NAME

HINA SAJID

ROLL NO

14650

SUBJECT

DATA STRUCTURE AND ALGORITHM

PROJECT

NAME :DATA SECURITY BY USING HASH ALGORITHM

<u>Project of DSA:</u>

<u>TITLE: Data Security by using hash algorithms</u>

<u>CODE:</u>

```
import hashlib

import re

from datetime import datetime


# Hashing functions

def md5_hash(data):

    hash_object = hashlib.md5()

    hash_object.update(data.encode('utf-8'))

    return hash_object.hexdigest()


def sha1_hash(data):

    hash_object = hashlib.sha1()

    hash_object.update(data.encode('utf-8'))

    return hash_object.hexdigest()


def sha256_hash(data):

    hash_object = hashlib.sha256()

    hash_object.update(data.encode('utf-8'))

    return hash_object.hexdigest()


# Function to check password strength

def check_password_strength(password):

    if len(password) < 8:

        return "Weak", "😟"

    elif re.search("[a-z]", password) and re.search("[A-Z]", password) and re.search("[0-9]", password)
and re.search("[!@#$%^&*(),.?\":{}|<>]", password):

        return "Strong", "🛡"

    else:
        return "Moderate", "😐"

# Function to create an account (hash the password)
def create_account(username, password, algorithm='sha256'):
    if algorithm == 'md5':
        hashed_password = md5_hash(password)
    elif algorithm == 'sha1':
        hashed_password = sha1_hash(password)
    else:
        hashed_password = sha256_hash(password)
    return username, hashed_password

# Function to verify password during login
def verify_password(username, input_password, stored_hash, algorithm='sha256'):
    if algorithm == 'md5':
        hashed_input = md5_hash(input_password)
    elif algorithm == 'sha1':
        hashed_input = sha1_hash(input_password)
    else:
        hashed_input = sha256_hash(input_password)

    if hashed_input == stored_hash:
        print("Password verified successfully! 🎉")
        return True
    else:
        print("Incorrect password! ❌")
        return False

# Function to display all accounts and their hashed passwords
def display_accounts(accounts):
    if accounts:
        print("\n--- Registered Accounts ---")
        for username, hashed_password in accounts.items():
            print(f"Username: {username}, Hashed Password: {hashed_password}")
    else:
        print("\nNo accounts registered yet.")

# Main function to run the Password Management System
def main():
    # Display a welcome message
    current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"🌟 Welcome to Data Security Using Hash Algorithms 🌟")
    print(f"Current Time: {current_time}\n")

    accounts = {}  # Dictionary to store username:hashed_password pairs

    while True:
        print("\n--- Password Management System ---")
        print("1. Create Account")
        print("2. Login")
        print("3. View All Accounts")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == '1':  # Create Account
            username = input("Enter a username: ")
            password = input("Enter a password: ")

            # Check password strength
            strength, emoji = check_password_strength(password)
            print(f"Your password strength: {strength} {emoji}")

            algorithm = input("Choose hashing algorithm (md5/sha1/sha256): ").lower()
            if algorithm not in ['md5', 'sha1', 'sha256']:
```

