2024

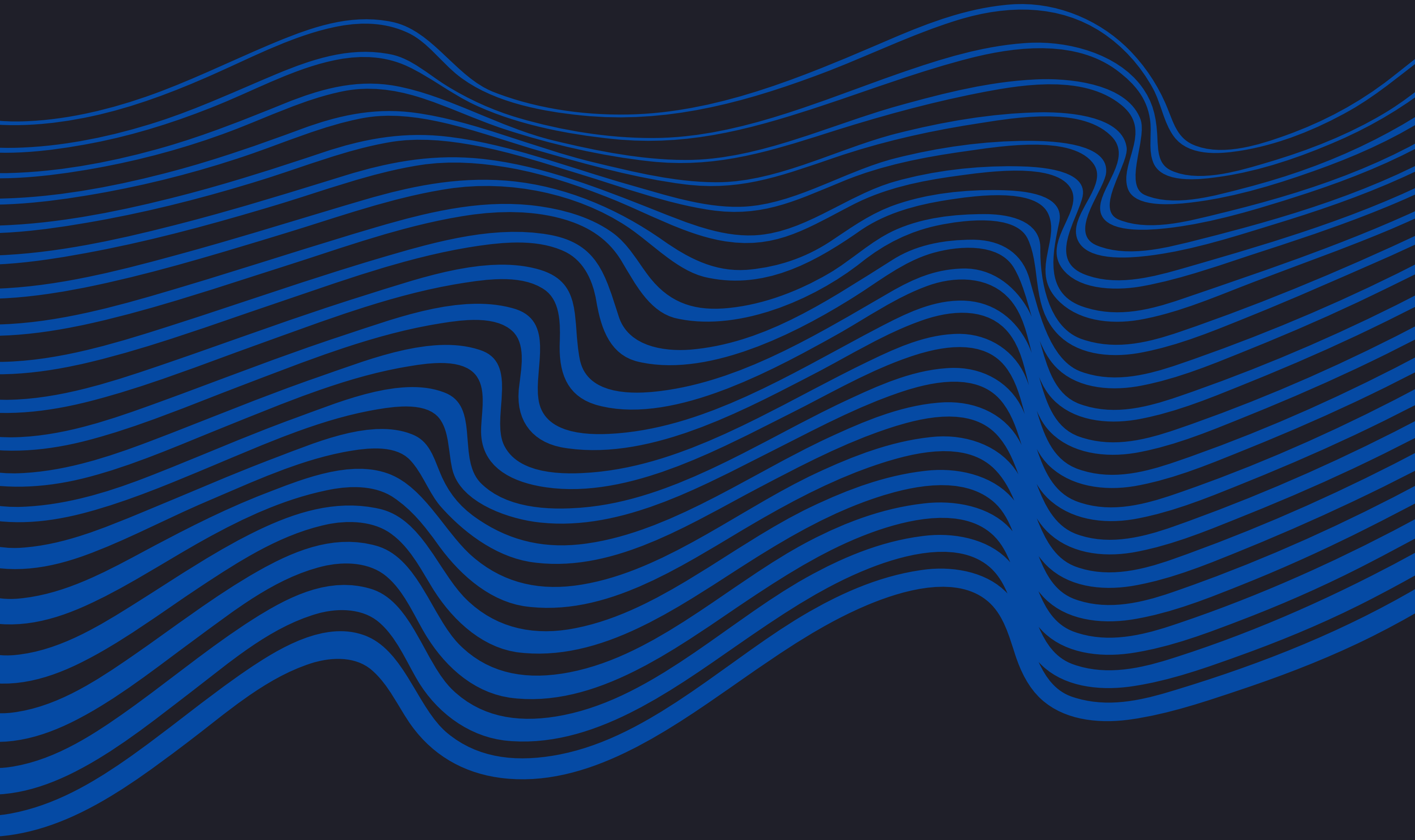
IT PROJECT REPORT

By Yassine Marrekchi

INTRODUCTION

In the realm of cybersecurity, password strength is paramount in safeguarding sensitive information from unauthorized access. To mitigate the risks associated with weak passwords, the concept of password strength evaluation emerges as a pivotal tool. The following project will test the strength of passwords according to the OWASP (Open Web Application Security Project) standards

MAIN COMPONENTS



INPUT INTERFACE

The input interface provides a seamless experience for users to submit their credentials securely. It includes a text field where users can enter their username along with a password. Additionally, there are two buttons available: Check Strength and Generate Passphrase.

The Check Strength button allows users to evaluate the strength of their password based on predefined criteria, ensuring robust security measures. On the other hand, the Generate Passphrase button enables users to generate a passphrase that can be suggested for use as a password. Passphrases are randomly generated combinations of words, offering a high level of security and ease of remembrance.

To enhance privacy and convenience, there's also a Show/Hide Password button, allowing users to toggle the visibility of their password entry. These features collectively enhance the usability and efficiency of the input phase, empowering users to create and manage their credentials effectively.

BACKEND LOGIC

Encompasses the algorithmic process of passwords and their strength evaluation, which means it includes the password strength evaluation algorithm that defines the criteria and rules for assessing password strength.

The Backend Logic validates user input, sanitizes data, and performs calculations to determine the strength level of passwords, along with the passphrase generator code, and the other needed code to structure the input interface.

In this project, we'll employ a passphrase generator :

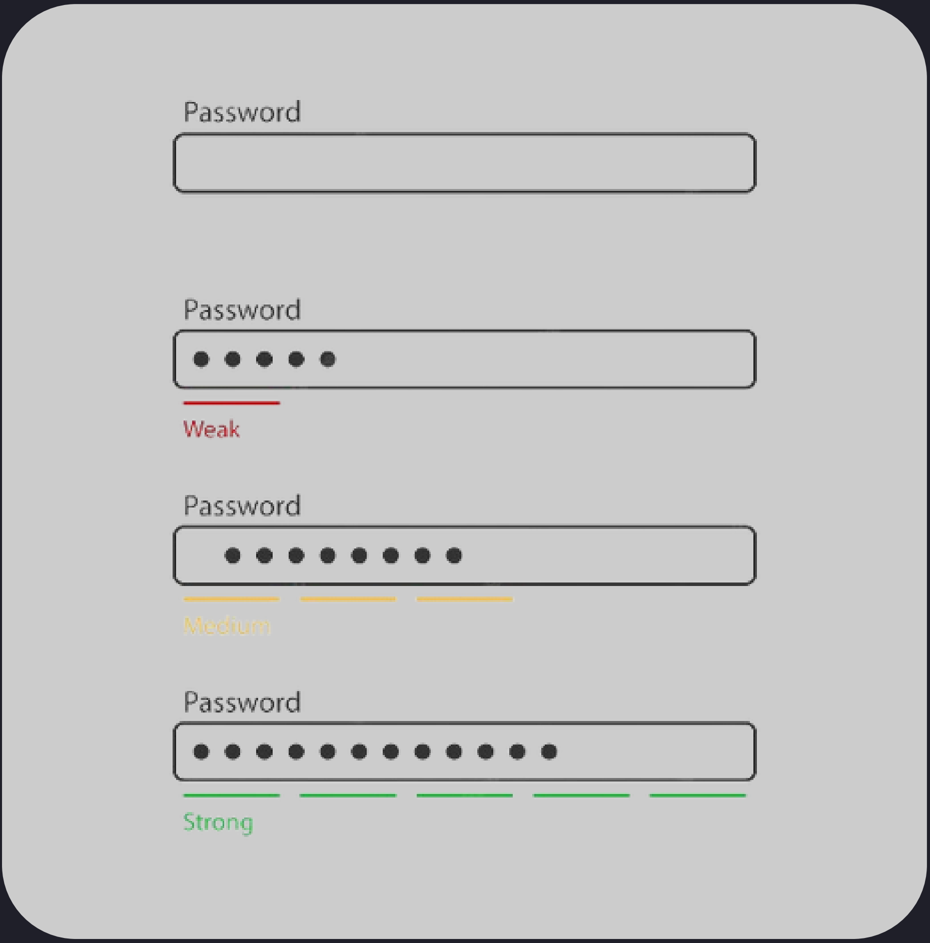
Passphrase generator

The passphrase generator is a tool designed to create strong and random passphrases composed of multiple words. Unlike traditional passwords, which are typically shorter and consist of a mix of characters, passphrases leverage the increased entropy provided by longer strings of words. This makes them more resistant to brute-force attacks while also being easier to remember for users.

