# Project Report: Java Password Generator

## 1. Introduction

In today's digital world, strong and secure passwords are essential for protecting user data and privacy. This project implements a Password Generator in Java that creates secure, random passwords containing a mix of uppercase letters, lowercase letters, digits, and special symbols. The generator ensures compliance with common security recommendations and provides customizable password lengths.

## 2. Problem Definition

The objective is to develop a tool that generates strong, unpredictable passwords to help users enhance their online security. The password generator must:

* Use secure randomization techniques.
* Guarantee the inclusion of all character types.
* Allow users to specify password length (with a minimum enforced).
* Be easy to use and integrate into other Java applications.

## 3. Methodology

## 3.1. Requirements Analysis

* Input: Desired password length.
* Output: A random, secure password string.

## 3.2. Design

* Utilize Java’s SecureRandom class for cryptographic security.
* Define character pools for lowercase, uppercase, digits, and symbols.
* Ensure that the generated password contains at least one character from each pool.
* Shuffle the final password to avoid predictable patterns.

## 3.3. Implementation

The main logic is encapsulated in the PasswordGenerator class, which provides a static method to generate passwords. The code ensures:

* Minimum password length of 8 characters.
* Inclusion of all character types.
* Secure random selection and shuffling.

## Code Snippet

java

public class PasswordGenerator {

*// ... (see previous code)*

}

## 3.4. Testing

* Generated passwords were checked for the presence of all required character types.
* Passwords were tested for randomness and unpredictability.
* Edge cases, such as minimum length and very large lengths, were validated.

## 4. Results

The password generator successfully produces secure passwords, such as:

* A9f$k2Lp!q
* bQ3@Wz7#uT
* m!8P2rS#vT

All generated passwords meet the criteria for length and complexity.

## 5. Conclusion

The Java Password Generator fulfills its objective by providing a simple, secure, and effective way to generate strong passwords. It can be integrated into larger systems or used as a standalone utility.

## 6. Future Enhancements

* Add a graphical user interface (GUI) for easier use.
* Allow users to customize the character set (e.g., exclude similar-looking characters).
* Add options to generate passphrases (multiple words).
* Integrate with password managers or browsers.

## 7. References

* [Java SecureRandom Documentation](https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/security/SecureRandom.html)
* [NIST Password Guidelines](https://pages.nist.gov/800-63-3/sp800-63b.html)
* [OWASP Password Storage Cheat Sheet](https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html)

Prepared by:  
[Ganesh Kale]  
[Date: June 20, 2025]

If you want, I can provide a Markdown or LaTeX version for even easier export to PDF. Let me know if you need further help!