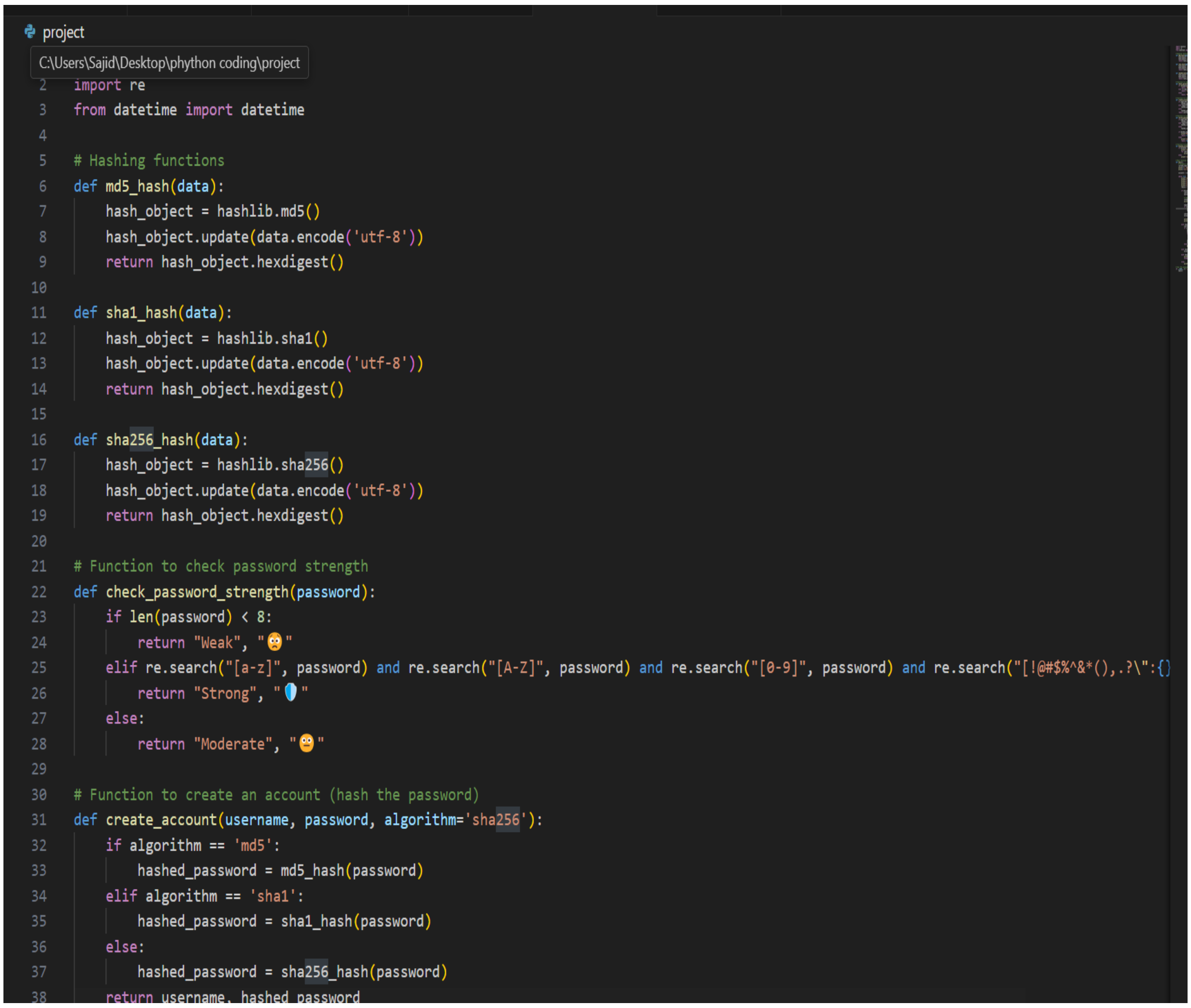
```
        print("Invalid algorithm selected! Defaulting to SHA-256.")
        algorithm = 'sha256'

    username, hashed_password = create_account(username, password, algorithm)
    accounts[username] = hashed_password

    print(f"\nAccount created for {username}.")
    print(f"Your password hashed with {algorithm.upper()}: {hashed_password}")

elif choice == '2':  # Login
    username = input("Enter your username: ")
    if username in accounts:
        password = input("Enter your password: ")
        algorithm = input("Choose hashing algorithm (md5/sha1/sha256): ").lower()
        if algorithm not in ['md5', 'sha1', 'sha256']:
            print("Invalid algorithm selected! Defaulting to SHA-256.")
            algorithm = 'sha256'

        stored_hash = accounts[username]
        verify_password(username, password, stored_hash, algorithm)
    else:
        print("Username not found! ❌")

elif choice == '3':  # View All Accounts
    display_accounts(accounts)

elif choice == '4':  # Exit
    print("Exiting the system. Goodbye! 👋")
    break

else:
    print("Invalid choice! Please try again. ❌")
```

