PASSWORD GENERATOR

**A PROJECT REPORT**

***Submitted by***

**DEVIPRIYA P (2303811710422024)**

***in partial fulfillment of requirements for the award of the course***

# CGB1201 - JAVA PROGRAMMING

***In***

# COMPUTER SCIENCE AND ENGINEERING

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

## SAMAYAPURAM – 621 112

**NOVEMBER- 2024**

# K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

**SAMAYAPURAM – 621 112**

**BONAFIDE CERTIFICATE**

Certified that this project report on **“PASSWORD GENERATOR”** is the bonafide work of **DEVIPRIYA P (2303811710422024)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



### SIGNATURE

Dr.A.Delphin Carolina Rani, M.E.,Ph.D., **HEAD OF THE DEPARTMENT** PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

### SIGNATURE

Mr. M. Saravanan, M.E., **SUPERVISOR** ASSISTANTP ROFESSOR

Department of CSE

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on 02.12.2024

|  |  |
| --- | --- |
|  |  |
| INTERNAL EXAMINER | EXTERNAL EXAMINER |

**DECLARATION**

I declare that the project report on **“PASSWORD GENERATOR”** is the result of original work done by us and best of our knowledge, similar workhas not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201- JAVA PROGRAMMING.**

.

**Signature**

DEVIPRIYA P

Place: Samayapuram

Date: 02.12.2024

#### ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequateduration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project guide **Mr. M. SARAVANAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patiencewhich motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

#### VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

#### MISSION OF THE INSTITUTION

* Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
* Be an institute with world class research facilities
* Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

#### VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

#### MISSION OF DEPARTMENT

**M1: Industry Specific:** To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

**M2: Research:** To prepare students for research-oriented activities.

**M3: Society:** To empower students with the required skills to solve complex technological problems of society.

#### PROGRAM EDUCATIONAL OBJECTIVES

1. **PEO1: Domain Knowledge**

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

#### PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

#### PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

#### PROGRAM SPECIFIC OUTCOMES (PSOs)

**PSO 1: Domain Knowledge**

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

#### PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

#### PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

#### PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

**ABSTRACT**

This program is a user-friendly, GUI-based password generator built in Java using the Swing framework. It allows users to create strong and secure passwords that can be customized to suit various security needs. The interface includes an input field for specifying the desired password length and a checkbox to include or exclude special characters in the password. Upon clicking the "Generate Password" button, the program validates the input to ensure a valid and positive password length is entered, preventing errors and providing immediate feedback if invalid data is detected. The generated password is displayed in a text field, making it easy to copy and use. The password generation logic combines uppercase letters, lowercase letters, numbers, and optionally special characters to create highly secure passwords. The program offers flexibility, catering to both basic and advanced password requirements by giving users control over the complexity of the output. This combination of security, customization, and ease of use makes the application suitable for a broad audience, including individuals managing personal accounts and professionals needing secure credentials. The intuitive design ensures a seamless user experience while reinforcing good password hygiene, making it a practical tool in today’s security-conscious environment.

#### ABSTRACT WITH POs AND PSOs MAPPING

**CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.**

|  |  |  |
| --- | --- | --- |
| **ABSTRACT** | **POs MAPPED** | **PSOs MAPPED** |
| This Java program provides a GUI-based password generator using the Swing framework. Users can customize password length and choose to include special characters. The application ensures secure password generation by combining uppercase letters, lowercase letters, numbers, and optional special characters. With a simple interface and input validation, it offers a reliable and user-friendly tool for creating strong passwords. | **PO1 -3** |  |
| **PO2 -3** |  |
| **PO3 -3** |  |
| **PO4 -3** |  |
| **PO5 -3**  **PO6 -3**  **PO7 -3**  **PO8 -3** | **PSO1 -3**  **PSO2 -3**  **PSO3 -3** |
| **PO9 -3** |  |
| **PO10 -3** |  |
| **PO11-3** |  |
| **PO12 -3** |  |