this passphrase generator will be used to generate a passphrase made of random words and suggest it for use to the user

Feedback mechanism:

The feedback mechanism communicates the results of the password evaluation to users, It presents feedback in a user-friendly format displayed through descriptive labels (e.g., weak, moderate, strong)



When defining strength criteria for password evaluation, it's essential to consider a balance between security and usability. In this project we'll be using these criteria to assess password strength:

- **Minimum Length :**

Require passwords to have a minimum length, typically between 8 to 12 characters. Longer passwords are generally more secure against brute-force attacks.

- **Complexity Requirements :**
    - Include a mix of character types in passwords, such as uppercase letters, lowercase letters, digits, and special characters (e.g., !, @, #, $).
    - avoiding parts of username or any public information.
    - enforce a rule that checks for sequential characters or keyboard patterns in the password. This helps to prevent users from selecting passwords that follow predictable patterns on a keyboard layout

Weak password

- Does not meet the minimum length requirement (less than 8 characters).
- Lacks diversity in character types (only lowercase letters or only digits or only uppercase).
- Contains sequential characters or keyboard patterns

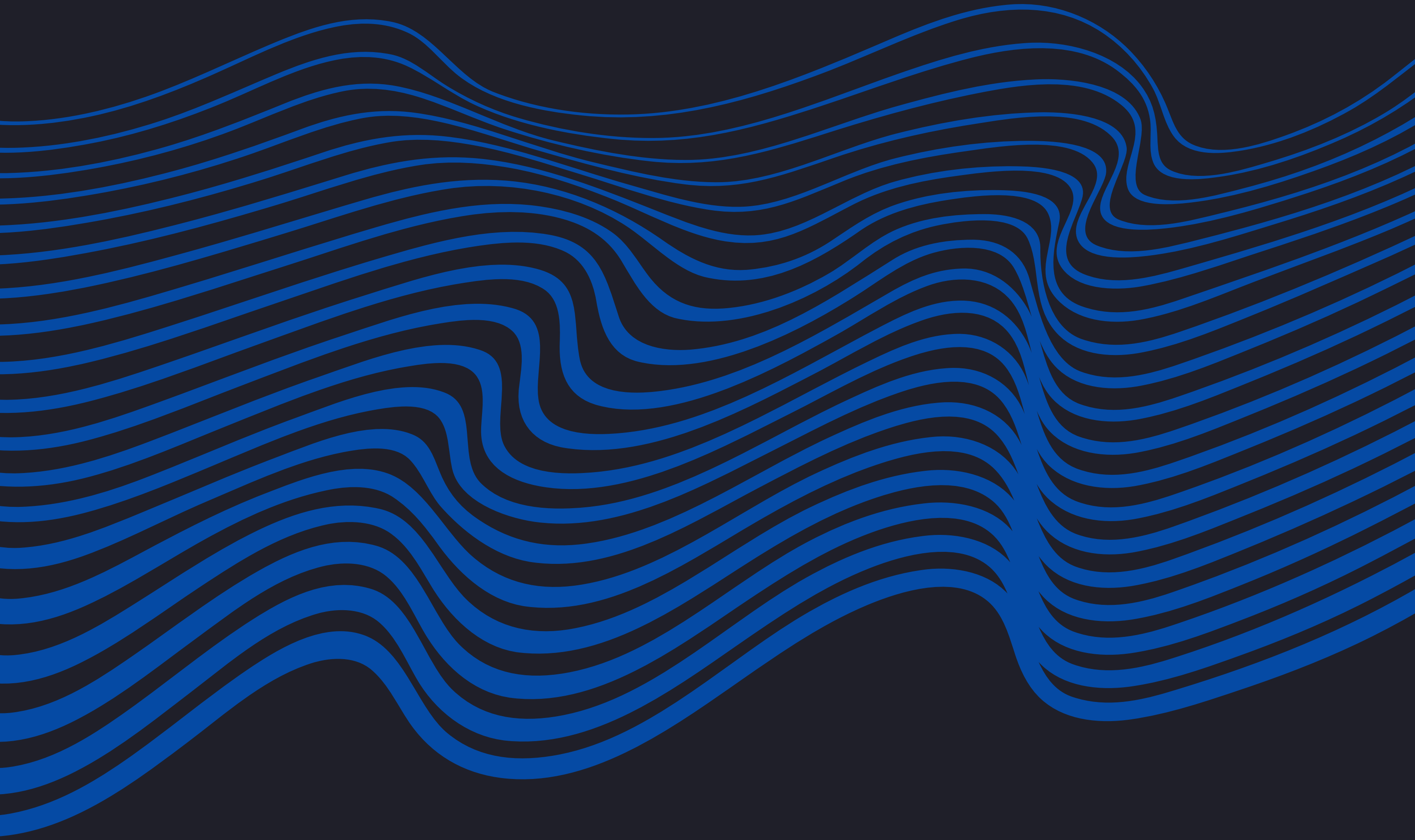
Moderate password

- Meets the minimum length requirement (between 8 to 12 characters).
- Includes a mix of character types (e.g., uppercase letters, lowercase letters, digits, and special characters).
- Does not contain sequential characters or keyboard patterns.
- Avoids using parts of the username or any public information.

Strong Password

- Exceeds the minimum length requirement (more than 12 characters).
- Includes a diverse range of character types, with a good balance of uppercase letters, lowercase letters, digits, and special characters.
- Avoids sequential characters, keyboard patterns, or easily guessable combinations.
- Ensures that no parts of the username or any public information are used.
- Additionally, it may incorporate passphrases or random combinations of words for increased security.

MAIN CHARACTERISTICS



Minimum Length Requirement : The password should be at least eight characters long to ensure a basic level of security.

Complexity : The password should contain a combination of uppercase and lowercase letters, numbers, and special characters to enhance its complexity and resilience preventing brute force attacks.