# Run the program

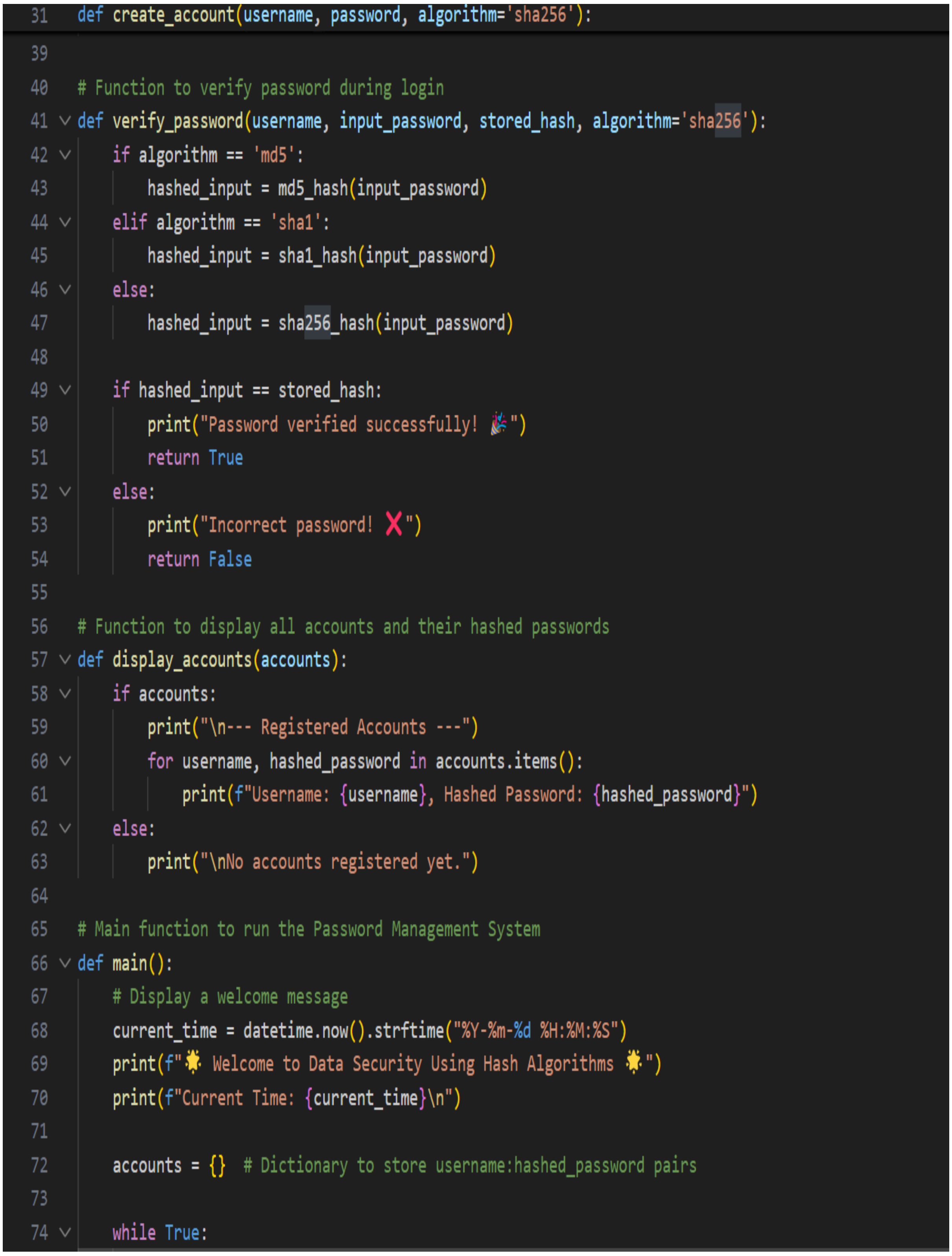
if __name__ == "__main__":

    main()

