Below is the README.md File and the Raw Code used in pdf

As from my Individual Essay, not only did it mention the benefits of the web-based Appointment and Scheduling Management Information System (ASMIS) to Queens Medical Centre and the sick residents, but also highlighted the potential cyberthreats, and know-how design and implementation of a secure system.

Below is a description of a few Python code solutions that can be applied in the Appointment and Scheduling Management Information System (ASMIS) to ensure reinforcement of Cybersecurity. These codes include:

1). Custom Password Validating Code:

In order to ensure that a user has a strong password, a code can be used to validate the password.

In the code, Regular Expressions (RegEx or re) is imported so that 're.search()' method can be used to search and detect a pattern of strings within a given criteria (Lenka, 2020).

First, it prompts for a password input.

Then checks the inputted password so as to confirm if all the below parameters are met:

- Is between 8 and 20 characters long - 'or' Boolean operator was used to join the code arguments into one statement (Novak, 2019)
- Has at least 1 capital letter
- Has at least 1 small letter

- Has any of these symbols, which are ~ ! @ # $ % ^ &
- Has at least 1 digit number
- Has no space

If any of the above criteria are not met, the system outputs an error message.

While loop is included in the code so as to allow repetition of the password_validator() function and 'break' is used to stop prompting the user for a password input when all the above criteria are met and validated (Campbell, 2022).

2). One-Time Pin/Password (OTP) Generating Code:

This code provides the user with an output of a One Time Pin or Password (OTP) that has 6 digits (by assigning a variable that has numbers and both capital and small letters and using for loop to define the 6-digit length of the OTP) (Anon, N.D.).

Importation of random library allows generation of any random numbers while math library gives access to common mathematical functions (Oliphant, 2007).

This 6-digit code can then be used by the user to authenticate the second or third time so as to be granted access to his/her account.

3). Custom Password Generator Code:

Since most users do not want to spend time thinking of new passwords every time they sign up to a new account or change passwords, password generating feature can be useful in eradicating password repetitions across a user's accounts.

This code, generate_password() function, has imported the random library and assigned a variable (string) and datatype (str) to all the characters that are to be included in the new password.

It first asks the user to input the length of the password he/she intends to generate.

If statement and ‘or’ Boolean operator checks if the length of the password inputted is between 8 to 20 and if it is not, it displays an error message.

While loop allows repetition of the user’s input until a correct number is entered.

After a true pass, the Random sample() method and join() method joins the random data as per the length and criteria entered, thus generating and outputting the password (Rawat, 2020).

4). Time-Based Login of 3 Attempts and Login Blocking Code:

This code helps in fighting cyberattacks like brute force, etc.

This code gives the user 3 attempts of inputting his/her username and password (which are already registered in the system as: **Username: Student1 and Password: Password1**).

After a failed attempt, the code gives you 5 seconds before inputting your credentials again and if the number of failed attempts is exhausted to 3, the program will block the login (Wind & , 2021).

This is made possible by importing the time library, and usage of nested if condition and while loop.

Below is the Raw Python Code

```
#Custom Password Validator

#https://www.geeksforgeeks.org/python-program-check-validity-password/

print("End of Module Assignment")

def password_validator():#defining a function

 """passwordvalidator"""

  import re #importing regex (re) library

  print("Password Validator")

  while (True): #allows looping till all have a positive check

    password= str(input("Please Enter Password: "))#assigns a variable and data type to user's input and also prints out a message asking the user for an input

    if (len(password)<8 or len(password)>20): #used or boolean operator to check the character length

       print('Password should be between 8 and 20 characters long ')

    elif not re.search("[A-Z]", password): #the searching regex checks the input and if it does not have a capital letter, the message below will be printed

       print('Password should have atleast 1 capital letter')
```

```
    elif not re.search("[a-z]", password): #the searching regex checks the input and if it does not have a small letter, the message  below will be printed
        print('Password should have atleast 1 small letter')
    elif not re.search("[1-9]",password): #the searching regex checks the input and if it does not have a number, the  message below will be printed
        print("Password should have atleast one number between 1 to 9")
    elif not re.search("[~!@#$%^&*]",password): #the searching regex checks the input and if it does not have a symbol, the  message below will be printed
        print("Password should have at least one of these symbols ~ ! @ # $ % ^ & * ")
    elif re.search(" ",password): #the searching regex checks the input and if it has a space, the message below will be printed
        print("Password should not contain any space")
    else: #if the input passes all the checks, it will output password is valid
        print("Password is valid")
        break  # it stops the loop once it outputs that the password is valid
 password_validator() #closing the function
```

