

Information assurance & security

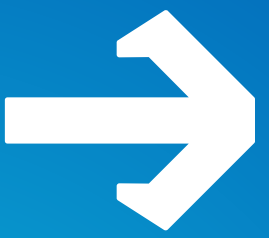
# PROJECT

---

By Yassine Marrekchi



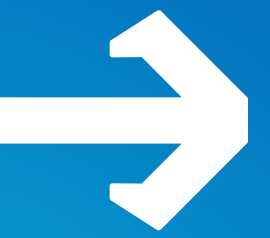
- sensitive information are facing unauthorized access risk which which arises the need for strong passwords .



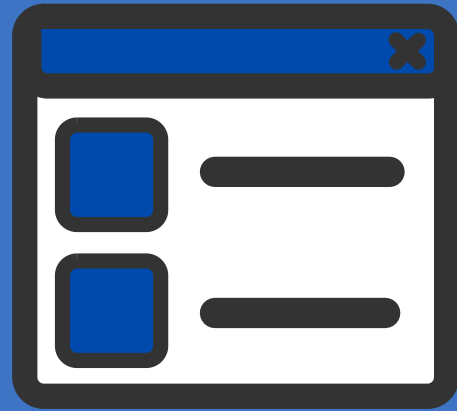
**OBJECTIVE :** test  
password  
strength using  
OWASP standards



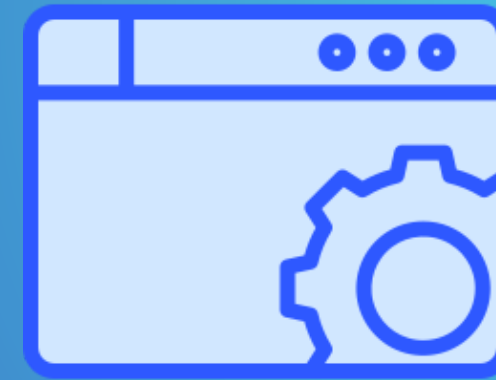
# Main components



## Input interface



## Backend Logic



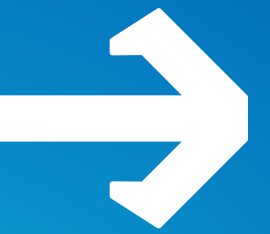
## Passphrase Generator



## Feedback Mechanism



# Input interface



Secure text field for username and password

username

password

Buttons for checking strength and generating passphrase

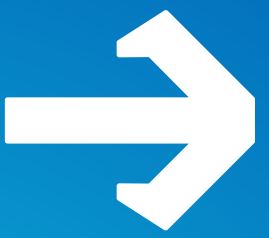
**CHECK  
STRENGTH**

**GENERATE  
PASSPHRASE**

Show/Hide Password feature



# Backend Logic

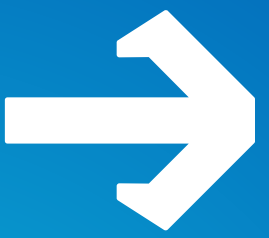


- Password strength evaluation algorithm
- Password validation upon prior data

- Passphrase Generator :
  - Generates random and secure passphrases to use as a password.
  - Higher security due to increased entropy.



# Password Strength Criteria



**Minimum Length** : 8-12 characters



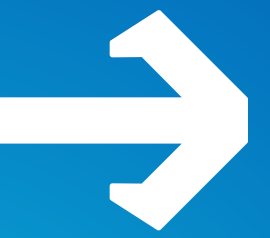
**Complexity** : Mix of characters (upper/lowercase, digits, special )



Avoid sequential characters and public information



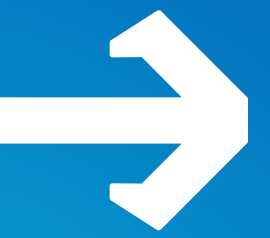
# Password Strength Levels



- **Weak** : Less than 8 characters, lacks diversity.
- **Moderate** : 8-12 characters , mixed types.
- **Strong** : More than 12 characters + complexity, diverse and unpredictable.



# Advantages of the solution



- Reduces risks of identity theft and cyber breaches.

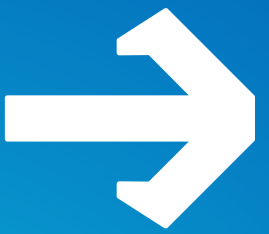
- User-friendly interface and real-time feedback.

- Integration with password management systems.





# Limitations of existing solutions



- **Authentication gaps**: ignoring the use of personal information as a password ( e.g : phone number )

- **Language restrictions**:

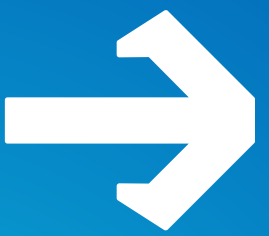
English



- **Limitations of criterion**:

ignores other important features such as vulnerability to brute force attacks

# Tools used in this project



- **JavaScript :**
  - basic input functions
  - passphrase generator mechanism
  - evaluation mechanism
  - SHA-256 hash + crack time estimation
  - hide/show password



- **HTML :**
  - structuring



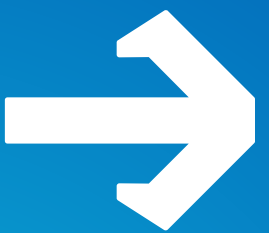
- **CSS :**
  - Layout aesthetics



- **JSON databse :**
  - for commonly used passwords



# Overview of the interface



**Buttons** for :

- Evaluating strength
- Generating passphrase

fields to input credentials

The screenshot shows the EVALUATOR interface with the following elements:

- Input field for username: "yassine"
- Input field for password: "myPassword\*230"
- Buttons: "Show/Hide Password" and "Check Strength"
- Feedback section: "Password Strength: Strong"
- Output section: "SHA-256 hash: ad2c95520a8002ed20799e73e7c243e6004e37005f84f9f115345d510229cf15" and "Time to crack : 12401769434657527808.0000000000 seconds"
- Button: "Generate Passphrase"

Annotations with arrows point to the following features:

- Two arrows point to the input fields, labeled "fields to input credentials".
- An arrow points to the "Show/Hide Password" button, labeled "Show/Hide password feature".
- An arrow points to the "Password Strength: Strong" feedback, labeled "Strength evaluation feedback".
- An arrow points to the SHA-256 hash and time to crack output, labeled "SHA 256 hash + time to crack estimation based on a set of assumptions".
- An arrow points to the "Check Strength" button, which is part of the "Buttons for" list.
- An arrow points to the "Generate Passphrase" button, which is also part of the "Buttons for" list.