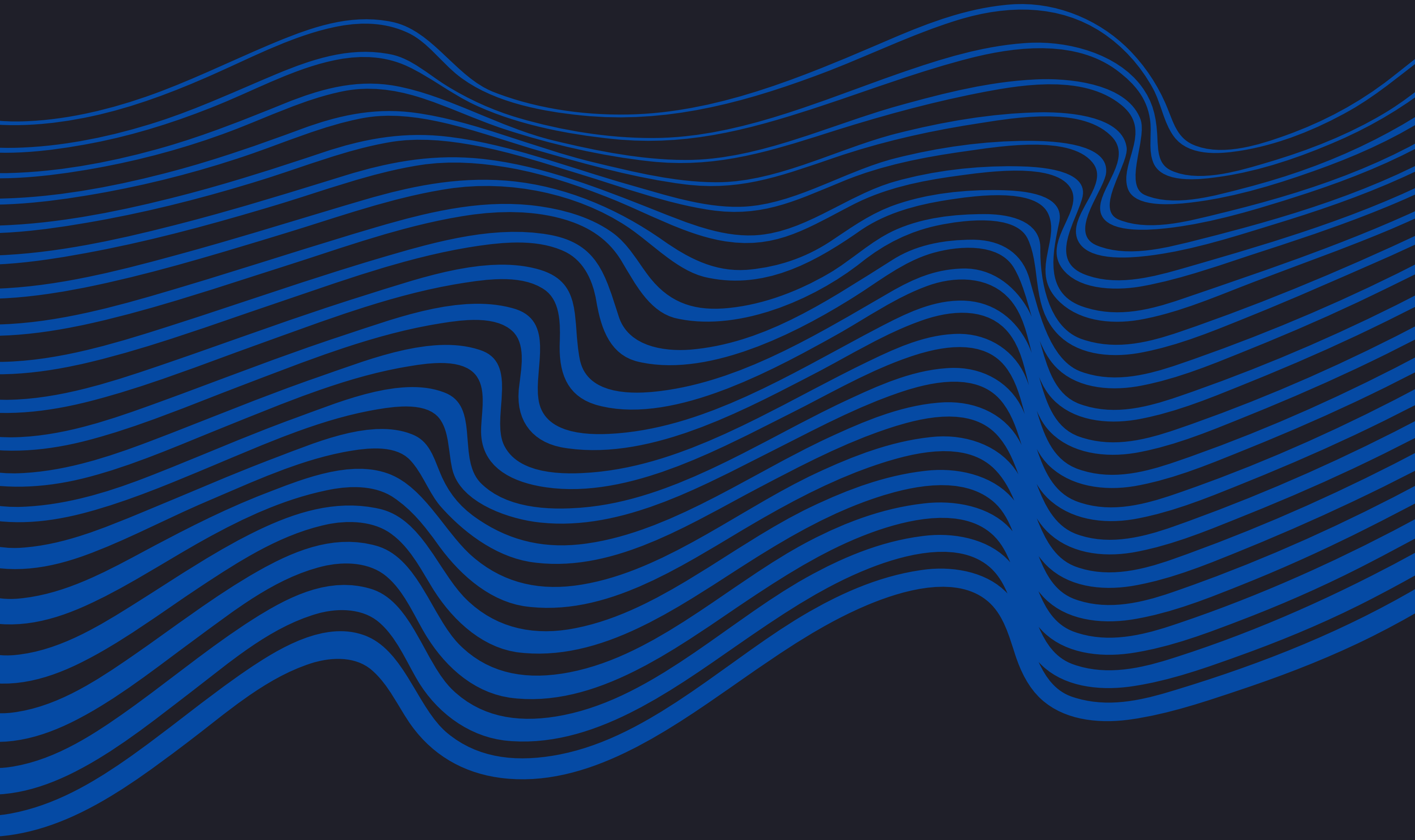
Avoid Accessible Information: To prevent simple guesses or cracking, users are suggested to create passwords that contain neither publicly accessible information nor any elements from their usernames.

Randomness : To deter guessing and cracking efforts, passwords should be random in nature and avoid predictable patterns

Passphrase passwords generator : The tool checks generates passphrases with strong criteria that is hard to crack

Immediate feedback : Users are provided with instant feedback regarding the strength of their passwords, allowing for timely modifications to improve their overall security posture..

ADVANTAGES OF THE EXISTING SOLUTION



Enhanced Password Security

This initiative helps to reduce the risks of identity theft and cyber breaches by promoting the creation of strong and distinctive passwords.

Instructional aspect

The program functions as an instructional platform that imparts best practices for creating secure passwords to users, enabling them to make informed decisions.

Flexibility in Integration

The project makes it easy to create and maintain secure passwords by integrating with password management systems in a smooth manner.

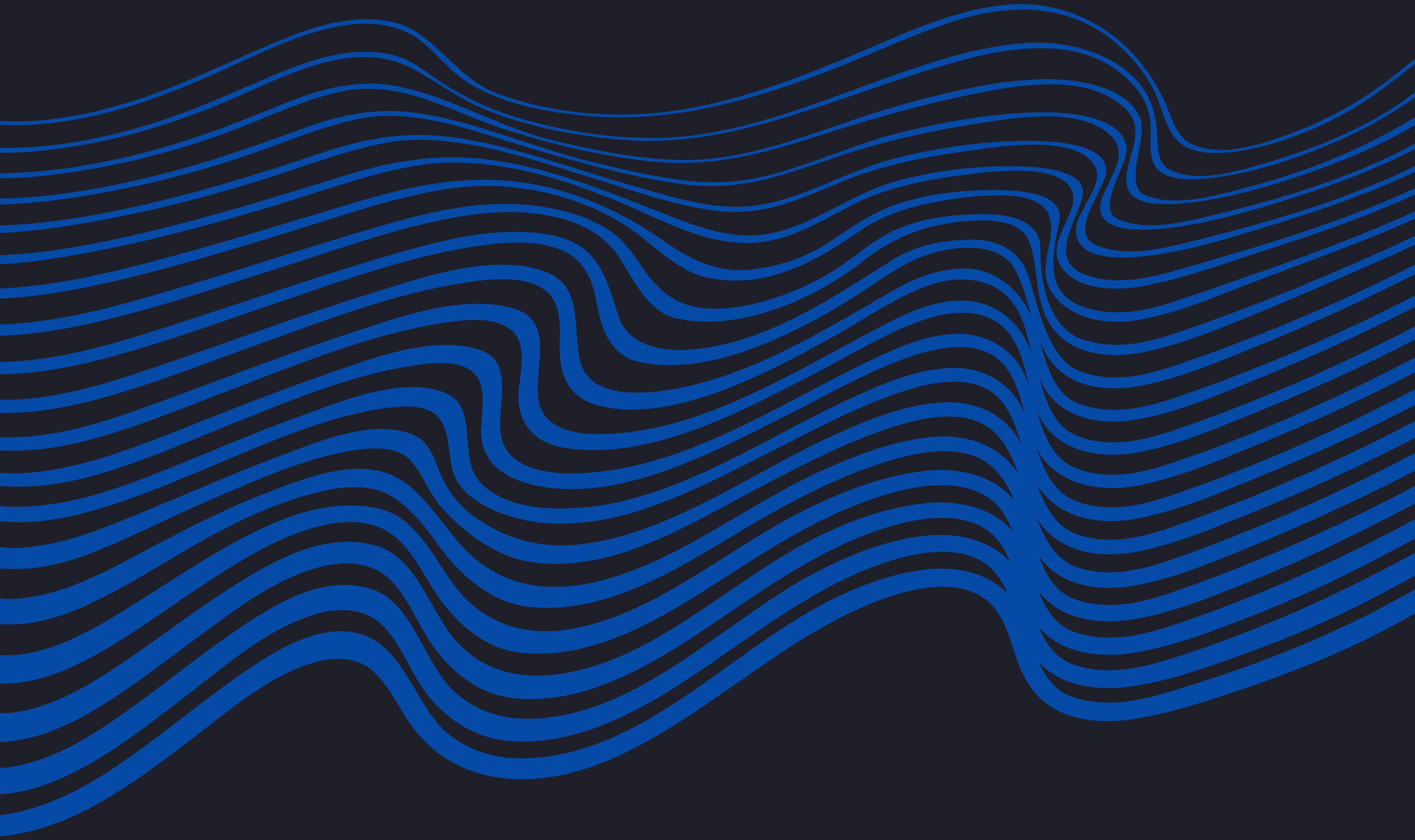
User-Friendly Interface :

The initiative's user-friendly interface ensures accessibility and usability by accommodating users with varying proficiency levels.

Real-Time Threat Intelligence

Some advanced password strength testers leverage real-time threat intelligence to analyze password patterns and identify potential security vulnerabilities. This proactive approach helps organizations stay ahead of emerging threats and mitigate risks effectively.

LIMITS OF THE EXISTING SOLUTION



Language restrictions

A lot of the tools that are now in use are limited to supporting words that are commonly used in English, which makes them less useful for users who speak a different language or dialect.

