KONGU ENGINEERING COLLEGE (Autonomous)

PERUNDURAI ERODE – 638 060

Hangman Game

A Project Report

submitted by

Dharnish B M

23ITR029

Anishma R S

23ITR004

Anbuchelvan A

23ITR003

PYTHON PROGRAMMING AND FRAMEWORKS ( 22ITT32 )

DEPARTMENT OF INFORMATION TECHNOLOGY

CODING :

import random

import sqlite3

import hashlib

def connect\_db():

    conn = sqlite3.connect('hangman\_scores.db')

    c = conn.cursor()

    c.execute('''CREATE TABLE IF NOT EXISTS users (name TEXT PRIMARY KEY, password TEXT)''')

    c.execute('''CREATE TABLE IF NOT EXISTS scores (name TEXT, score INTEGER)''')

    conn.commit()

    return conn, c

def hash\_password(password):

    return hashlib.sha256(password.encode()).hexdigest()

def signup(conn, c):

    while True:

        name = input("Enter a username: ").strip().lower()

        password = input("Enter a password: ").strip()

        c.execute("SELECT \* FROM users WHERE name=?", (name,))

        if c.fetchone():

            print("Username already exists. Try another one.")

        else:

            hashed\_password = hash\_password(password)

            c.execute("INSERT INTO users (name, password) VALUES (?, ?)", (name, hashed\_password))

            conn.commit()

            print("Signup successful!")

            return name  # Proceed to game

def login(conn, c):

    while True:

        name = input("Enter your username: ").strip().lower()

        password = input("Enter your password: ").strip()

        c.execute("SELECT password FROM users WHERE name=?", (name,))

        result = c.fetchone()

        if result and result[0] == hash\_password(password):

            print(f"Welcome back, {name}!")

            return name

        else:

            print("Invalid username or password. Please try again.")

def update\_scores(conn, c, name, score):

    c.execute("INSERT INTO scores (name, score) VALUES (?, ?)", (name, score))

    conn.commit()

def restart\_game(conn, c, user):

    while True:

        restart = input("Do you want to restart the game? (yes/no): ").lower()

        if restart == "yes":

            start\_game(conn, c, user)

            break

        elif restart == "no":

            print("Thanks for playing! Goodbye.")

            break

        else:

            print("Invalid input. Please enter 'yes' or 'no'.")

def choose\_difficulty():

    while True:

        difficulty = input("Choose your difficulty level (easy, medium, hard): ").lower()

        if difficulty == "easy":

            return easy\_words, 8

        elif difficulty == "medium":

            return medium\_words, 6

        elif difficulty == "hard":

            return hard\_words, 4

        else:

            print("Invalid input. Please choose easy, medium, or hard.")

def start\_game(conn, c, user):

    word\_list, max\_attempts = choose\_difficulty()

    chosen\_word = list(random.choice(word\_list))

    blank\_list = ['\_'] \* len(chosen\_word)

    attempts = 0

    hints\_used = False

    print("Starting Hangman!")

    while attempts < max\_attempts:

        print(HANGMANPICS[attempts])

        print(f"Word: {' '.join(blank\_list)}")

        guess = input("Make a guess or type 'hint' for a hint: ").lower()

        if guess == "hint" and not hints\_used:

            for i, letter in enumerate(chosen\_word):

                if blank\_list[i] == '\_':

                    blank\_list[i] = letter

                    hints\_used = True

                    break

        elif guess in chosen\_word:

            for i, letter in enumerate(chosen\_word):

                if letter == guess:

                    blank\_list[i] = letter

        else:

            attempts += 1

            print(f"Wrong guess! Attempts left: {max\_attempts - attempts}")

        if "\_" not in blank\_list:

            print("YOU WIN!")

            update\_scores(conn, c, user, max\_attempts - attempts)

            restart\_game(conn, c, user)

            return

    print(HANGMANPICS[attempts])

    print(f"GAME OVER. The word was: {''.join(chosen\_word)}")

    restart\_game(conn, c, user)

HANGMANPICS = ['''

  +---+

  |   |

      |

      |

      |

      |

=========''', '''

  +---+

  |   |

  O   |

      |

      |

      |

=========''', '''

  +---+

  |   |

  O   |

  |   |

      |

      |

=========''', '''

  +---+

  |   |

  O   |

 /|   |

      |

      |

=========''', '''

  +---+

  |   |

  O   |

 /|\  |

      |

      |

=========''', '''

  +---+

  |   |

  O   |

 /|\  |

 /    |

      |

=========''', '''

  +---+

  |   |

  O   |

 /|\  |

 / \  |

      |

=========''']

easy\_words = ["cat", "dog", "fish", "tree", "bird", "ball", "bag", "home", "light", "word", "watch"]

medium\_words = ["python", "mountain", "flower", "castle", "planet", "animal", "shift", "google", "laptop"]

hard\_words = ["astronaut", "philosophy", "hangman", "amphibious", "transcendent", "processor", "codetantra"]

if \_\_name\_\_ == "\_\_main\_\_":

    conn, c = connect\_db()

    print("Welcome to Hangman!")

    logged\_in\_user = None

    while not logged\_in\_user:

        choice = input("Do you want to (1) Sign Up or (2) Log In? Enter 1 or 2: ")

        if choice == "1":

            logged\_in\_user = signup(conn, c)

        elif choice == "2":

            logged\_in\_user = login(conn, c)

        else:

            print("Invalid input. Please enter 1 or 2.")

    start\_game(conn, c, logged\_in\_user)

    conn.close()

OUTPUT :





