**Report file**

**---------------------------------------TASK 6 ---------------------------------------------------**

**\* Objective**

Understand what makes a password strong, test different passwords using an online password strength checker, and analyze results to recommend best practices.

**\* Tools Used**

* Password Strength Checker: passwordmeter.com Password.Kaspersky.com & security.org Password Checker
* Estimation Assumption: Crack time calculated at 1 billion guesses/second (common high-end GPU cluster rate).
* 3.Test Scope: Example/demo passwords only; real personal passwords not tested for security reasons.

**\* Methodology**

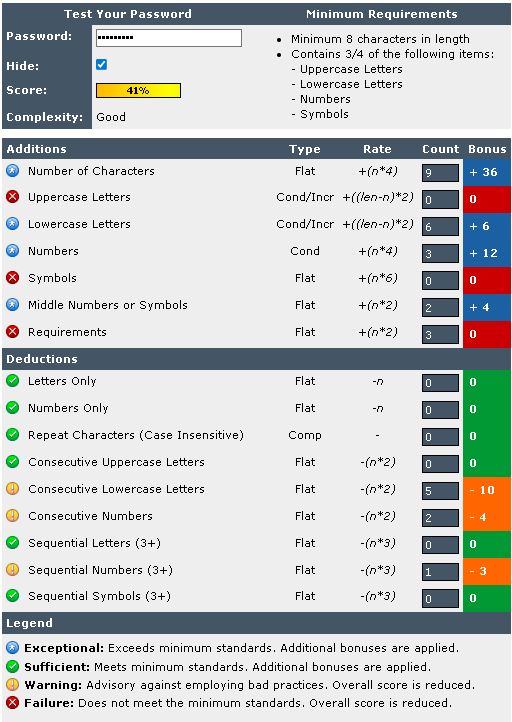
* Created 6 example passwords of varying complexity (very weak → very strong).
* Tested each password on password strength checker websites.
* Recorded tool scores, estimated crack times, and feedback.
* Compared brute-force time vs. dictionary attack feasibility.
* Summarized best practices for creating strong passwords.

**\*Steps for Password creating and checking on different online platforms.**

* Take a Random phase like ….. family , love ,life ,happy , together forever .
* Make random passwords for testing by using these five phases .
* Passwords like ……
* **family@123,** **Family2028!,** **love,** **life@in,** **happy@e,** **Tog3ther@forever#.**
* Then , test these strongness through the given websites .
* Now observe the results and check the fault that what are loopholes for making them strong.