Authentication gaps

Certain programs have a tendency to ignore passwords that contain personal information, which leads to less rigorous verification procedures.

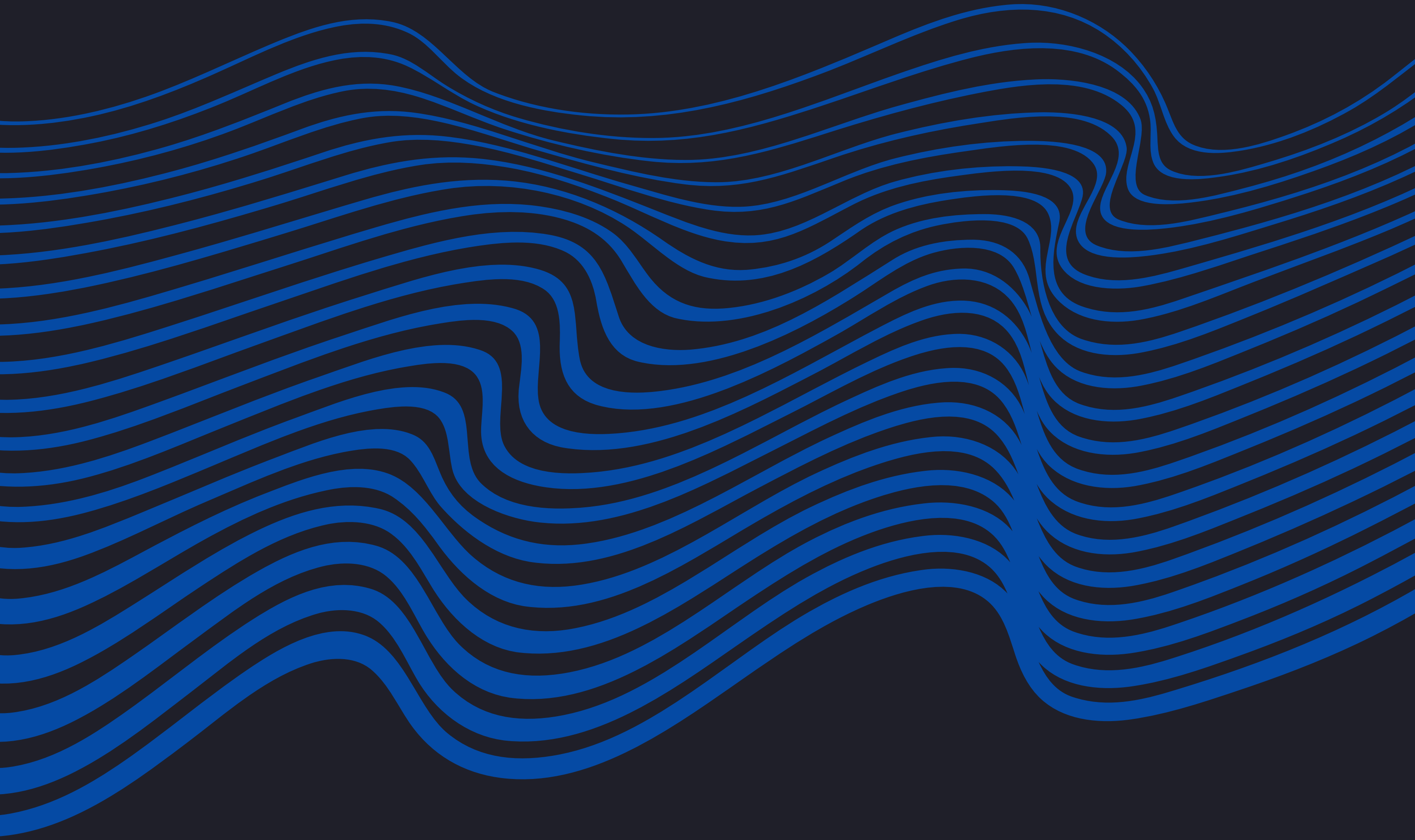
Risks Associated with Confidentiality

User Considerations regarding Reliability: When using old algorithms or ignoring important evaluation criteria, some tools may show errors or discrepancies.

Limitations of criterion

Assessment tools might only pay attention to one criteria, like password length, ignoring other important aspects like vulnerability to brute force invasions and dictionary assaults.

REAL-LIFE EXAMPLES OF THE EXISTING SOLUTIONS :



PassGenius is a password strength checker and generator that helps users create strong and unique passwords. It assesses password strength based on various factors, including length, complexity, and avoidance of common patterns. PassGenius offers personalized feedback and suggestions for optimizing password security.

SecureKey is a browser extension that evaluates the strength of passwords entered on websites in real-time. It analyzes password complexity, uniqueness, and vulnerability to common attacks, such as dictionary attacks and brute-force attacks. SecureKey provides immediate feedback to users and encourages them to choose stronger passwords.

SecurePass is an online password strength checker that evaluates the strength of passwords based on length, complexity, uniqueness, and avoidance of common patterns. It provides instant feedback on password strength and offers suggestions for creating stronger passwords.

DESIGN OF THE PROPOSED SOLUTIONS :