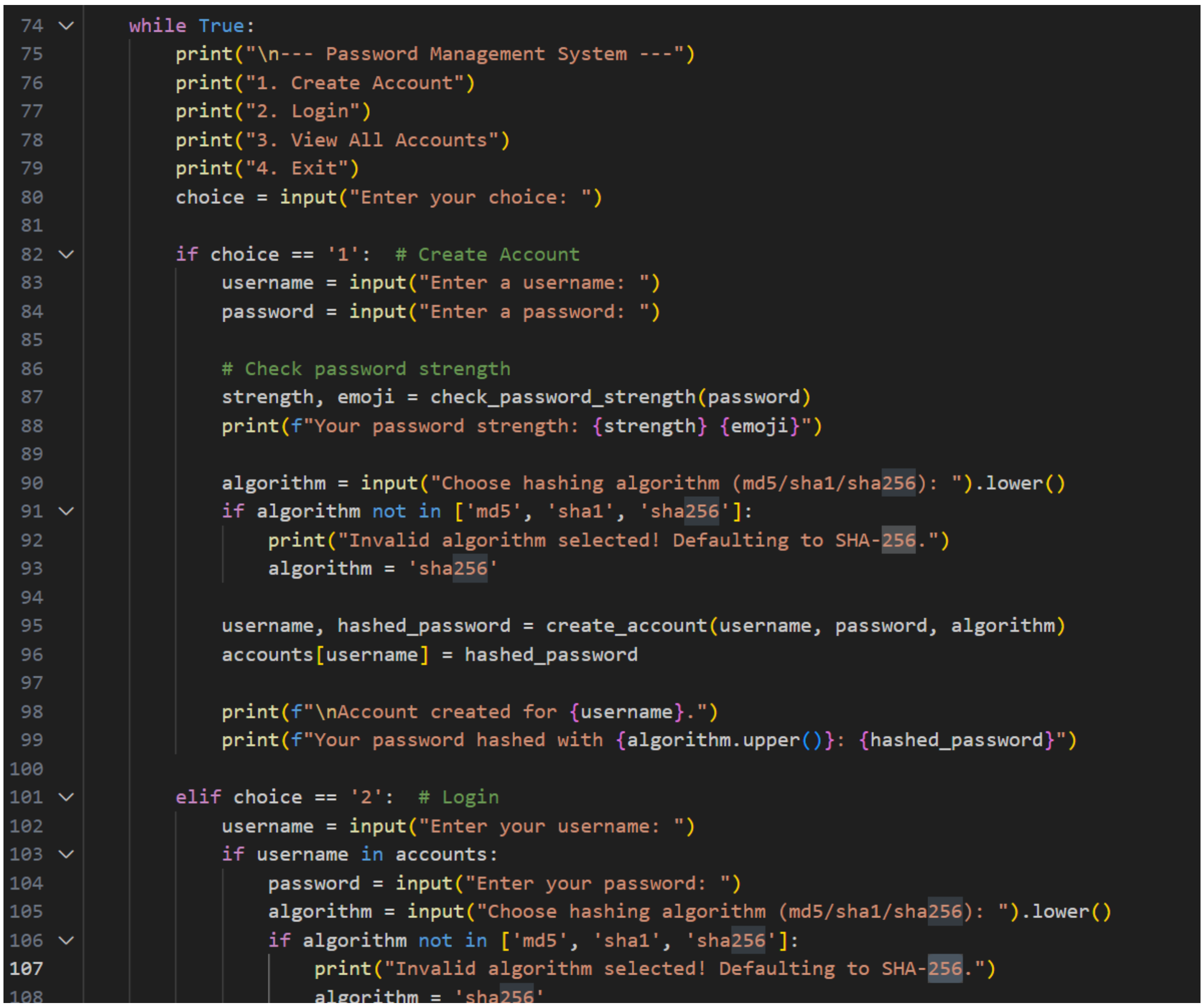
IMPLEMENTATION ON VISUAL STUDIO:



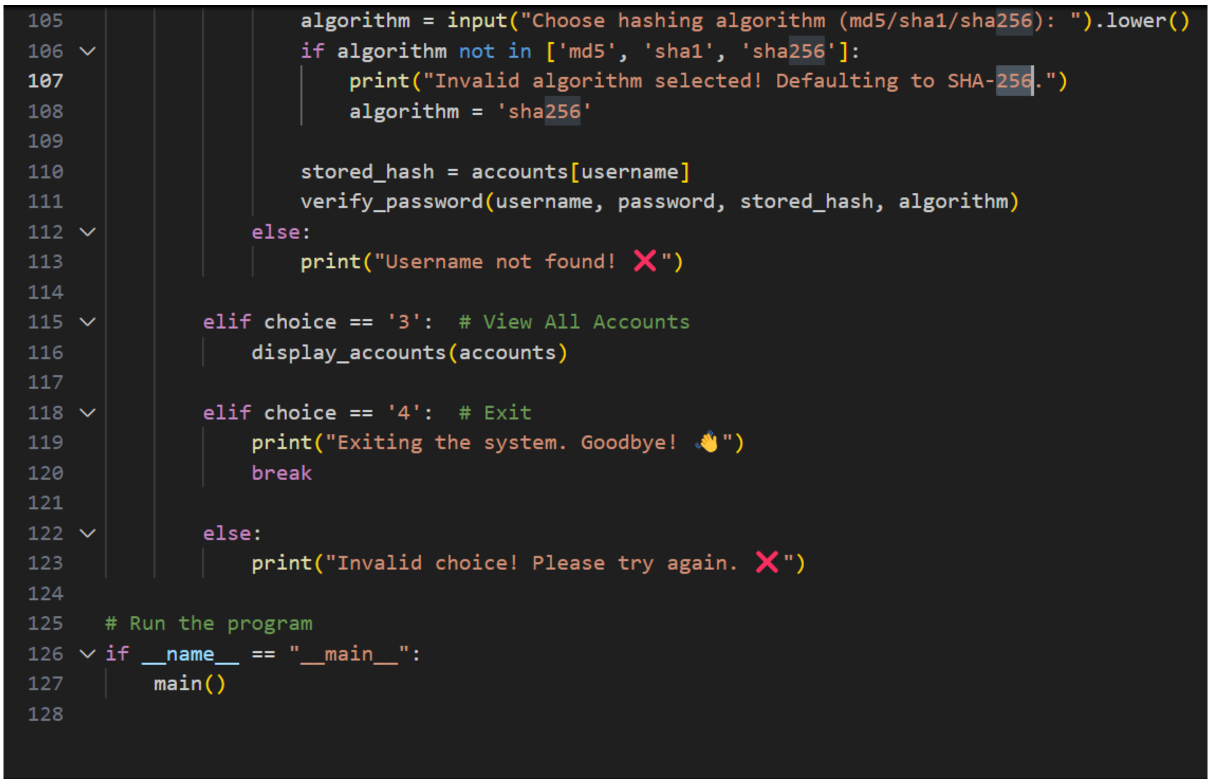
```python
import re
from datetime import datetime

# Hashing functions
def md5_hash(data):
    hash_object = hashlib.md5()
    hash_object.update(data.encode('utf-8'))
    return hash_object.hexdigest()

def sha1_hash(data):
    hash_object = hashlib.sha1()
    hash_object.update(data.encode('utf-8'))
    return hash_object.hexdigest()

def sha256_hash(data):
    hash_object = hashlib.sha256()
    hash_object.update(data.encode('utf-8'))
    return hash_object.hexdigest()

# Function to check password strength
def check_password_strength(password):
    if len(password) < 8:
        return "Weak", "😟"
    elif re.search("[a-z]", password) and re.search("[A-Z]", password) and re.search("[0-9]", password) and re.search("[!@#$%^&*(),.?\":{}
        return "Strong", "🛡️"
    else:
        return "Moderate", "😐"

# Function to create an account (hash the password)
def create_account(username, password, algorithm='sha256'):
    if algorithm == 'md5':
        hashed_password = md5_hash(password)
    elif algorithm == 'sha1':
        hashed_password = sha1_hash(password)
    else:
        hashed_password = sha256_hash(password)
    return username, hashed_password
```



```python
def create_account(username, password, algorithm='sha256'):

# Function to verify password during login
def verify_password(username, input_password, stored_hash, algorithm='sha256'):
    if algorithm == 'md5':
        hashed_input = md5_hash(input_password)
    elif algorithm == 'sha1':
        hashed_input = sha1_hash(input_password)
    else:
        hashed_input = sha256_hash(input_password)

    if hashed_input == stored_hash:
        print("Password verified successfully! 🎉")
        return True
    else:
        print("Incorrect password! ❌")
        return False

# Function to display all accounts and their hashed passwords
def display_accounts(accounts):
    if accounts:
        print("\n--- Registered Accounts ---")
        for username, hashed_password in accounts.items():
            print(f"Username: {username}, Hashed Password: {hashed_password}")
    else:
        print("\nNo accounts registered yet.")

# Main function to run the Password Management System
def main():
    # Display a welcome message
    current_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    print(f"🌟 Welcome to Data Security Using Hash Algorithms 🌟")
    print(f"Current Time: {current_time}\n")

    accounts = {}  # Dictionary to store username:hashed_password pairs

    while True:
```

