Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 28 No. 2 (2025)

Designing a Random Password Generator and OTP Using Java Programming Language

Rakesh Kumar Giri

Associate Professor,
Department of Computer Science & Engineering,
Saisha Institutions,Chennai, India
Email: rakeshgiriap@gmail.com

Article History:

Received: 13-02-2025
Revised: 25-03-2025
Accepted: 06-04-2025

Abstract:

A password generator is a tool that helps users create strong and secure passwords. In this presentation, we will explore the topic of password generators using Java programming language. Java is a popular programming language that is widely used for developing secure applications. It has many built-in libraries and functions that make it easy to develop password generators with various features. Passwords are the first line of defense against cyber attacks. Weak passwords can be easily hacked, putting personal and sensitive information at risk. Password generators help users create complex and unique passwords that are difficult to guess or crack. They also save time and effort compared to manually creating passwords. Password generators use algorithms to generate random strings of characters that meet certain criteria, such as length, complexity, and inclusion of special characters. Java provides several libraries and functions that can be used to generate random numbers and characters. These can be combined to create a custom password generator that meets specific requirements.

Keywords: Java, Password, OTP, Generator, Cyber Security, Random, Interface

I.INTRODUCTION

In ancient times, without technology, private information was kept in lockers. These lockers provided optimal security for the files, making the risk of any valuable information being compromised nearly non-existent, as data owners always kept the key to the lockers securely with them. Nonetheless, in contemporary times, the concept of data protection has changed significantly. Sensitive information is currently kept on devices such as computers and smart phones. They are kept in different files and folders that are integrated into these devices. Therefore, to offer better security for such information, the idea of passwords has been developed.

Users are able to generate their own passwords and security codes to safeguard their information. They frequently choose passwords that are simple to recall, like their own names, the names of loved ones, birth dates, or other significant dates. Sadly, hacking and cyber security breaches are increasing nowadays, and these easily understandable passwords are quickly compromised by attackers. Therefore, it has become crucial to enhance cyber security and safeguard data. Developing strong passwords for users to safeguard their

information in browsers and applications is a significant measure in that direction. This provides enhanced security and makes it harder for hackers to breach them.

The password generator developed by the team in Java aids this mission by offering users complex password combinations for enhanced protection of their information. The password-generating system was developed on the Java coding platform, utilizing meticulously crafted code that leverages Java's capabilities. The project has thoroughly examined these libraries and their applications in developing a system that offers distinct combinations of passwords.

This document provides a thorough overview of the generator's code and interface design, the planning and execution processes, the challenges encountered during the system's development, and the successful final result achieved.

II.METHODOLOGY

We may go through the generate a one time password or unique identification url article before this for a better understanding of how to generate passwords and OTP in Java. Have you ever clicked on "Forgot Password" and received a new password or OTP instantly on your email or phone? This process uses dynamically generated passwords and one-time passwords (OTPs) to enhance security. In this article, you'll learn how to generate secure passwords and OTP's in Java using simple techniques.

What are Passwords and OTPs?

Password: A static secret string used to verify identity during login.
OTP (One-Time Password): A temporary, randomly generated code used once for verification .

Method 1:

Java program explaining the generation of Password

```
import java.util.Random;
public class Tester{
  public static void main(String[] args) {
    System.out.println(generatePassword(8));
  }

  private static char[] generatePassword(int length) {
    String capitalCaseLetters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    String lowerCaseLetters = "abcdefghijklmnopqrstuvwxyz";
    String specialCharacters = ”!@#$%^&*()_-=+[]{}\|:;”’,.<>/?”;
    String numbers = "1234567890";
```

```
        String combinedChars = capitalCaseLetters + lowerCaseLetters + specialCharacters + numbers;
      Random random = new Random();
      char[] password = new char[length];

     password[0]=lowerCaseLetters.charAt(random.nextInt(lowerCaseLetters.length()));
     password[1]=capitalCaseLetters.charAt(random.nextInt(capitalCaseLetters.length()));
     password[2]=specialCharacters.charAt(random.nextInt(specialCharacters.length()));
      password[3]=numbers.charAt(random.nextInt(numbers.length()));

      for(int i = 4; i< length ; i++) {
        password[i] = combinedChars.charAt(random.nextInt(combinedChars.length()));
      }
      return password;
   }
}
```

Output

cF#0KYbY

Method 2:

Java program explaining Random Password Generator

```
import java.security.SecureRandom;

public class Password Generator{
private static final string LOWERCASE_CHARS="abcdefghijklmnopqrstuvwxyz";
private static final string UPPERCASE_CHARS="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
private static final string DIGIT_CHARS="1234567890";
private static final string SPECIAL_CHARS="!@#$%^&*()_-=+[]{}\|:;"',.<>/?";
private static final string ALL_CHARS=LOWERCASE_CHARS+ UPPERCASE_CHARS+ DIGIT_CHARS+ SPECIAL_CHARS:
public static String generateRandomPassword(int length)
{
```

```
If(length<=0){
throw new IllegalArgumentException(“Password length must be greater than zero:”):
}
Secure Random random = new Secure Random();
String Builder password=new String Builder(length):
for(inti=0;i<length;i++)
{
int randomIndex= random.nextInt(ALL_CHARS.length());
password.append(ALL_CHARS.charat(randomIndex));
}
return password to string();
}
public static void main(string[]args)
{
int passwordLength=12:
String generated Password=generate Random Password(passwordlength):
System.out.println(“Generated Password:”+generated Password):
}
}
```

Note : The password we are generating will change every time. As we have used random() method to generate the password.