**\*TEST RESULTS screen Shots (passwordmeter.com )**

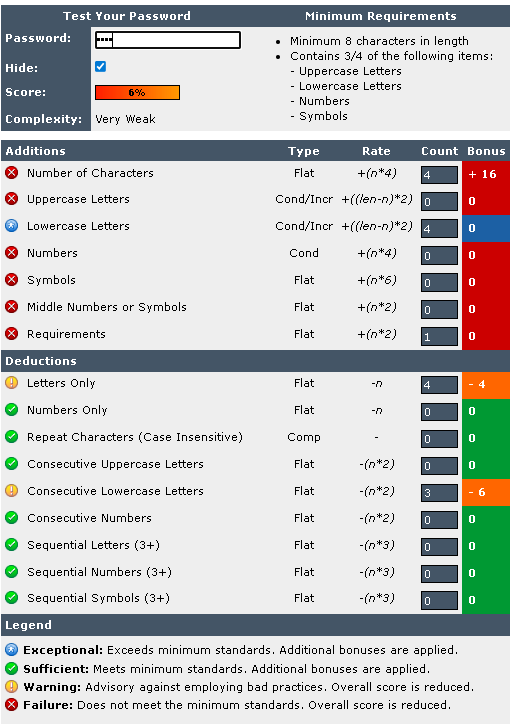
**Family123**



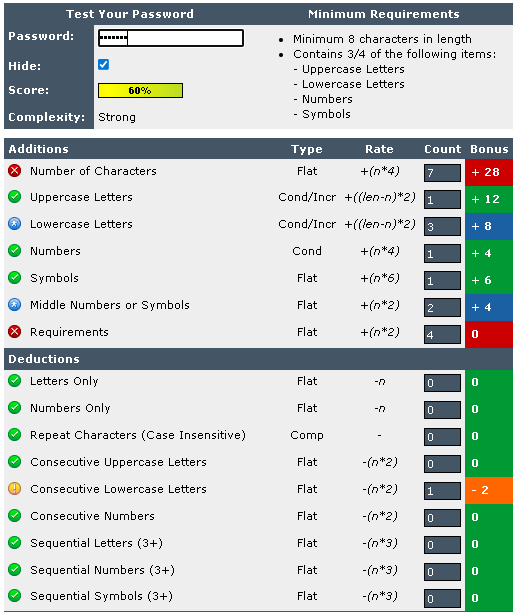
**Family2028!**



**Love**



**L!fe3\_in**

****

**happy@e**

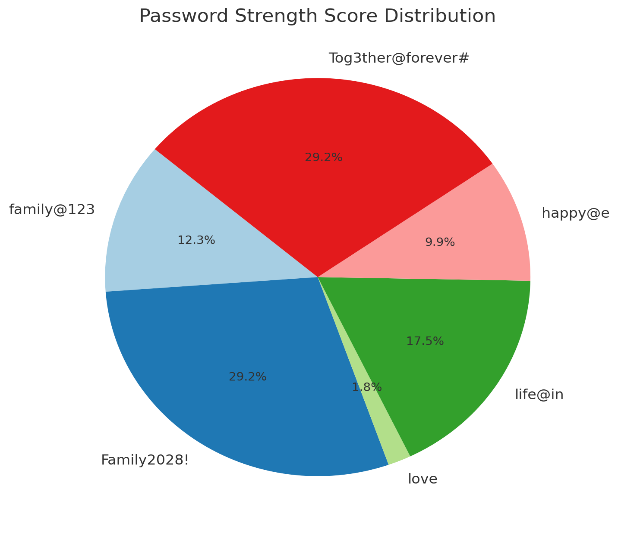


**Together@Forever#**



**\* Test Results**

| **S. No.** | **Password (Example)** | **Length** | **Char Types Used** | **Strength Score** | **Est. Crack Time (Brute Force)** | **Dictionary Attack Risk** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | family@123 | 10 | Lowercase + Symbol + Digits | 42% (Good) | ~42 mins | Very High |
| 2 | Family2028! | 11 | Uppercase + Lowercase + Digits + Sym | 100% (Very Strong) | Centuries | Medium-High |
| 3 | love | 4 | Lowercase only | 6% (Very Weak) | Instant (<1 sec) | Very High |
| 4 | life@in | 7 | Lowercase + Symbol | 60% (Strong) | Few Hours | High |
| 5 | happy@e | 7 | Lowercase + Symbol | 34% (Weak) | Few Hours | High |
| 6 | Tog3ther@forever# | 18 | Uppercase + Lowercase + Digits + Sym | 100% (Very Strong) | Trillions of years | Low |



**1. family@123 (42 min to crack as per your note)**

* Attack type: Dictionary + common substitutions
* Reason: "family" ek common dictionary word hai, "@" ek predictable substitution hai, aur "123" pattern bahut common hai.
* Brute force: ~ minutes se hours (high-speed cracking hardware pe ~42 mins)
* Dictionary attack: Seconds me crack ho sakta hai kyunki ye common pattern wordlist me hota hai.

**2. Family2028! (100%, very strong)**

* Length: 11 characters
* Complexity: Uppercase + lowercase + digits + symbol
* Brute force: ~ centuries lag sakte hain agar attacker random guess kare, lekin dictionary + year pattern detect hone par kuch ghanto me crack ho sakta hai.
* Dictionary risk: "Family" ek dictionary word hai, "2028!" ek common year+symbol pattern hai — targeted attack me speed faster ho sakti hai.

**3. love (very weak)**

* Attack type: Pure dictionary
* Brute force: Seconds
* Dictionary attack: Instant crack (bahut common word hai)

**4. life@in (60%, strong)**

* Length: 7 characters
* Complexity: lowercase + symbol
* Brute force: ~ few hours (high-speed setup)
* Dictionary attack: Agar attacker ke paas dictionary me "life" aur "in" separate words ho, to easily combine karke crack ho sakta hai.

**5. happy@e (34%, weak)**

* Length: 7 characters
* Complexity: lowercase + symbol
* Brute force: ~ hours
* Dictionary attack: Common word + predictable symbol → fast crack

**6. Tog3ther@forever# (100%, very strong)**

* Length: 18 characters
* Complexity: Uppercase + lowercase + digits + symbols + length > 15
* Brute force: Trillions of years (random guessing)
* Dictionary attack: If attacker tries word combos (“together forever”), they can find, but mixed casing + numbers + symbols make it extremely hard without targeted personal info.

**\* Brute Force vs Dictionary Attack**

**1. Brute Force Attack:**

* Attacker tries every possible combination of characters until the password is guessed. The more characters & types used, the longer it takes.
* Weak passwords (short length, simple words) → seconds to minutes
* Strong passwords (long, mixed chars) → centuries to trillions of years

**2. Dictionary Attack:**

* Attacker uses a precompiled list of common words, names, and patterns.
* Passwords containing dictionary words like love, family, happy can be cracked in seconds
* Even strong-looking passwords with common words can be guessed faster than pure random strings.

**\* Best Practices for Strong Passwords**

* 1.Use minimum 12–16 characters, preferably more.
* 2.Combine uppercase, lowercase, numbers, and symbols.
* 3.Avoid dictionary words or personal info (name, DOB, etc.).
* 4.Use a passphrase: Combine random, unrelated words with symbols and numbers.
* 5.Never reuse passwords across accounts.
* 6.Store passwords in a trusted password manager (Bitwarden, KeePass, etc.).
* 7.Enable Multi-Factor Authentication (MFA) wherever possible.

**\* Conclusion**

* The test clearly shows that:
* 1.Short & common passwords are extremely easy to crack.
* 2.Even long passwords can be weak if they use predictable patterns or dictionary words.
* 3.The most secure passwords are long, random, and unpredictable, making both brute-force and dictionary attacks impractical.