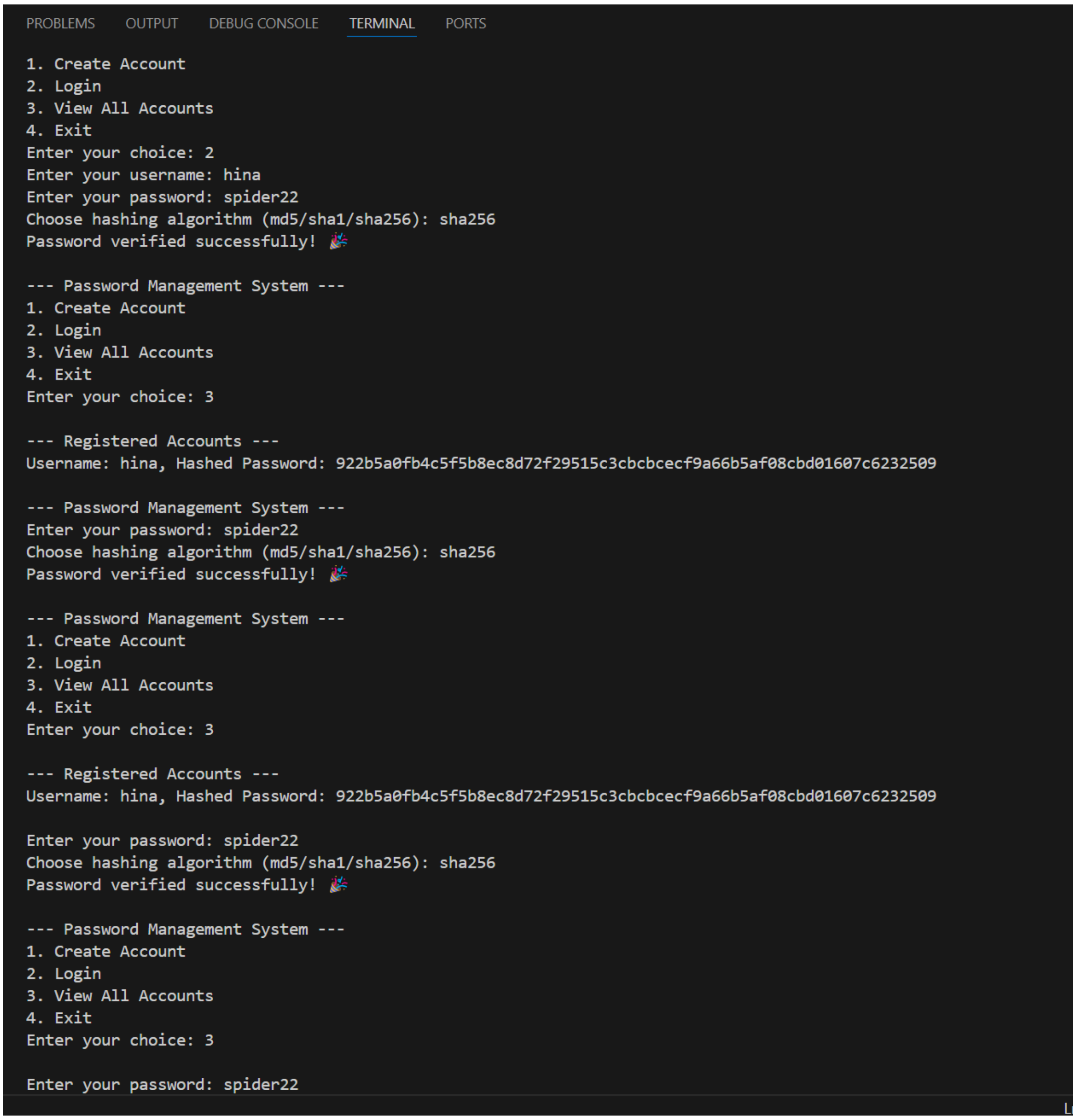


```python
    while True:
        print("\n--- Password Management System ---")
        print("1. Create Account")
        print("2. Login")
        print("3. View All Accounts")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == '1':  # Create Account
            username = input("Enter a username: ")
            password = input("Enter a password: ")

            # Check password strength
            strength, emoji = check_password_strength(password)
            print(f"Your password strength: {strength} {emoji}")

            algorithm = input("Choose hashing algorithm (md5/sha1/sha256): ").lower()
            if algorithm not in ['md5', 'sha1', 'sha256']:
                print("Invalid algorithm selected! Defaulting to SHA-256.")
                algorithm = 'sha256'

            username, hashed_password = create_account(username, password, algorithm)
            accounts[username] = hashed_password

            print(f"\nAccount created for {username}.")
            print(f"Your password hashed with {algorithm.upper()}: {hashed_password}")

        elif choice == '2':  # Login
            username = input("Enter your username: ")
            if username in accounts:
                password = input("Enter your password: ")
                algorithm = input("Choose hashing algorithm (md5/sha1/sha256): ").lower()
                if algorithm not in ['md5', 'sha1', 'sha256']:
                    print("Invalid algorithm selected! Defaulting to SHA-256.")
                    algorithm = 'sha256'
```



```python
                algorithm = input("Choose hashing algorithm (md5/sha1/sha256): ").lower()
                if algorithm not in ['md5', 'sha1', 'sha256']:
                    print("Invalid algorithm selected! Defaulting to SHA-256.")
                    algorithm = 'sha256'

                stored_hash = accounts[username]
                verify_password(username, password, stored_hash, algorithm)
            else:
                print("Username not found! ❌")

        elif choice == '3':  # View All Accounts
            display_accounts(accounts)

        elif choice == '4':  # Exit
            print("Exiting the system. Goodbye! 👋")
            break

        else:
            print("Invalid choice! Please try again. ❌")

# Run the program
if __name__ == "__main__":
    main()
```