Output :

```
Generating password using random() :
Your new password is :  Ke4;cZ/TM8yp
```

Java program explaining the generation of OTP(One Time Password)

```
import java.util.*;
public class NewClass
{
static char[] OTP(int len)
```

```
{
        System.out.println("Generating OTP using random():");
        System.out.print("You OTP is:");
        String numbers = "0123456789";
        Random rndm_method = new Random();
        char[] otp = new char[len];

        for (int i = 0; i< len; i++)
        {
    otp[i] = numbers.charAt(rndm_method.nextInt(numbers.length()));
        }
        return otp;
    }
    public static void main(String[] args)
    {
        int length = 4;
        System.out.println(OTP(length));
    }
}
```

Note : The OTP we are generating will change every time. As we have used random() method to generate the OTP.

Output :

Generating OTP using random() :
You OTP is : 5291

III.CONCLUSION

Random Password Generator and OTP Theory

A secure password generator must produce strings that are unpredictable, sufficiently long, and composed of a variety of character types.

Key Principles:

1. **True Randomness:** Avoid simple Math.random() or the basic java.util.Random. These are pseudo-random number generators (PRNGs) that can be predictable in secure contexts.
   - **Solution:** Use java.security.SecureRandom, which utilizes the operating system's entropy sources (like system noise or disk timings) to provide cryptographically strong random values [1].
2. **Character Sets:** A strong password should mix different character types to increase entropy and complexity. Standard sets include:
   - Uppercase letters (A-Z)

- Lowercase letters (a-z)
- Numbers (0-9)
- Special characters (!@#$%^&*)

3. **Length and Complexity:** The strength of a password is a function of its length and the size of the character set used. The generator should allow users to specify desired length and complexity (e.g., minimum number of special characters required) .

One-Time Password (OTP) Generator Theory

An OTP system generates a single-use code that is valid for a short duration.

In this paper i have designed a Random Password Generator and OTP using Java Programming Language. It is a programming based paper and a real realistic need in this E-world.

REFERENCES

1.Bonneau, J., Herley, C., Van Oorschot, P. C., & Stajano, F. (2012). "The quest to replace passwords: A framework for comparative evaluation of web authentication schemes." IEEE Symposium on Security and Privacy, 2012, pp. 553-567.

2. Florêncio, D., & Herley, C. (2007). "A large-scale study of web password habits." Proceedings of the 16th International Conference on World Wide Web, pp. 657-666.

3.International Journal of Research Publication and Reviews, Vol 6, Issue 3, pp 4578-4584 March 2025 4584

4. NIST (National Institute of Standards and Technology). (2021). "Digital Identity Guidelines: Authentication and Lifecycle
Management." NIST Special Publication 800-63B.

5. Wang, D., Wang, P., & Wang, X. (2017). "Defending against offline dictionary attacks with password stretching techniques." IEEE Transactions on Information Forensics and Security, 12(6), 1438-1452.

6. Das, A., Bonneau, J., Caesar, M., Borisov, N., & Wang, X. (2014). "The tangled web of password reuse." Proceedings of the 2014 Network and Distributed System Security Symposium (NDSS).

7. Ferguson, N., & Schneier, B. (2010). "Cryptographic randomness and password security." IEEE Security & Privacy, 8(4), 45-52.

8. Ur, B., Kelley, P. G., Komanduri, S., Lee, J., Maass, M., Mazurek, M. L., & Reeder, R. W. (2016). "How does your password measure up? The effect of strength meters on password creation." Proceedings of the 21st USENIX Security Symposium, pp. 65-80.

9. AlSabah, M., & Chiasson, S. (2021). "The impact of password managers on password security and memorability." ACM Transactions on Privacy and Security, 24(2), 1-31.

10.Kalyanapu Srinivas and V. Janaki, “A Novel Approach for Generation of OTP’S using Images”, Procedia Computer
Science, Vol. 85, pp. 511-518, 2016

11. Ankita Patil, Kiran Zambare, Preeti Yadav, Pankaj Wasulkarand Nisha Kimmatkar, “Integration of Encryption of File
and One Time Password for Secure File Access on Cloud”,International Journal of Advances in Computer Science and
Cloud Computing, Vol. 3, No. 1, pp. 1-13, 2015.