PMIST • PERIYAR MANIAMMAI INSTITUTE OF SCIENCE & TECHNOLOGY

PERIYAR MANIAMMAI

INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University)

Established Under Sec. 3 of UGC Act, 1956 • NAAC Accredited

think • innovate • transform

PROJECT REPORT

Topic: Password Generator

Course Name: Introduction to Artificial intelligence

Course Code: XCSHA1

By

R. Rathisri (123011019026)

B. Kanniga Parameswari(123011019016)

G. Anupriya(123011019005)

Project Title: Password Generator

Project Objective:

The objective of this project is to develop a Python-based password generator application that creates secure, random passwords. The user can specify the desired length and complexity of the password (e.g., including uppercase, lowercase letters, digits, and special characters).

Technologies Used:

Programming Language: Python

Libraries Used: random, string

Project Overview:

The Password Generator program generates a secure, random password based on user-specified requirements. Users can choose the length of the password, and the program will create a password that includes a mix of uppercase letters, lowercase letters, digits, and special characters to ensure high security.

Program Design:

1. User Input:

The user is prompted to enter the length of the password.

The program also asks whether the user wants to include uppercase letters, lowercase letters, digits, and special characters in the password.

2. Random Password Generation:

Based on the user's input, the program creates a pool of characters to choose from.

It uses the random.choice() function to randomly select characters from the pool and generate the password.

3. Error Handling:

The program checks if the user input for the password length is a valid integer and whether it's a positive number.

4. Output:

The program displays the generated password to the user.

Code Explanation:

```
import random
import string
# Function to generate a secure random password
def generate_password(length, use_uppercase, use_lowercase, use_digits, use_special):
  characters = ""
  # Add character sets to the pool based on user preferences
  if use_uppercase:
    characters += string.ascii_uppercase  # A-Z
  if use_lowercase:
    characters += string.ascii_lowercase  # a-z
  if use_digits:
    characters += string.digits  # 0-9
  if use_special:
    characters += string.punctuation  # Special characters
  # Check if at least one character set is selected
  if not characters:
    raise ValueError("At least one character set (uppercase, lowercase, digits, or special) must be selected."
  # Generate a random password by selecting characters from the pool
  password = ''.join(random.choice(characters) for i in range(length))
  return password
```

```
# Function to get input from user and generate the password

def main():

  try:

    # Get password length from the user

    length = int(input("Enter the length of the password: "))

password

    # Validate password length

    if length < 1:

      print("Password length must be at least 1.")

      return

    # Get the user's preferences for password complexity

    use_uppercase = input("Include uppercase letters? (y/n):
").lower() == 'y'

    use_lowercase = input("Include lowercase letters? (y/n):
").lower() == 'y'

    use_digits = input("Include digits? (y/n): ").lower() == 'y'

    use_special = input("Include special characters? (y/n):
").lower() == 'y'

    # Generate the password

    password = generate_password(length, use_uppercase,
use_lowercase, use_digits, use_special)

    print(f"Generated password: {password}"

  except ValueError as ve:

    print(f"Error: {ve}")

  except Exception as e:

    print(f"An unexpected error occurred: {e}")

# Run the program

if __name__ == "__main__":

  main()
```

Sample Input and Output:

Sample Input:

```
Enter the length of the password: 12
Include uppercase letters? (y/n): y
Include lowercase letters? (y/n): y
Include digits? (y/n): y
Include special characters? (y/n): y
```

Sample Output:

```
Generated password: Wb2#FzQ8@eA1
```

Conclusion:

The Password Generator program successfully demonstrates how to generate secure, random passwords based on user input. It uses Python's random and string modules to create passwords that are both random and secure. The program is simple but effective, and there is potential for further enhancement to improve the user experience and security features.