**Student Name:** Aakash Thapliyal  **UID:** 24MCC20027

**Branch:** MCA(ccd) **Section/Group:** A 1

**Semester:** 1st  **Date of Performance:** 04-11-2024

**Subject Name: :** Linux Administration Lab  **Subject Code:** 24CAP-607

**Q. Title of Project.** Password Generator Using Shell-script

**Aim/Overview of the practical:** Password security is a critical aspect of digital security practices. Using complex, unpredictable passwords helps to protect sensitive data from unauthorized access. This project demonstrates a simple method to generate secure passwords using shell scripting in Linux, which relies on the randomness provided by /dev/urandom, a pseudo-random number generator. The script filters characters to include only alphanumeric characters for readability and versatility.

1. **Task to be done:** The main objective of this project is to create a shell script that generates a secure, random password of a specified length. This script will prompt the user to enter the desired password length and generate a random alphanumeric password based on the user's input. It serves as a simple but effective way to create strong passwords for secure systems.
2. **Script’s Explanation:** The script is written in Bash and includes the following key components:
3. **Function Definition (**generate\_password**)**:
   1. This function accepts one argument: length, which determines the number of characters in the generated password.
   2. It uses the tr command with /dev/urandom to generate a random alphanumeric string:

\* /dev/urandom produces a stream of random bytes.

\* tr -dc 'A-Za-z0-9' filters these bytes to include only uppercase and lowercase letters (A-Za-z) and numbers (0-9).

\* head -c "$length" restricts the output to the specified number of characters.

1. **User Input**:

\* The script prompts the user to input their desired password length using the read command.

\* It then validates the input to ensure it is a positive integer using a regular expression.

1. **Input Validation**:

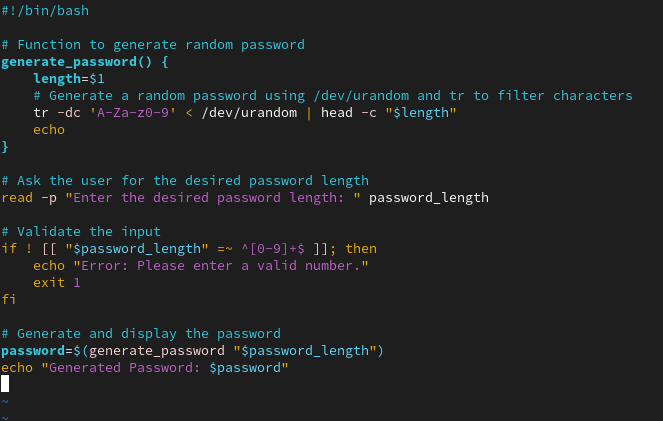
If the input does not match the pattern for a positive integer (^[0-9]+$), the script outputs an error message and exits with a status of 1 (indicating an error).

1. **Password Generation and Display**:

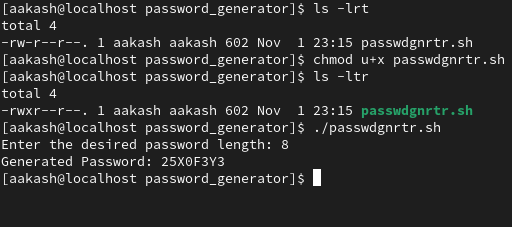
\* After validating the input, the script calls generate\_password with the specified length and stores the generated password in the password variable.

\* The password is then displayed to the user.

**3. Code for experiment/practical:**

****

**4. Result/Output/Writing Summary:**

****

**5. Future Scope:**To further improve the script, additional options could be added to include special characters, enforce minimum length, and check for repeated characters or character classes (e.g., uppercase, lowercase, digits).

**6. Modules Used:**

The script ensures secure password generation by:

1.Using /dev/urandom for randomness, which is non-blocking and suitable for most non-cryptographic uses.

1. Restricting the password characters to alphanumeric symbols for compatibility across various systems.

**7. Conclusion:**This project demonstrates a basic shell scripting technique for generating random passwords in a secure and user-friendly way. The script allows users to quickly create strong passwords of customizable lengths, enhancing overall system security.

**8. Learning outcomes (What I have learnt):**

· **Understanding of Shell Scripting Basics**: Knowledge of fundamental shell scripting concepts, including functions, user input handling, and input validation.

· **Secure Password Generation:** Understanding how to utilize /dev/urandom to generate secure, random passwords using a shell script.

· **Practical Application of Linux Commands:** Using commands like tr, head, and read to manipulate data, handle user input, and control output formatting.