**PASSWORD GENERATOR AND**

**STRENGTH CHECKER**

**Created by:**

Yashika

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Topic** | **Page No.** |
| 1. | Introduction | 3 |
| 2. | Problem Statement | 4 |
| 3. | UML Class Diagram | 5 |
| 4. | Block Diagram | 6 |
| 5. | Output Screenshots | 7-12 |
| 6. | Advantages | 13 |
| 7. | Limitation of Project | 14 |
| 8. | Future Scopes | 15 |
| 9. | Conclusion | 16 |
| 10. | References | 17 |

**INTRODUCTION**

A password generator is a program that creates random passwords for users. These passwords can be generated using various methods such as using Java’s built-in Random class or using the Apache Commons Lang library. The generated passwords can be of a specified length and can include a combination of alphabets, numbers, and special characters.

A password strength checker is another program that checks the strength of a user’s input password. It can check for various criteria such as the length of the password, the use of special characters, numbers, and upper- and lower-case letters. A strong password is usually at least 8 characters long and includes a combination of these different types of characters.

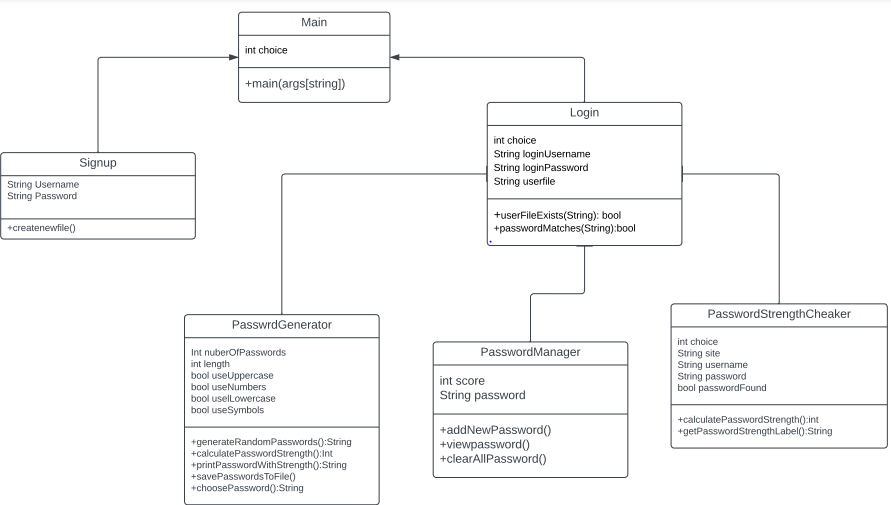
A password manager is a software application designed to securely store and manage passwords for various online accounts. The primary purpose of a password manager is to alleviate the burden of remembering multiple complex passwords for various online accounts. Instead of relying on easily guessable or reused passwords, users can generate strong, unique passwords for each account and store them in the password manager. The password manager then securely stores the passwords and automatically fills them in when needed.

These programs can be useful in helping user create strong and secure password to protect their accounts and personal information. By using a password generator and strength checker users can ensure that their passwords are difficult to guess or crack by hackers.