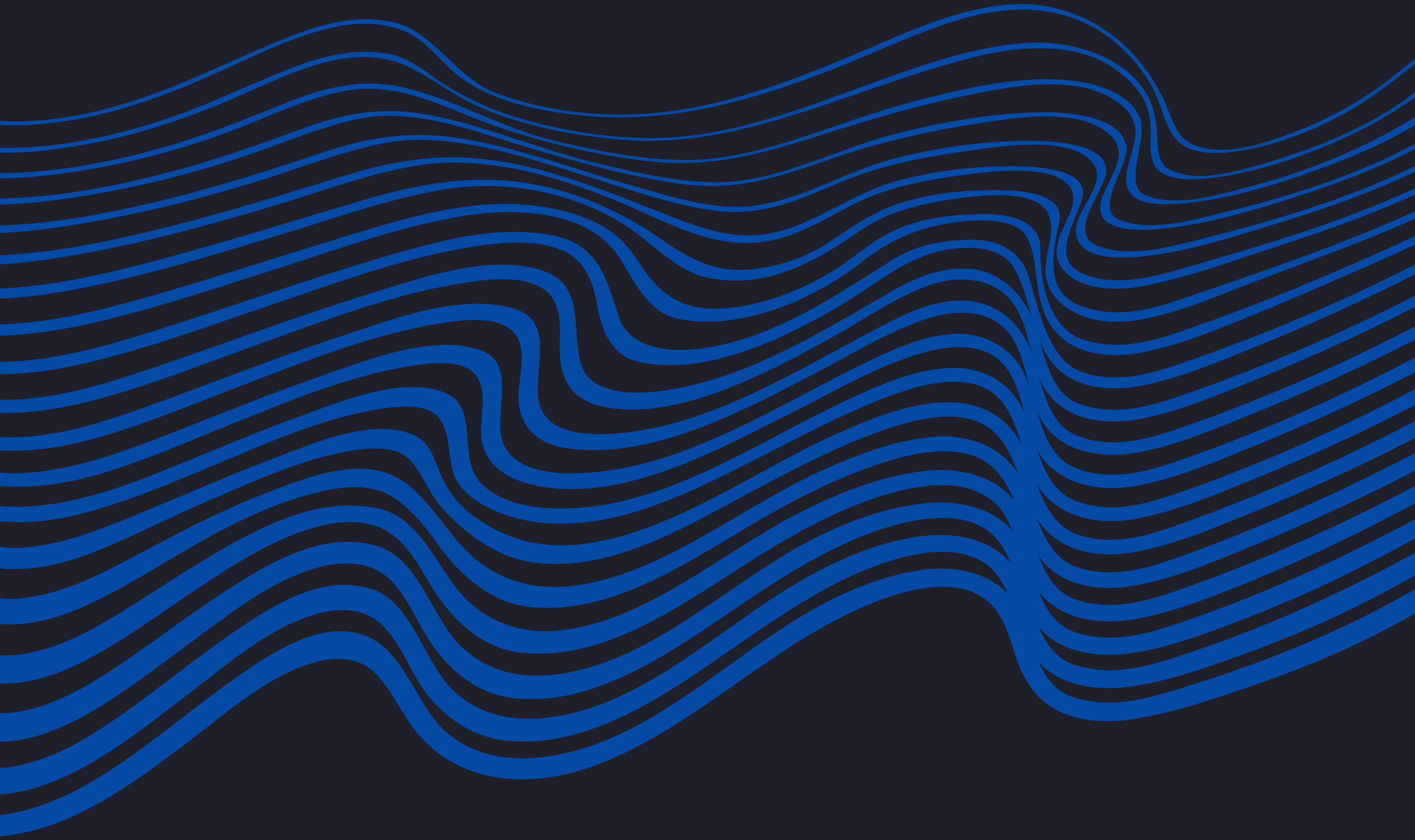
DESIGN OVERVIEW

The PW Evaluator solution is structured around a client-side user interface and JavaScript functionalities for password validation and strength assessment.

FRONT-END INTERFACE

The front-end interface, implemented in HTML, facilitates user interaction by providing input fields for username and password, buttons for checking password strength and generating passphrases, and areas for displaying feedback on password strength, common password warnings, SHA-256 hash, and estimated time to crack the password.

SOLUTION COMPONENTS :



USER INTERFACE

The HTML-based interface serves as the primary interaction point for users, enabling them to seamlessly input passwords and receive real-time feedback on their strength and security.

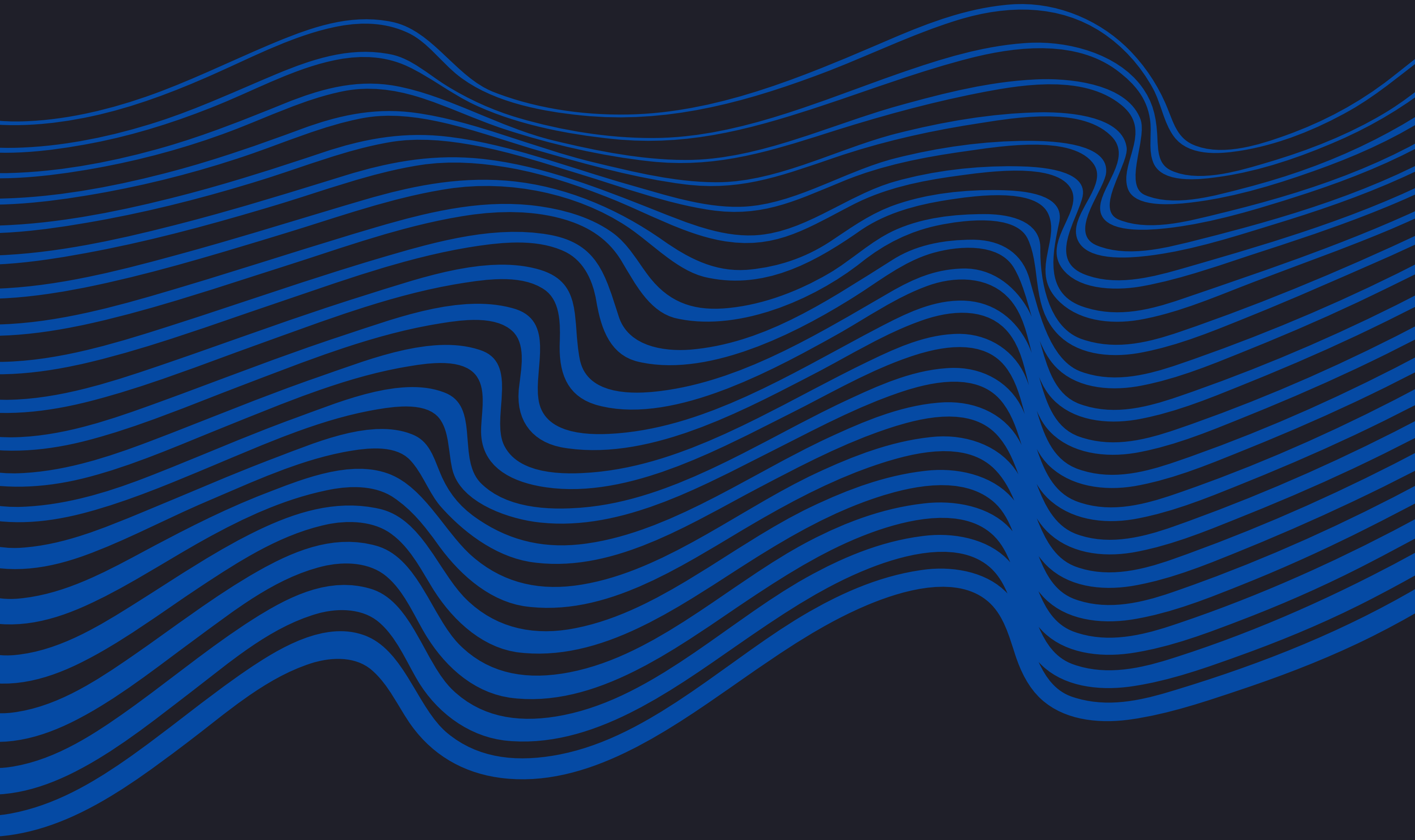
CLIENT-SIDE JAVASCRIPT

Driving the interactivity of the solution, client-side JavaScript functions play a pivotal role in orchestrating password strength validation, passphrase generation, toggling password visibility, and presenting users with crucial insights such as common password warnings, SHA-256 hash values, and estimated time-to-crack metrics.

JSON DATABASE

Integral to the solution is a JSON database containing passwords that are blacklisted due to their commonality. This database is utilized by the client-side JavaScript functions to cross-reference user-entered passwords and provide warnings if they match any entries in the blacklist, thereby enhancing security awareness and promoting the use of unique and robust passwords.

EXCHANGED MESSAGES/DATA OF THE PROPOSED SOLUTION :



Interactions within our Password Strength Checker solution involve smooth communication between its components to ensure effective password validation and strength assessment:

USER INPUT

Users interact with the front-end user interface by entering their desired username and password into the designated input fields.

CLIENT-SIDE EVALUATION

Upon inputting the required data , client-side JavaScript functions execute internal validation checks to ensure that the password meets predefined criteria for strength. These criteria typically include length, character variety, and avoidance of common patterns.

COMMON PASSWORD CHECK

The client-side script fetches a JSON database of commonly used passwords to check if the user's password matches any entries. This step helps identify passwords that are frequently used and potentially weak.

SHA-256 HASH CALCULATION

Once the password passes internal validation, the script calculates its SHA-256 hash using browser-based cryptography APIs. This hash serves as a unique representation of the password and is used for further analysis.