#Generating and Printing a One Time Password/Pin (OTP) of 6 digit length

#https://codedec.com/tutorials/write-a-python-program-to-generate-a-one-time-password-otp/

```
def generate_OTP(): #defined the function
  import math, random # imported math and random libraries
  string = '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ' #defines all the characters that will be included in the generated OTP as a string data type
  OTP = "" #assigns a variable
  print('One Time Password/Pin (OTP) Generator') #outputs this as a title
  length = len(string) #assigns a variable
  for i in range(6): #defing the length of the OTP using for loop
      OTP += string[math.floor(random.random() * length)] #using the math and random library to generate a 6 digit OTP
  return OTP #returns the 6 digit OTP
if __name__ == "__main__":
  print("6 length OTP is:", generate_OTP()) #outputs the 6 digit OTP
```

#Custom Password Generator

#https://medium.com/analytics-vidhya/create-a-random-password-generator-using-python-2fea485e9da9

```
def generate_password(): #defining the function
  import random #importing random library
  string = str('0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ~!@#$%^&*')
  password = ""#assigns a variable
  print ("Password Generator") #outputs this as a title
  while (True): #enables looping of code until it reaches a true statement
    length = int(input("Enter the length of password: ")) #assigns a variable and data type to user's input and also prints out a message asking the user for an input
    if ((length < 8) or (length>20)): #used or boolean operator to check that the input entered is between 8 and 20, and if it is not it will output the message below
      print('Password length should be between 8 to 20 characters long')
    else:
      password="".join(random.sample(string,length))#Assigns a variable and joins the random data as per th length entered
      return password
```

```
    print("Your Password is: ", generate_password()) #outputs the generated password
```

#5 second Time-based login of 3 Attempts and after it is blocked

#https://stackoverflow.com/questions/69892361/block-login-if-too-many-login-attempts

```
def login_attempt(): #defining the function
    import time #importing time library
  print("Time-based Login Attempt")
  print('Enter the Correct Username and Password to Continue') #outputting an instruction to the user
  max_attempts=int(3) #assigning a variable and data type to show the number of times/attempts it will accept to run
  attempts=int(0) #assigning a variable and data type
  while True: #use while loop to repeat the code of input and validation of credentials
    username = str(input("Enter the Username: "))  #prompting for an input from the user and Assigning a variable and datatype on the inputted data
    password = str(input("Enter the Password: ")) #prompting for an input from the user and Assigning a variable and datatype on the inputted data
    if password=='Password1' and username=='Student1':#using if statement to set a condition
```

```
        print('Access Granted')
        break
    else:
        attempts+=1
        if attempts>= max_attempts: #setting a condition of what will be outputted after 3 failed attempts are up
            print("Access Blocked After 3 Attempts. Please Restart the Program and Try Again.")
            break
        print("Access Denied. Try again in 5 seconds.") #the output will be shown after a failed input during the 3 attempts
            time.sleep(5) #the time in seconds set to wait before the user tries to input the credentials again
login_attempt()   #closing the function
print('Thank You and Goodbye')
```

References

Anon, N.D.. Write a Python program to generate a One Time Password OTP. [Online]

Available at: https://codedec.com/tutorials/write-a-python-program-to-generate-a-one-time-password-otp/

[Accessed 28 May 2022].

Campbell, S., 2022. Python Loops While For Break, Continue Enumerate. [Online]

Available at: https://www.guru99.com/python-loops-while-for-break-continue-enumerate.html

[Accessed 29 May 2022].

Lenka, C., 2020. Python program check validity Password. [Online]

Available at: https://www.geeksforgeeks.org/python-program-check-validity-password/

[Accessed 28 May 2022].

Novak, N., 2019. Python if statements. [Online]

Available at: https://academicworks.cuny.edu/cgi/viewcontent.cgi?article=1036&context=bx_oers

[Accessed 29 May 2022].

Oliphant, T. E., 2007. Python for Scientific Computing. Computing in Science and Engineering, 9(3), pp. 10-20.

Rawat, A., 2020. Create a Random Password Generator using Python. [Online]

Available at: https://medium.com/analytics-vidhya/create-a-random-password-generator-using-python-2fea485e9da9

[Accessed 29 May 2022].

Wind, A. & R., 2021. Block login if too many login attempts. [Online]

Available at: https://stackoverflow.com/questions/69892361/block-login-if-too-many-login-attempts

[Accessed 29 May 2022].