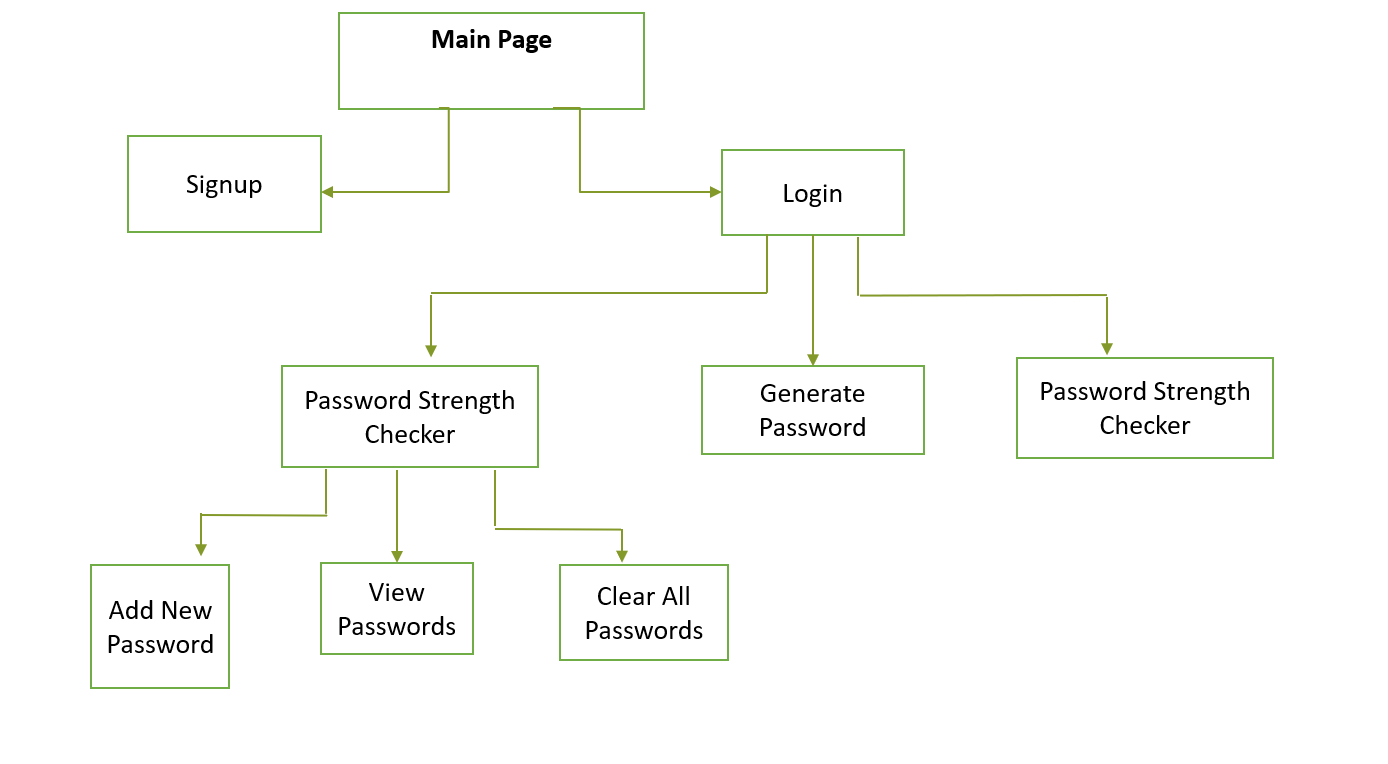
**PROBLEM STATEMENT**

* Create a Java program that generates random passwords for users and checks the strength of user input passwords.
* The password generator should allow users to specify the length of the generated password and should include a combination of alphabets, numbers, and special characters.
* The password strength checker should check for various criteria such as the length of the password, the use of special characters, numbers, and upper- and lower-case letters. The program should provide feedback to the user on the strength of their input password and suggest ways to improve it.

**UML CLASS DIAGRAM**

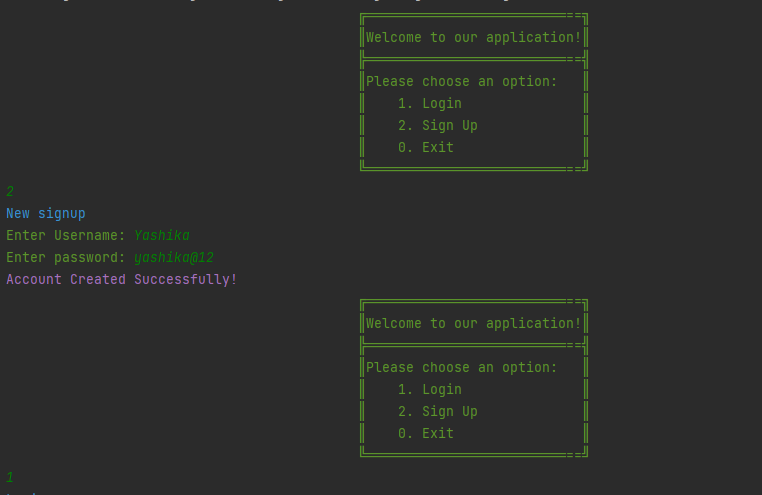
****

**BLOCK DIAGRAM**



**OUTPUT SCREENSHOTS**

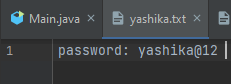
Signup:



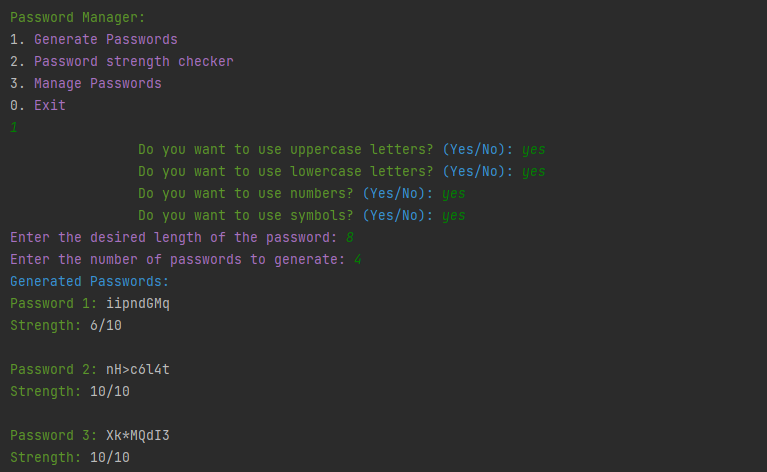
Login:



File:



Password generator:

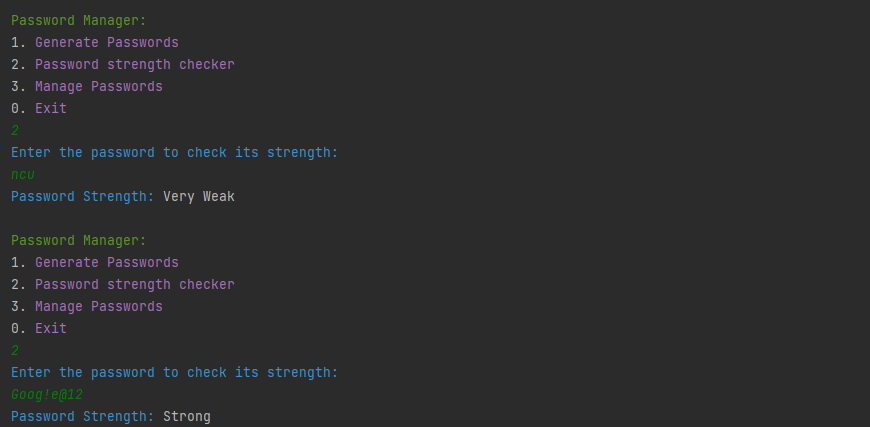




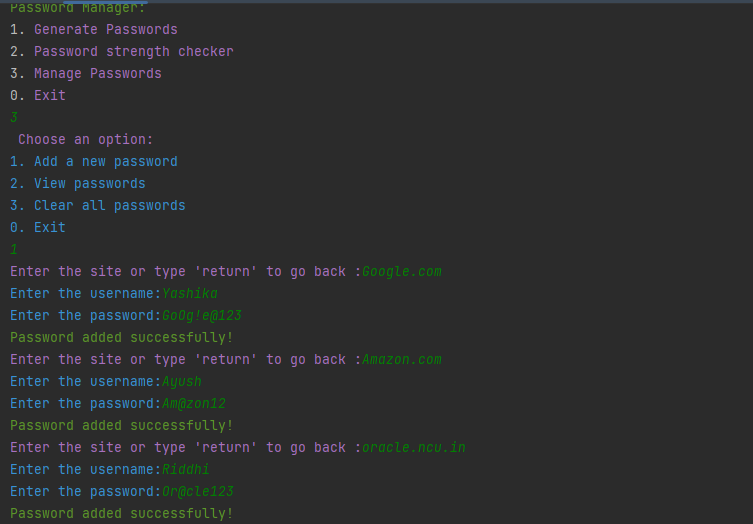
File:



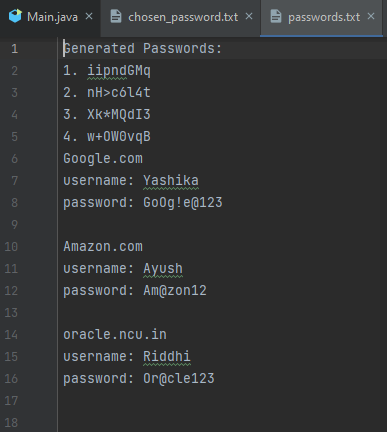
Password strength checker:

****

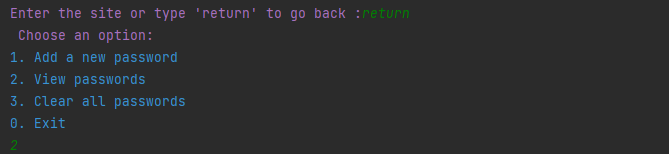
Password Manager(Add new Password):

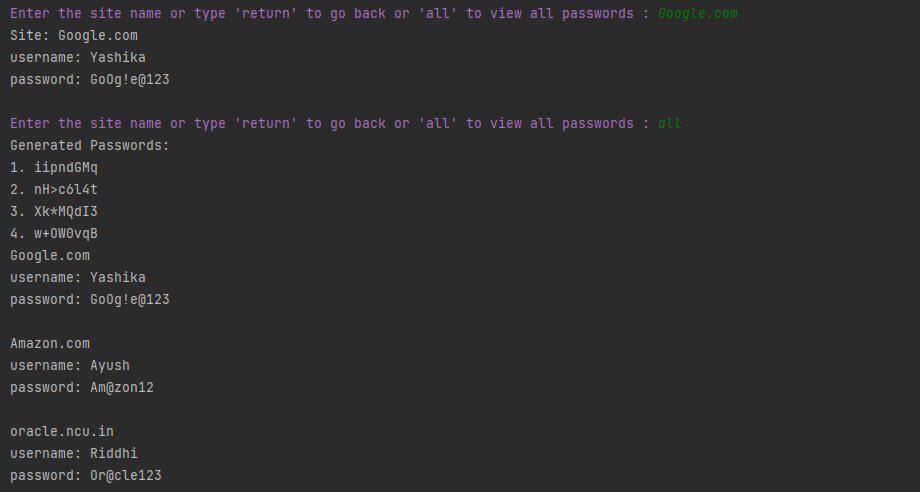
****

File:

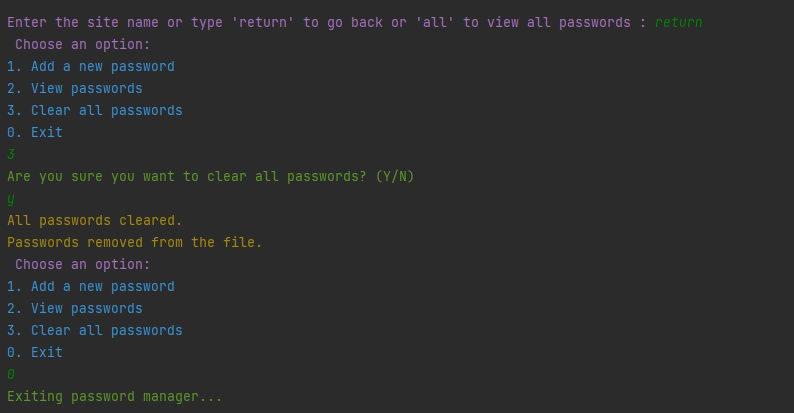


Password Manager(view password):





Password Manager(Clear All Password):



**ADVANTAGES**

1. **Increased security:** By using a password generator and strength checker, users can create strong and secure passwords that are difficult to guess or crack by hackers. This can help protect their accounts and personal information from unauthorized access.

2. **Convenience:** A password generator can quickly create random passwords for users, saving them time and effort in coming up with secure passwords on their own.

3. **Password variety:** A password generator can create a wide variety of passwords using different combinations of characters. This can help users avoid using the same password for multiple accounts, which is a common security risk.

4. **Improved password habits:** By using a password strength checker, users can receive feedback on the strength of their input passwords and learn how to create stronger passwords. This can help improve their overall password habits and security awareness.

5. **Reduced risk of human error:** Humans are prone to creating weak passwords that are easy to remember but also easy to guess or crack. By using a password generator and strength checker, users can reduce the risk of human error in creating weak passwords.

**LIMITATIONS**

Although we have added many main features in the program but it can be modified or can be made more advanced by adding some crucial features/methods which we can’t as per now. It may be due to lack of knowledge and skills as we are in the beginner stage.

**Some limitations are:**

**1. Limited criteria:** A password strength checker can only check for certain criteria such as the length of the password and the use of different types of characters. It may not be able to check for other factors that can affect password strength such as the use of common words or patterns.

**2. False sense of security:** Users may rely too heavily on a password generator and strength checker and assume that their passwords are secure without taking other security measures such as enabling two-factor authentication or regularly changing their passwords.

**3. Technical issues:** Like any software program, a password generator and strength checker can have technical issues or bugs that can affect its performance. Users should always keep their software up to date and report any issues to the developer.

# FUTURE SCOPE

* Make the application go online
* Enhanced security and privacy policies
* New methods of use**r** verification while signing in
* Intelligent password strength checker to send alert for commonly used passwords
* Helping the user account to withstand modern cyber attacks
* GUI based application
* Forgot password feature in case the user forgets his credentials

# CONCLUSION

In conclusion, our project on developing a password generator and strength checker has successfully tackled the challenges associated with password security and management. By providing users with a reliable password generator and strength assessment tool, we have empowered them to create strong, unique passwords that enhance their online security.

Throughout the project, we have focused on user-friendliness and convenience, offering features such as password storage and auto-fill functionality to simplify the password management process. These additions not only save users time but also reduce the risk of password-related vulnerabilities.

We have overcome challenges and gained valuable insights into password security, encryption techniques, and user interface design. The project's success demonstrates our ability to implement best practices and contribute to a safer online environment.

While our project has achieved its objectives, there are opportunities for future enhancements, such as integrating two-factor authentication (2FA) and exploring secure synchronization across multiple devices.

In conclusion, our project provides a valuable tool for addressing password security concerns. By promoting strong passwords and user-friendly management, we aim to make a positive impact on users' online security practices.

Overall, this project has been a rewarding experience, allowing us to apply our knowledge, enhance our programming skills, and contribute to a more secure digital landscape.

# REFERENCES

## Project Geeks

<https://projectsgeek.com>

## Foss life

<https://www.fosslife.org/build-random-password-generator-java>

## Tutorials point

## <https://www.tutorialspoint.com/Generating-password-in-Java>