OUTPUT:



```
1. Create Account
2. Login
3. View All Accounts
4. Exit
Enter your choice: 2
Enter your username: hina
Enter your password: spider22
Choose hashing algorithm (md5/sha1/sha256): sha256
Password verified successfully! 🎉

--- Password Management System ---
1. Create Account
2. Login
3. View All Accounts
4. Exit
Enter your choice: 3

--- Registered Accounts ---
Username: hina, Hashed Password: 922b5a0fb4c5f5b8ec8d72f29515c3cbcbcecf9a66b5af08cbd01607c6232509

--- Password Management System ---
Enter your password: spider22
Choose hashing algorithm (md5/sha1/sha256): sha256
Password verified successfully! 🎉

--- Password Management System ---
1. Create Account
2. Login
3. View All Accounts
4. Exit
Enter your choice: 3

--- Registered Accounts ---
Username: hina, Hashed Password: 922b5a0fb4c5f5b8ec8d72f29515c3cbcbcecf9a66b5af08cbd01607c6232509

Enter your password: spider22
Choose hashing algorithm (md5/sha1/sha256): sha256
Password verified successfully! 🎉

--- Password Management System ---
1. Create Account
2. Login
3. View All Accounts
4. Exit
Enter your choice: 3

Enter your password: spider22
```