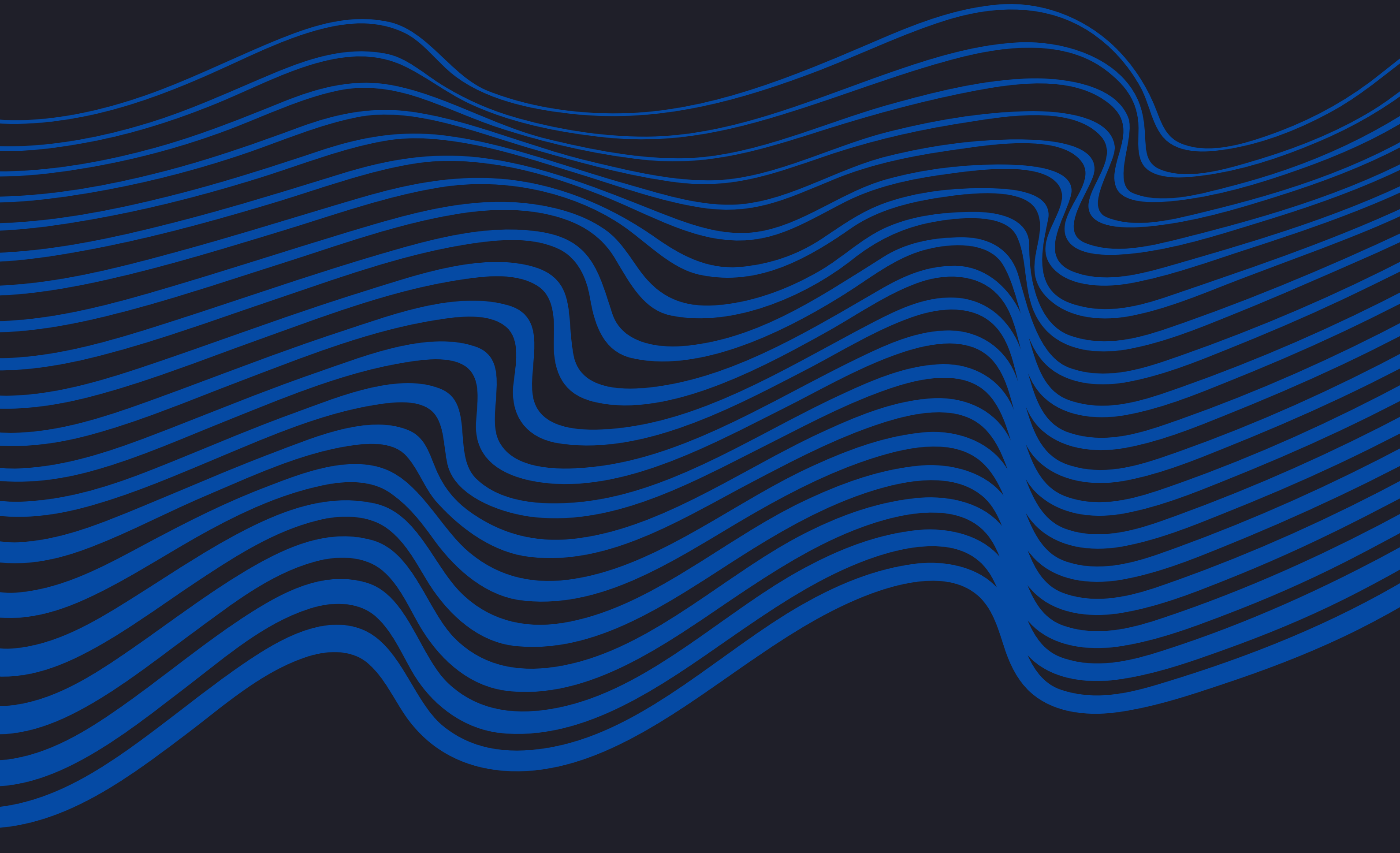
ESTIMATION OF CRACK TIME

Using the SHA-256 hash, the script estimates the time required for an attacker to crack the password based on the complexity of its characters and length. This estimation provides users with insights into the strength of their chosen password.

FEEDBACK DISPLAY

Upon inputting the required data , client-side JavaScript functions execute internal validation checks to ensure that the password meets predefined criteria for strength. These criteria typically include length, character variety, and avoidance of common patterns.

FUNCTIONAL REQUIREMENTS :



USER INPUT ACCEPTANCE

The system shall accept the following user inputs:

- Username
- Password

PASSWORD DICTIONARY CHECK

- The system shall verify whether the entered password exists in the provided JSON database of common passwords.
- If a match is found, the system shall display a warning stating the commonality of password and lack of security

OWASP STANDARDS COMPLIANCE CHECK

- The system shall ensure that the password meets the OWASP standards by checking that:
- It is at least 8 characters long.
- It contains a combination of numbers, uppercase letters, lowercase letters, and special characters.
- It does not contain patterns of the username.

PASSPHRASE GENERATION

- The system shall include a passphrase generator feature.
- Users shall have the option to generate a secure passphrase meeting predefined criteria, such as minimum length and character diversity.
- The generated passphrase shall be displayed to the user and can be used as an alternative to manually entering a password.
- This feature enhances user convenience and promotes the creation of strong, memorable passwords.

PASSWORD ENCRYPTION

- The system shall output the SHA-256 encrypted version of the user's password.
- The SHA-256 encrypted version of the password shall be displayed .
- This ensures a better judgment of the password’s performance in terms of security

ESTIMATED CRACK TIME CALCULATION

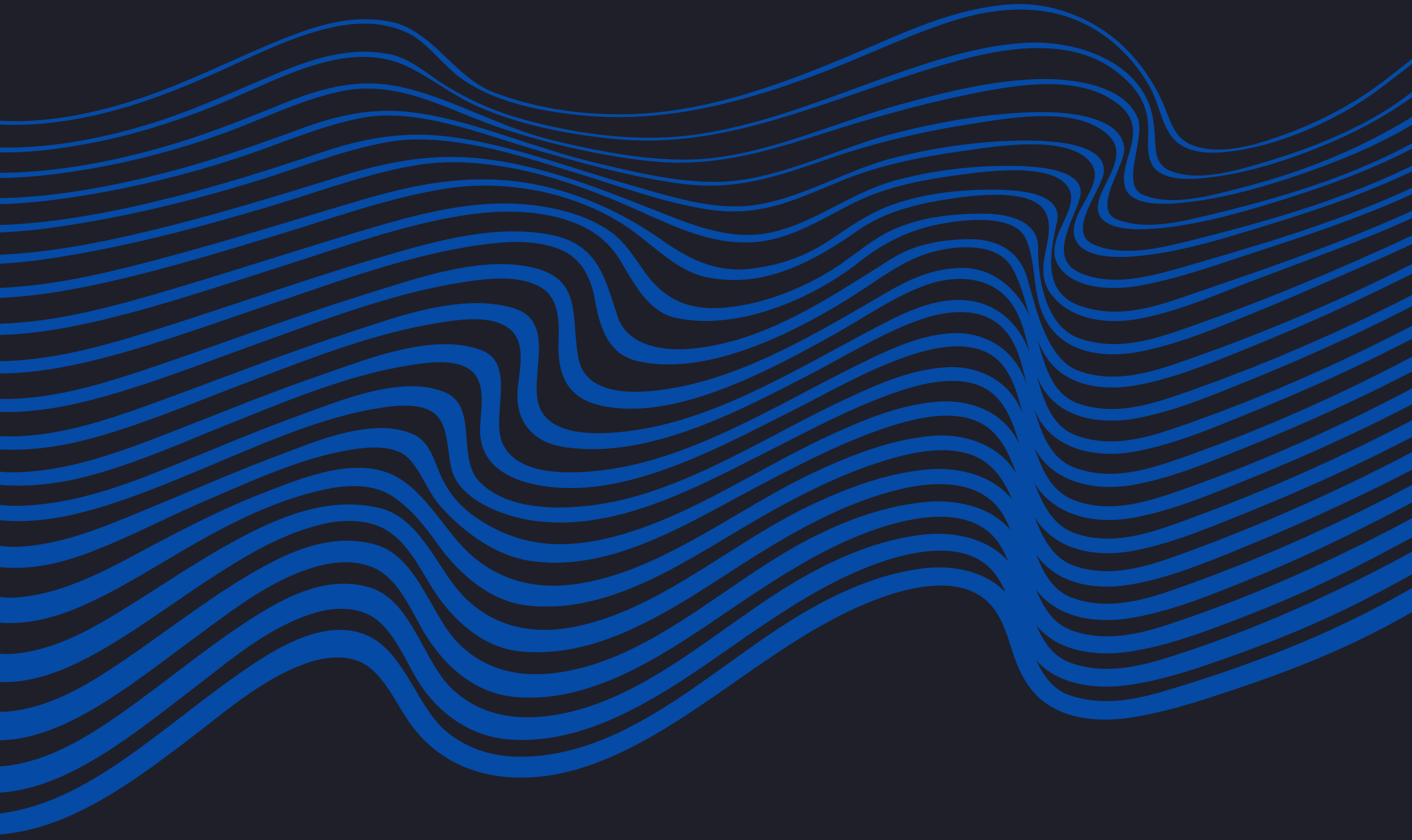
Using the selected algorithm (SHA-256), the system shall calculate an estimated time required to crack the password.