Note: 1- Low, 2-Medium, 3- High

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | **viii** |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1 Objective | 1 |
|  | 1.2 Overview | 1 |
|  | 1.3 Java Programming Concepts | 1 |
| **2** | **PROJECT METHODOLOGY** | **4** |
|  | * 1. Proposed Work | 4 |
|  | * 1. Block Diagram | 4 |
| **3** | **MODULE DESCRIPTION** | **5** |
|  | 3.1 Generate Password Module | 5 |
|  | 3.2 User Input Validation Module | 5 |
|  | 3.3 Graphical User Interface Module | 5 |
|  | 3.4 Event Handling Module | 5 |
| **4** | **CONCLUSION & FUTURE SCOPE** | **6** |
|  | 4.1 Conclusion | **6** |
|  | 4.2 Future Scope | 6 |
|  | **REFERENCES** | **7** |
|  | **APPENDIX A(SOURCE CODE)** | **8** |
|  | **APPENDIX B(SCREENSHOTS)** | **11** |

**CHAPTER 1**

**INTRODUCTION**

* 1. **Objective**

This Java Swing application provides a user-friendly interface for generating secure and customizable passwords. Users can specify the desired length of the password and decide whether to include special characters, enhancing password complexity and security. The program validates user input, ensuring the entered length is a positive number, and displays helpful error messages for invalid inputs. Once the criteria are set, the program generates a random password using a combination of uppercase letters, lowercase letters, numbers, and optionally special characters. This makes it ideal for personal use, where strong passwords are needed for online accounts, devices, or applications. Additionally, the application serves as a practical educational tool for learning about randomness, user input handling, and GUI development in Java. By promoting the creation of strong passwords, it also contributes to better security practices.

**1.2 Overview**

The Password Generator GUI is a Java-based application designed to create secure and customizable passwords through an intuitive graphical user interface built with the Swing framework. Users can specify the desired password length and choose whether to include special characters, offering flexibility for different security needs. The program ensures user-friendly operation with clear instructions, real-time input validation, and straightforward options for customization The core password generation process utilizes a combination of uppercase letters, numbers, and optional special characters to deliver strong, random passwords. The generated password is displayed in a text field for easy access. This program highlights simplicity and practicality, making it suitable for both personal and professional use while promoting strong password practices in a security-conscious environment.

**1.3 Java Programming Concepts**

**The basic concepts of Object-Oriented Programming (OOP) are:**

* **Class and Object:** A class is a blueprint, and an object is an instance of the class.
* **Encapsulation:** Bundles data and methods into a single unit (class) while restricting direct access to data.
* **Inheritance:** Enables a class (child) to inherit properties and methods from another class (parent), promoting code reuse.
* **Polymorphism:** Allows methods to perform differently based on the object context (e.g., method overloading and overriding).
* **Abstraction:** Hides implementation details and exposes only essential features, simplifying system design.

**Project related concepts**

**1. Classes and Objects**

* The program defines a class PasswordGeneratorGUI that encapsulates all the components and logic for the password generator.
* An object of the PasswordGeneratorGUI class is created in the main method to launch the application.

**2. GUI Components**

* **Swing Components:** The program uses Java's Swing library (javax.swing) to create a graphical user interface (GUI). Components like JFrame, JButton, JTextField, JLabel, and JCheckBox are used for displaying the application window and taking user input.
* **Layout Management:** The FlowLayout layout manager is used to arrange the GUI components in a simple flow (from top to bottom or left to right).

**3. Event Handling**

* **ActionEvent:** The program responds to user actions, specifically button clicks, using the ActionListener interface. When the user clicks the "Generate Password" button, an ActionEvent is triggered.
* **EventListener:** The ActionListener is implemented in the generateButton.addActionListener method. This listener method processes the user input and triggers the password generation logic.

**4.Randomization**

* **Random Class:** The program uses the java.util.Random class to generate random numbers. The random numbers are used to select characters from the predefined pools (uppercase, lowercase, numbers, and special characters) to form the password.
* **StringBuilder:** To efficiently build the password string, the program uses StringBuilder. This class is ideal for building strings dynamically, especially when modifying strings frequently (like appending random characters).