SUMMARY:

- This project demonstrates how to secure user passwords by using cryptographic hash algorithms (MD5, SHA-1, and SHA-256) to ensure that passwords are not stored in plaintext. The program allows users to create accounts by entering a username and password, which is then hashed using the chosen algorithm. The program checks the strength of the password (Weak, Moderate, Strong) based on specific criteria like length and character variety. When users log in, the program hashes the entered password and compares it with the stored hash to verify the user's identity. Additionally, users can view all registered accounts and their hashed passwords. The project educates about data security by applying hash functions and emphasizing the importance of password protection in a secure, practical manner

1. What is Hashing?

Hashing is a process of converting input data into a fixed-size string of characters, which is usually a hexadecimal value. It is commonly used in data security to store passwords securely.

2. What are MD5, SHA-1, and SHA-256?

- **MD5**:
    - Generates a 128-bit hash (32 hexadecimal characters).
    - Considered less secure due to vulnerabilities (e.g., collisions).
- **SHA-1**:
    - Produces a 160-bit hash (40 hexadecimal characters).
    - More secure than MD5 but has known weaknesses.
- **SHA-256**:
    - Part of the SHA-2 family, it generates a 256-bit hash (64 hexadecimal characters).
    - Widely used and considered very secure.

3. Python Libraries Used

- **`hashlib`**:
    - A built-in Python library for hashing.
    - Supports various algorithms like MD5, SHA-1, SHA-256.
    - Functions used:
        - `hashlib.md5()`: Creates an MD5 hash object.
        - `hashlib.sha1()`: Creates a SHA-1 hash object.
        - `hashlib.sha256()`: Creates a SHA-256 hash object.
        - `hash_object.update(data.encode('utf-8'))`: Hashes the given data.
- **`re`**:
    - Used for regular expression operations.
    - Helps validate the strength of a password by checking for:
        - Lowercase letters.
        - Uppercase letters.
        - Digits.
        - Special characters.
- **`datetime`**:
    - A standard Python module to work with dates and times.
    - Used here to display the current time in the welcome message.

4. Key Functions and Their Purpose

a. Hashing Functions

- `md5_hash(data)`:
  - Creates an MD5 hash of the input data.
- `sha1_hash(data)`:
  - Creates a SHA-1 hash of the input data.
- `sha256_hash(data)`:
  - Creates a SHA-256 hash of the input data.

b. `check_password_strength(password)`

- Determines the strength of a password based on:
  - Length (at least 8 characters).
  - Presence of uppercase, lowercase, numbers, and special characters.
- Returns:
  - "Weak" if the password lacks all required characteristics.
  - "Moderate" if it partially satisfies the criteria.
  - "Strong" if it satisfies all criteria.

c. `create_account(username, password, algorithm='sha256')`

- Hashes the password using the specified algorithm (default: SHA-256).
- Stores the hashed password for the provided username.

d. `verify_password(username, input_password, stored_hash, algorithm='sha256')`

- Hashes the input password using the same algorithm as during account creation.
- Compares the new hash with the stored hash to verify if they match.

e. `display_accounts(accounts)`

- Displays all registered accounts and their hashed passwords.

…………………………..