. Provide Feedback: Feedback is generated and displayed to the user, including suggestions for improving the password.

6. Software/Hardware Requirements

6.1 Software Requirements

- Python (latest version): Python is the core programming language used for developing the algorithm and the web interface.
- Flask: Flask is a lightweight web framework for Python, used to create the web interface and handle user interactions.
- Regular Expressions (re library in Python): Regular expressions are used to check for the presence of different character types in passwords.

6.2 Hardware Requirements

- Computer or Laptop: Any device capable of running Python and Flask along with internet access is sufficient for development and testing.

7. Programme

7.1 Install Flask

Install Flask using pip:

```
pip install flask
```

7.2 Create the Flask Application

Create a file named app.py and enter the following programme:

```
from flask import Flask, render_template, request
import re

app = Flask(__name__)

# Password strength criteria regex patterns
PATTERNS = {
    "length": r".{8,}",
    "uppercase": r"[A-Z]",
    "lowercase": r"[a-z]",
    "number": r"\d",
    "special": r"[!@#$%^&*(),.?\":{}|<>]"
}

def check_password_strength(password):
    """Check the strength of the given password."""
    strength = {
        "length": bool(re.search(PATTERNS["length"], password)),
        "uppercase": bool(re.search(PATTERNS["uppercase"], password)),
```

```
        "lowercase": bool(re.search(PATTERNS["lowercase"], password)),
        "number": bool(re.search(PATTERNS["number"], password)),
        "special": bool(re.search(PATTERNS["special"], password))
    }

    score = sum(strength.values())

    if score == 5:
        return "Strong", strength
    elif 3 <= score < 5:
        return "Medium", strength
    else:
        return "Weak", strength

@app.route("/", methods=["GET", "POST"])
def index():
    feedback = None
    strength_details = None

    if request.method == "POST":
        password = request.form.get("password")
        feedback, strength_details = check_password_strength(password)

    return render_template("index.html", feedback=feedback,
strength_details=strength_details)
```

```
if __name__ == "__main__":
    app.run(debug=True)
```

7.3 Create the HTML Template

Create a folder named templates in the same directory as app.py. Inside the templates folder, create a file named index.html and enter the following HTML programme for our interface:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Password Strength Checker</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            height: 100vh;
            background-color: #f0f0f0;
        }
        .container {
```

```
    background-color: white;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
.feedback {
    font-size: 1.2em;
    margin-top: 10px;
}
.criteria {
    margin-top: 10px;
}
.criteria li {
    list-style: none;
}
.criteria li span {
    font-weight: bold;
}
.weak {
    color: red;
}
.medium {
    color: orange;
}
.strong {
```