**5. String Manipulation**

* **String Concatenation:** The program concatenates different sets of characters (uppercase, lowercase, numbers, special characters) to form the allCharacters string, from which random characters will be selected.
* **String Indexing:** The program selects a random character from the allCharacters string using the charAt() method, which accesses a character at a specific index.

**6. Methods**

* **Custom Methods:** The generatePassword method is a custom method that contains the logic for generating the password based on the user’s input (length and whether special characters are included).
* **Return Statement:** The generatePassword method returns the generated password as a string to be displayed in the password field.

**CHAPTER 2**

**PROJECT METHODOLOGY**

* 1. **Proposed Work**

To enhance the Password Generator GUI, several improvements can be made. Adding a Clear Password button would let users reset the form easily, while a Copy to Clipboard button would make it convenient to use the generated password elsewhere. A password strength indicator could provide real-time feedback, showing whether the password is weak, medium, or strong based on length and complexity. Tooltips on components could improve usability by explaining features like the checkbox for special characters. Validation could also be enhanced by dynamically ensuring the length is sufficient and warning users about the risks of excluding special characters.

* 1. **Block Diagram**

FALSE

TRUE

START

STOP

Generate random password character and added to the previous password

Is password reached the length?

Define character set ,password length and initialize password

**CHAPTER 3**

**MODULE DESCRIPTION**

**3.1 Generate Password Module**

This module is responsible for generating secure and random passwords based on user-defined parameters. It takes into account the desired password length and whether to include special characters. The password is created by randomly selecting characters from a pool of uppercase letters, lowercase letters, numbers, and optionally special characters.It define character pools (uppercase, lowercase, numbers, special characters). Randomly select characters from the combined pool until the desired length is reached. Return the generated password to the user.The module ensures that the generated password is both secure and satisfies the user's preferences. This is the core functionality of the program.

**3.2 User Input Validation Module**

This module handles the validation of user inputs, ensuring that the program can process the inputs correctly without errors. It validates the password length must be a valid positive integer.If an invalid input is detected (e.g., non-numeric length), an error message is displayed to the user. This module ensures robustness by preventing crashes and guiding users with clear instructions.

**3.3 Graphical User Interface (GUI) Module**

This module provides a user-friendly interface for interacting with the password generator. It includes a text field for the user to specify the desired password length,checkbox to allow users to decide whether to include special characters and a button to generate the password. A display area where the generated password is shown. The GUI uses Java Swing components such as JFrame, JTextField, JLabel, JCheckBox, and JButton. The layout ensures the application is intuitive and easy to do.

**3.4 Event Handling Module**

This module manages user interactions with the GUI. It listens for events such as button clicks and processes user inputs accordingly. When the Generate Password button is clicked, it triggers the password generation process. Displays error messages using a popup dialog (JOptionPane) for invalid inputs

**CHAPTER 4**

**CONCLUSION & FUTURE SCOPE**

**4.1 CONCLUSION**

In conclusion, the Password Generator GUI is a straightforward yet effective Java application that enables users to create secure, customizable passwords. Through a simple graphical interface, users can specify the desired password length and choose whether to include special characters, offering flexibility in password creation. The program uses Java’s Swing framework for the user interface and employs basic programming concepts like event handling, input validation and random password generation to ensure both functionality and user-friendliness .The program also includes error handling to manage invalid inputs, ensuring a smooth user experience without crashes or unexpected behavior. By leveraging the Event Dispatch Thread for event-driven programming, the application remains responsive, even with user interactions. Overall, this project serves as a practical tool for generating secure passwords and demonstrates key aspects of Java programming, including GUI design, event handling, and error management.

**4.2 FUTURE SCOPE**

The Password Generator GUI project has significant potential for future enhancements to increase its utility and appeal. Advanced customization options could allow users to define specific requirements, such as minimum counts of certain character types or excluding ambiguous characters like O and 0. A password strength indicator can provide immediate feedback on the security of generated passwords. Additional features like saving passwords securely, copying them to the clipboard, or generating multiple passwords at once could make the application more practical. For improved security, switching to cryptographically secure random number generation (e.g., SecureRandom) is essential. Enhancements in UI/UX, such as modern designs, dark mode, and multi-language support, could improve accessibility and user satisfaction. Furthermore, integrating the application with password managers, creating standalone desktop or mobile versions, and enabling secure storage of passwords would expand its functionality and reach. These improvements would make the tool not only a password generator but a comprehensive solution for password management.