FEEDBACK ON PASSWORD STRENGTH

The system shall provide feedback to the user regarding the strength of their password, indicating whether it is:

- Weak
- Moderate
- Strong

NON-FUNCTIONAL REQUIREMENTS :



PERFORMANCE

- The system shall provide quick response times for password strength checks and SHA-256 encryption, ensuring a smooth user experience.
- Password strength checks and encryption processes shall be completed within a reasonable timeframe, even under peak load conditions.

SECURITY

The system shall ensure secure constraints of strength evaluating passwords

RELIABILITY

- The system shall be available and accessible to users at all times, with minimal downtime for maintenance or updates.
- Password strength checks and encryption processes shall be reliable and accurate, providing consistent results for all users.

SCALABILITY

- The system shall be capable of handling a large number of concurrent users and password strength checks, scaling resources dynamically as needed.
- As user demand increases, the system shall seamlessly expand its capacity to maintain performance and responsiveness.

USABILITY

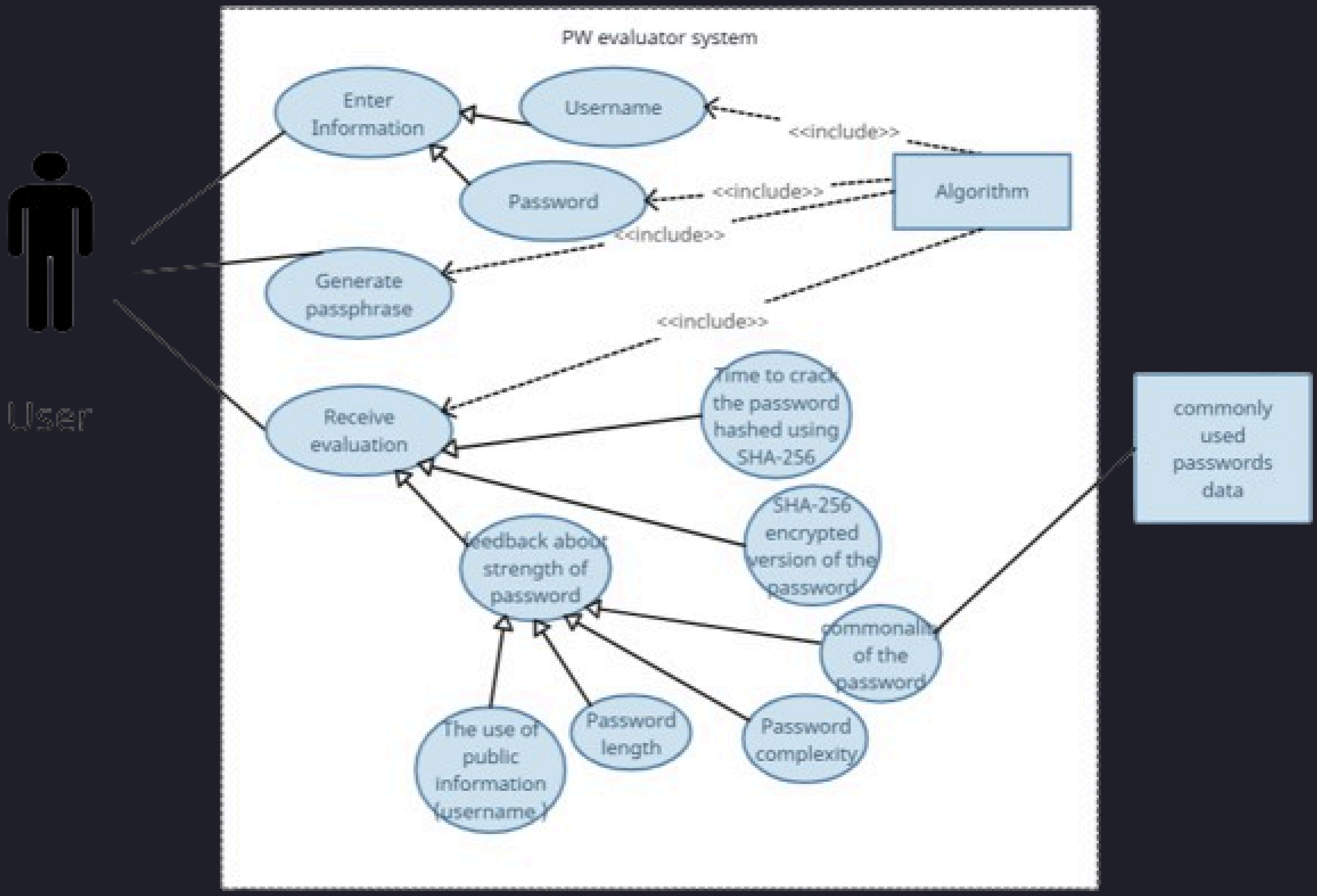
- The user interface shall be intuitive and user-friendly, guiding users through the password strength checking process with clear instructions and feedback.
- Password strength feedback and SHA-256 encrypted versions shall be presented in a readable format, ensuring easy comprehension for users.