```
            color: green;
        }
    </style>
</head>
<body>
    <div class="container">
        <h1>Password Strength Checker</h1>
        <form method="POST">
            <input type="password" name="password" placeholder="Enter your password" required>
            <button type="submit">Check Strength</button>
        </form>
        {% if feedback %}
        <div class="feedback {{ feedback | lower }}">
            Password Strength: {{ feedback }}
        </div>
        <ul class="criteria">
            <li>Length (8+ characters): <span>{{ '✔' if strength_details['length'] else '✗' }}</span></li>
            <li>Uppercase Letter: <span>{{ '✔' if strength_details['uppercase'] else '✗' }}</span></li>
            <li>Lowercase Letter: <span>{{ '✔' if strength_details['lowercase'] else '✗' }}</span></li>
            <li>Number: <span>{{ '✔' if strength_details['number'] else '✗' }}</span></li>
```

Special Character: {{ '✔' if strength_details['special'] else '✗' }}

{% endif %}

</body>

</html>

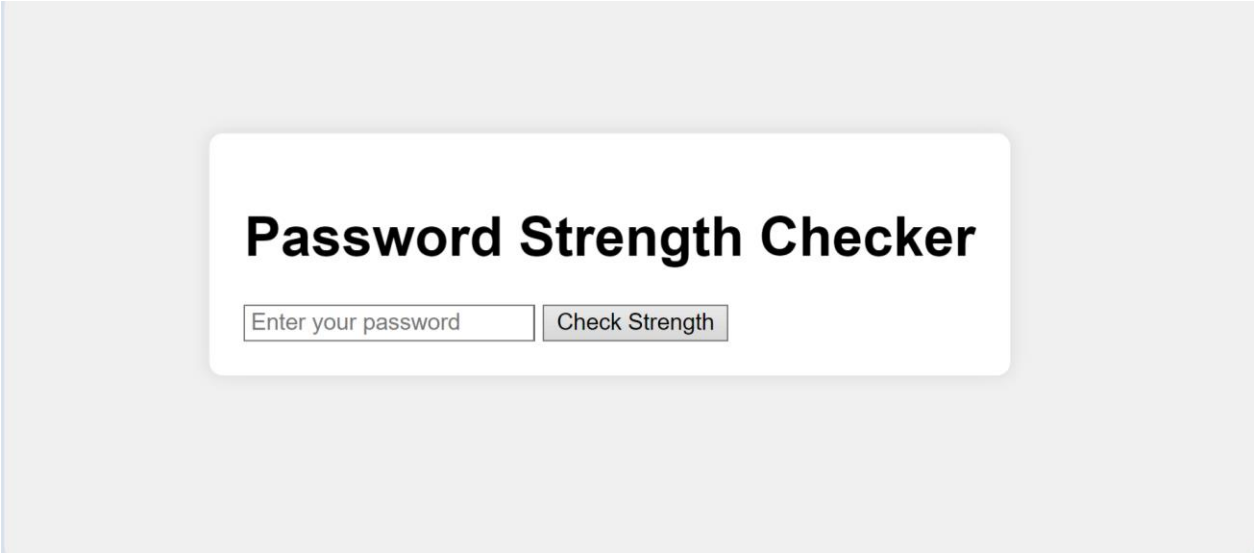
7.4 Run the Application

Navigate to the directory containing app.py and execute the following command:

python app.py

7.5 Access the Application

Open your web browser and go to http://127.0.0.1:5000/ or enter Ctrl + click the link displayed after running the application. You should see the Password Strength Checker interface. Enter a password and click "Check Strength" to see the evaluation and feedback.



8. Testing and Results

8.1 Testing Process

The Password Strength Checker was tested with various passwords to ensure accuracy and usability. The testing process involved evaluating passwords of different strengths and observing the feedback provided by the tool.

- Weak Passwords: Short passwords or those with common patterns. These passwords were expected to be categorized as "Weak" and receive feedback suggesting increasing length and adding diverse characters.
- Medium Passwords: Passwords that meet some but not all criteria. These passwords were expected to be categorized as "Medium" and receive feedback focused on improving specific criteria (e.g., adding special characters).
- Strong Passwords: Passwords that meet all criteria. These passwords were expected to be categorized as "Strong" and receive feedback confirming their strength and suggesting further enhancements if needed.

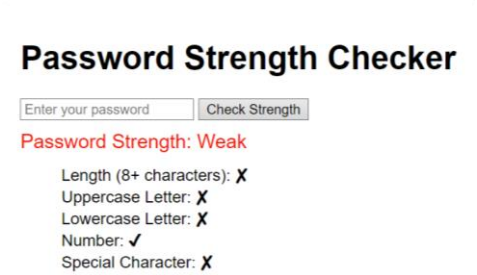
8.2 Results

The testing results confirmed the effectiveness of the tool in providing accurate feedback and suggestions:

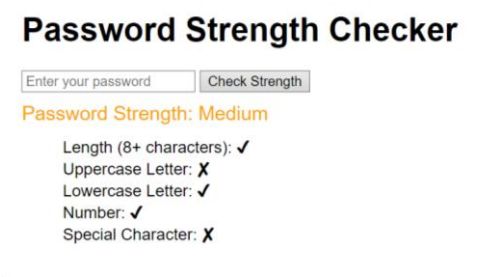
- Weak Passwords: The tool correctly identified weak passwords and provided specific suggestions for improvement, such as increasing length and adding a mix of uppercase letters, lowercase letters, numbers, and special characters.
- Medium Passwords: The tool accurately categorized medium-strength passwords and offered targeted feedback to help users enhance their passwords, such as adding special characters or increasing length.
- Strong Passwords: The tool effectively recognized strong passwords and confirmed their strength. In some cases, it provided additional tips for further improving password security.

8.3 Screenshots

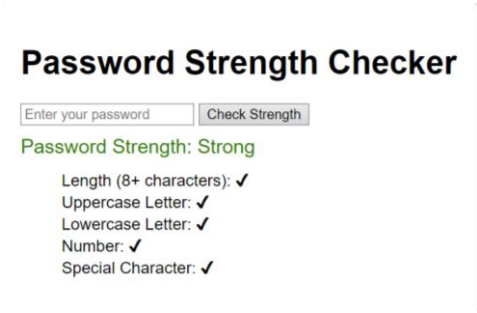
Test Password 1: 123456



Test Password 2: scde1234



Test Password 3: Scde@1234



Functionalities of the Password Strength Checker:

- Input Field and Submit Button: Show the user interface where users can enter their passwords and trigger the evaluation.
- Feedback Display: Display examples of the feedback provided for weak, medium, and strong passwords.

9. Conclusion

9.1 Summary

The Password Strength Checker project successfully demonstrates how a simple tool can significantly enhance user awareness and practices regarding password security. By providing immediate, actionable feedback, the tool helps users create stronger, more secure passwords, thereby reducing the risk of unauthorized access and cyber threats. This project highlights the importance of strong passwords in cybersecurity and serves as an educational platform to promote better password practices among users.

9.2 Educational Impact

The educational impact of the Password Strength Checker is significant, as it addresses a common vulnerability in cybersecurity. By providing users with a tool that evaluates password strength and offers actionable feedback, the project promotes better password practices and enhances overall cybersecurity awareness. The tool can be integrated into cybersecurity awareness programs in schools, universities, and businesses, helping to educate users on the importance of strong passwords and how to create them.

10. Future Scope

10.1 Potential Enhancements

There are several potential enhancements to the Password Strength Checker that could further improve its functionality and educational value:

- More Criteria: Additional criteria could be included in the evaluation, such as checking for common words or patterns, and ensuring that passwords do not contain easily guessable information.
- Integration with Password Managers: The tool could be integrated with password managers, allowing users to save and manage their passwords securely.
- Detailed Feedback: Providing more comprehensive feedback and suggestions for password improvement, including explanations of why certain criteria are important, could further educate users.

10.2 Real-World Applications

The Password Strength Checker has several real-world applications, particularly in enhancing password security in various contexts:

- Business Use: Businesses can use the tool to enforce password policies and ensure that employees are creating strong, secure passwords. This can help protect sensitive company information and reduce the risk of data breaches.
- Enterprise Integration: The tool can be integrated into enterprise security systems, providing an additional layer of protection by ensuring that all passwords used within the organization meet certain strength criteria.

11. References

List all sources, libraries, and tools used in the project:

1. Python documentation:

https://docs.python.org/

2. Flask documentation:

https://flask.palletsprojects.com/

3. Regular Expressions (re library) documentation:

https://docs.python.org/3/library/re.html

4. Geeks for Geeks:

https://www.geeksforgeeks.org/create-a-password-strength-checker-using-html-css-and-javascript/

5. GitHub:

https://github.com/SwamiTheDev/Password_strength