**REFERENCES**

### Java Books:

**1."Head First Java" by Kathy Sierra and Bert Bates**

This book is a great resource for beginners learning Java, with a focus on object-oriented programming concepts and real-world application development.

**2. "Effective Java" by Joshua Bloch**

A deeper dive into best practices for writing clean, maintainable Java code. It covers advanced topics like Java collections, concurrency, and design patterns that could be applied to more complex payroll systems.

### Websites:

1. **GeeksforGeeks - Java Tutorials**

* URL: https://www.geeksforgeeks.org/java/
* A comprehensive collection of tutorials on Java, covering topics like classes, objects, inheritance, encapsulation, and more. Great for learning the core concepts of Java and applying them in projects like EPMS.

1. **W3Schools - Java Tutorial**

* URL: https://www.w3schools.com/java/
* A beginner-friendly resource that offers tutorials on Java programming, including object-oriented principles and core Java concepts.

### YouTube Links:

### ****Java for Beginners - Java Brains****

* URL: [https://www.youtube.com/user/koushks](https://www.youtube.com/user/koushks" \t "_new)
* Offers Java tutorials from the basics to advanced concepts. The channel provides detailed guides on Java programming, including working with objects and classes, which are crucial for building an EPMS.

#### APPENDIX A

#### (SOURCE CODE)

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Random;

public class PasswordGeneratorGUI extends JFrame {

private JTextField passwordField;

private JTextField lengthField; // Input field for password length

private JCheckBox specialCharsCheckBox; // Checkbox to include special characters

private JButton generateButton;

private JLabel instructionLabel;

private JLabel lengthLabel; // Label for password length input

public PasswordGeneratorGUI() {

// Set up the frame

setTitle("Password Generator");

setSize(400, 250);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

setLayout(new FlowLayout());

// Initialize components

instructionLabel = new JLabel("Click the button to generate a password:");

lengthLabel = new JLabel("Enter password length: ");

lengthField = new JTextField(5); // Text field for length input

specialCharsCheckBox = new JCheckBox("Include special characters"); // Checkbox for special chars

passwordField = new JTextField(20);

generateButton = new JButton("Generate Password");

// Add action listener to the button

generateButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

int passwordLength = Integer.parseInt(lengthField.getText()); // Get user-defined length

if (passwordLength <= 0) {

JOptionPane.showMessageDialog(null, "Please enter a positive number for length.");

} else {

// Get whether special characters should be included

boolean includeSpecialChars = specialCharsCheckBox.isSelected();

String generatedPassword = generatePassword(passwordLength, includeSpecialChars);

passwordField.setText(generatedPassword);

}

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null, "Please enter a valid number for the length.");

}

}

});

// Add components to the frame

add(instructionLabel);

add(lengthLabel);

add(lengthField); // Text field for length input

add(specialCharsCheckBox); // Checkbox for special chars

add(passwordField);

add(generateButton);

}

// Function to generate a random password

public String generatePassword(int length, boolean includeSpecialChars) {

String uppercase = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

String lowercase = "abcdefghijklmnopqrstuvwxyz";

String numbers = "0123456789";

String specialChars = "!@#$%^&\*()-\_+=<>?/[]{}|";

String allCharacters = uppercase + lowercase + numbers;

// If user wants to include special characters, add them to the pool

if (includeSpecialChars) {

allCharacters += specialChars;

}

Random random = new Random();

StringBuilder password = new StringBuilder();

for (int i = 0; i < length; i++) {

int index = random.nextInt(allCharacters.length());

password.append(allCharacters.charAt(index));

}

return password.toString();

}

public static void main(String[] args) {

// Create an instance of the GUI and make it visible

PasswordGeneratorGUI frame = new PasswordGeneratorGUI();

frame.setVisible(true);

}

**APPENDIX B**

**(SCREENSHORTS)**



