Test Plan

Introduction

Purpose:

* Define the testing strategy and ensure the implementation meets the business requirements.

Scope:

* Cover functional and non-functional requirements, including unit tests, integration tests, performance tests, and usability tests.

Test Strategy

Unit Testing:

* Test individual components and functions.

Integration Testing:

* Ensure different components work together as expected.

Performance Testing:

* Measure the efficiency and resource usage of the system.

Usability Testing:

* Verify the user interface is intuitive and user-friendly.

Test Cases

1. User Input Tests

**Test Case 1.1**

Description:

* Validate User Input for Password Field

Objective:

* Ensure that the password input field accepts valid input and rejects invalid input.

Steps:

* Enter a valid password (e.g., "password123").
* Submit the form.
* Enter an invalid password (e.g., empty string, special characters only).
* Submit the form.

Expected Results:

* The form accepts and processes the valid password.
* The form rejects the invalid password and displays an appropriate error message.

**Test Case 1.2:**

Description:

* Validate User Input for Hash Field

Objective:

* Ensure that the hash input field accepts valid input and rejects invalid input.

Steps:

* Enter a valid hash (e.g., a SHA-256 hash).
* Submit the form.
* Enter an invalid hash (e.g., incorrect length, non-hexadecimal characters).
* Submit the form.

Expected Results:

* The form accepts and processes the valid hash.
* The form rejects the invalid hash and displays an appropriate error message.

2. Password Cracking Tests

**Test Case 2.1**

Description:

* Validate Brute Force Algorithm

Objective:

* Ensure that the brute force algorithm correctly generates and tests password combinations.

Steps:

* Provide a known password and its corresponding hash.
* Run the password cracking process.

Expected Results:

* The system successfully cracks the password and outputs the correct result.

**Test Case 2.2**

Description:

* Validate Parallel Processing

Objective:

* Ensure that the parallel processing implementation correctly divides the task among multiple threads.

Steps:

* Configure the system to use multiple threads.
* Monitor the execution and resource utilization.

Expected Results:

* The system efficiently utilizes multiple threads to speed up the password cracking process.

3. Output Tests

**Test Case 3.1**

Description:

* Display Cracked Password

Objective:

* Ensure the system correctly displays the cracked password if found.

Steps:

* Provide a known password and its corresponding hash.
* Run the password cracking process.

Expected Results:

* The system outputs the correct cracked password.

**Test Case 3.2**

Description:

* Display Time Taken

Objective:

* Ensure the system correctly displays the time taken to crack the password.

Steps:

* Provide a known password and its corresponding hash.
* Run the password cracking process.

Expected Results:

* The system displays the correct time taken for the process.

**Test Case 3.3**

Description:

* Display Message if Password Not Found

Objective:

* Ensure the system displays an appropriate message if the password is not found within a reasonable time.

Steps:

* Provide an unrealistic or complex hash that cannot be cracked in a short time.
* Run the password cracking process.

Expected Results:

* The system displays a message indicating that the password was not found.

4. Performance Tests

**Test Case 4.1**

Description:

* Test System Performance with Large Input Space

Objective:

* Measure the system's performance when handling a large input space.

Steps:

* Provide a large character set and a complex hash.
* Run the password cracking process.

Expected Results:

* The system maintains acceptable performance and resource usage.

**Test Case 4.2**

Description:

* Measure Resource Usage and Execution Time

Objective:

* Monitor the system's resource usage and execution time for various input sizes.

Steps:

* Provide different character sets and hashes.
* Run the password cracking process for each set.

Expected Results:

* The system provides resource usage and execution time within acceptable limits.

Test Environment

Hardware:

* Personal computer | specifications: Ryzen 9 CPU, 64GB RAM.

Software:

* Windows OS, .net 8, Visual Studio 2022.

Tools: todo: list any tools or frameworks used for testing (e.g., NUnit, MSTest).

Acceptance Criteria

Functionality:

* All functional test cases pass, ensuring the system performs as expected.

Performance:

* The system meets performance benchmarks and handles large input spaces efficiently.

Usability:

* The user interface is intuitive and user-friendly, passing all usability tests.

Traceability Matrix:

Business Requirement Test Case ID Test Case Description

User Input Requirements

Users can input a password or hash to be cracked TC-1.1 Validate User Input for Password Field

TC-1.2 Validate User Input for Hash Field

Users can specify the character set to use for brute force TC-1.3 Validate Character Set Selection

Users can set the number of threads TC-1.4 Validate Thread Count Setting

Password Cracking Requirements

Implement brute force algorithm using efficient data structures TC-2.1 Validate Brute Force Algorithm

Support parallel processing TC-2.2 Validate Parallel Processing

Provide real-time feedback on progress TC-2.3 Validate Real-Time Feedback

Output Requirements

Display the cracked password if found TC-3.1 Display Cracked Password

Show time taken to crack the password TC-3.2 Display Time Taken

Display a message if password is not found TC-3.3 Display Message if Password Not Found

Performance Requirements

Handle large input spaces efficiently TC-4.1 Test System Performance with Large Input Space

Efficient resource usage TC-4.2 Measure Resource Usage and Execution Time

Usability Requirements

Intuitive and user-friendly interface TC-5.1 Validate User Interface Usability

Clear instructions and documentation TC-5.2 Validate Documentation and Instructions