Report (part2)

Information Assurance and Security

Password Strength Tester

Prepared by :
Arij Kadhi
Tasnim Ben Brahim

Submitted to :
Dr. Manel Abdelkader

This project aims to create and put into use a password strength testing system that assesses passwords' resilience per OWASP guidelines. The design prioritizes user understanding and promotes the adoption of secure password/passphrase habits

Components of the System:

1. **Frontend Interface (UI)**
   The web page that the End User interacts with. Responsible for capturing input and displaying feedback dynamically.

Elements:

- Password Input Field (type="password")
- Username Input Field(Optional)
- Final Assessment Indicator ("Thumbs Up" icon / "Improve" or "Thumbs Down" icon)
- Informational Tips/Suggestions area.

2. **Backend Engine (Validation & Analysis)**

   Description: A server-side application or API that receives the password and context, performs the analysis based on configured rules, and returns the results.

   - Password parser and evaluator.
   - OWASP rule-based validator.
   - Feedback generator.
   - Backend (Strength Analysis Engine)

3. **Reporting and Logging**

   - Log entries for password testing (excluding the actual password).
   - Metrics on common weaknesses.

4. **Functional Flow:**
   - The user enters the username and proposed password.
   - System checks:

   a. Length ≥ 12 characters.

   b. Includes required character sets.

c. Does not contain parts of the username or known words.

d. Is not from blacklist.

e. Measures entropy and detects patterns.

- The system provides:

- A rating: Weak / Medium / Strong.

- Real-time suggestions to improve.

- If all checks pass, → " Password accepted."

From → To	Data	Purpose
User → System	Username, password	To validate password strength
System → User	Validation feedback	To guide improvement
System (internally)	Password metrics	To calculate entropy, check blacklist
System (logs)	Test outcomes	For usage and analytics (password never logged)

Tools and Technologies:

- **Frontend:** HTML5, CSS3, JavaScript (React or Vanilla JS)
- **Backend:** Python (Flask or FastAPI), Node.js (optional)
- **Libraries:** `zxcvbn`, `passlib`, `bcrypt`, `OWASP password policy` packages
- **Testing:** PyTest, Jest (for JS parts)