COMPATIBILITY

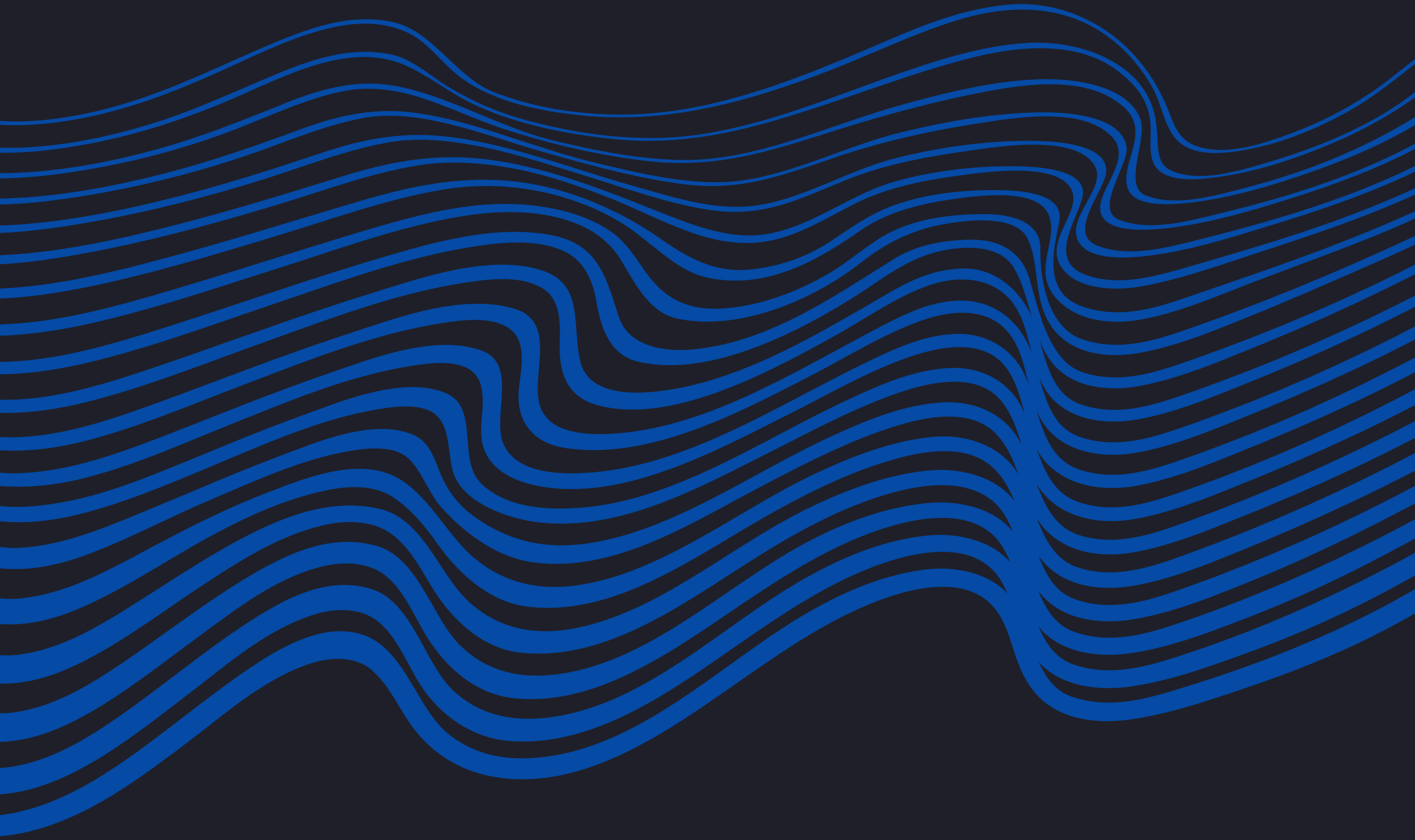
Compatibility shall be maintained across different operating systems and screen sizes to accommodate diverse user preferences.

ROLES OF THE SOLUTION :

USE CASE DIAGRAM


PW evaluator system
Enter Information
Username
<<include>>
Password
<<include>>
Algorithm
<<include>>
Generate passphrase
<<include>>
User
Receive evaluation
Time to crack the password hashed using SHA-256
SHA-256 encrypted version of the password
feedback about strength of password
commonality of the password
commonly used passwords data
The use of public information (username)
Password length
Password complexity

DEVELOPMENT PHASES :



REQUIREMENT GATHERING AND ANALYSIS

Collect and analyze the project requirements, including both functional and non-functional requirements. This involves understanding the needs of stakeholders and documenting the specific features and capabilities the software should have.

SYSTEM DESIGN

Design the overall system architecture, including the back-end, database, and front-end components. Define data models, decide on technologies and frameworks to be used, and create mockups or wireframes for the user interface.

BACK-END DEVELOPMENT

Develop the back-end application logic, which includes the password strength evaluation algorithm that defines the criteria and rules for assessing password strength

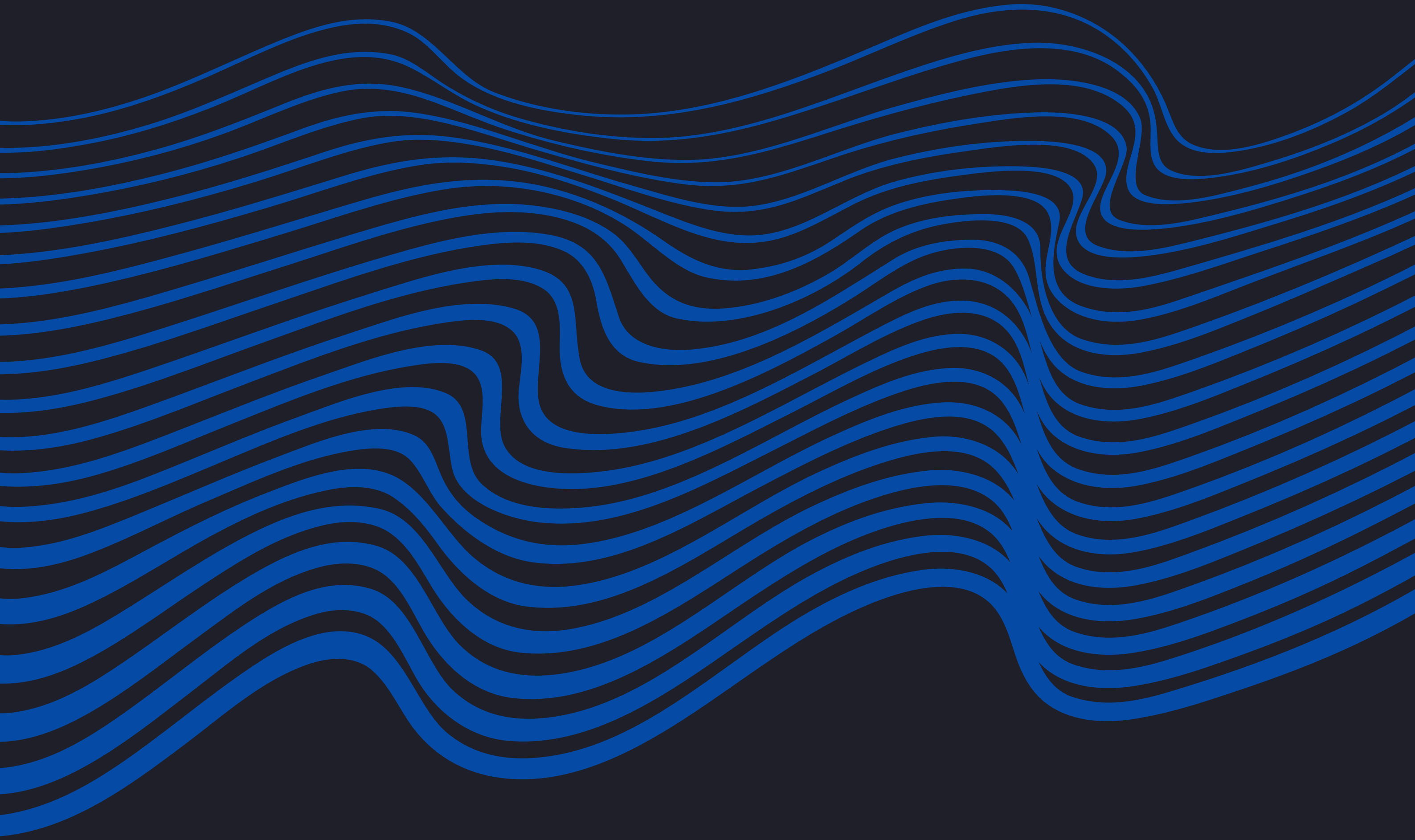
FRONT-END DEVELOPMENT

Develop the user interface for the web application using HTML, CSS, and JavaScript. Create interactive and visually appealing user interfaces that facilitates user interaction by providing input fields for username and password, buttons for checking password strength and generating passphrases, and areas for displaying feedback on password strength, common password warnings, SHA-256 hash, and estimated time to crack the password.

INTEGRATION AND TESTING

Integrate the back-end, front-end, and JSON database components to ensure they work together seamlessly. Perform thorough testing

TOOLS :



USED TOOLS FOR EVALUATOR DEVELOPMENT :

- Programming Languages : HTML , CSS , JS
- Imported JSON database : Includes the commonly used and easy to brute force passwords to be used for strength evaluation
- Adobe illustrator for logo design