```
import random
import sqlite3
from tkinter import Tk, Label, Button, Entry, StringVar, END

# Hangman Core Logic
def choose_word():
    words = ["python", "hangman", "developer", "software", "algorithm"]
    return random.choice(words)

def display_word(word, guessed_letters):
    return " ".join([letter if letter in guessed_letters else "_" for letter in word])

class HangmanGame:
    def __init__(self):
        self.word = choose_word()
        self.guessed_letters = set()
        self.remaining_attempts = 6

    def guess(self, letter):
        if letter in self.word and letter not in self.guessed_letters:
            self.guessed_letters.add(letter)
            return True
        elif letter not in self.word:
            self.remaining_attempts -= 1
            return False
        return None

    def is_won(self):
        return all(letter in self.guessed_letters for letter in self.word)

    def is_lost(self):
        return self.remaining_attempts <= 0

# SQLite Integration (Optional)
def setup_database():
    conn = sqlite3.connect("hangman_history.db")
    cursor = conn.cursor()
    cursor.execute("CREATE TABLE IF NOT EXISTS history (id INTEGER PRIMARY KEY, word TEXT, attempts INTEGER, status TEXT)")
    conn.commit()
    conn.close()

def save_game_to_database(word, attempts, status):
    conn = sqlite3.connect("hangman_history.db")
    cursor = conn.cursor()
    cursor.execute("INSERT INTO history (word, attempts, status) VALUES (?, ?, ?)", (word, attempts, status))
    conn.commit()
    conn.close()

# Console UI (Fallback)
def console_ui():
    game = HangmanGame()
    print("Welcome to Hangman!")
    while not game.is_won() and not game.is_lost():
        print("\n" + display_word(game.word, game.guessed_letters))
        print(f"Remaining attempts: {game.remaining_attempts}")
        guess = input("Enter a letter: ").lower()
        if len(guess) != 1 or not guess.isalpha():
            print("Invalid input. Please enter a single letter.")
            continue
        result = game.guess(guess)
        if result is True:
            print(f"Good job! '{guess}' is in the word.")
        elif result is False:
            print(f"Sorry, '{guess}' is not in the word.")
        else:
            print(f"You've already guessed '{guess}'.")

    if game.is_won():
        print(f"Congratulations! You guessed the word: {game.word}")
        save_game_to_database(game.word, game.remaining_attempts, "Won")
    else:
        print(f"Game over! The word was: {game.word}")
        save_game_to_database(game.word, game.remaining_attempts, "Lost")

# GUI (Optional with Tkinter)
class HangmanGUI:
    def __init__(self, root):
        self.game = HangmanGame()

        self.word_var = StringVar()
        self.word_var.set(display_word(self.game.word, self.game.guessed_letters))
```

```
            self.attempts_var = StringVar()
            self.attempts_var.set(f"Attempts left: {self.game.remaining_attempts}")

            self.word_label = Label(root, textvariable=self.word_var, font=("Arial", 18))
            self.word_label.pack()

            self.attempts_label = Label(root, textvariable=self.attempts_var, font=("Arial", 12))
            self.attempts_label.pack()

            self.input_var = StringVar()
            self.input_entry = Entry(root, textvariable=self.input_var, font=("Arial", 14))
            self.input_entry.pack()

            self.submit_button = Button(root, text="Submit", command=self.submit_guess)
            self.submit_button.pack()

            self.message_label = Label(root, text="", font=("Arial", 12))
            self.message_label.pack()

        def submit_guess(self):
            guess = self.input_var.get().lower()
            self.input_var.set("")

            if len(guess) != 1 or not guess.isalpha():
                self.message_label.config(text="Please enter a valid single letter.")
                return

            result = self.game.guess(guess)
            if result is True:
                self.message_label.config(text=f"Good job! '{guess}' is in the word.")
            elif result is False:
                self.message_label.config(text=f"'{guess}' is not in the word.")
            else:
                self.message_label.config(text=f"You've already guessed '{guess}'.")

            self.word_var.set(display_word(self.game.word, self.game.guessed_letters))
            self.attempts_var.set(f"Attempts left: {self.game.remaining_attempts}")

            if self.game.is_won():
                self.message_label.config(text=f"Congratulations! You guessed the word: {self.game.word}")
                save_game_to_database(self.game.word, self.game.remaining_attempts, "Won")
                self.disable_inputs()
            elif self.game.is_lost():
                self.message_label.config(text=f"Game over! The word was: {self.game.word}")
                save_game_to_database(self.game.word, self.game.remaining_attempts, "Lost")
                self.disable_inputs()

        def disable_inputs(self):
            self.input_entry.config(state="disabled")
            self.submit_button.config(state="disabled")

    if __name__ == "__main__":
        setup_database()
        mode = input("Choose mode: console or gui (default: console): ").strip().lower()
        if mode == "gui":
            root = Tk()
            root.title("Hangman Game")
            app = HangmanGUI(root)
            root.mainloop()
        else:
            console_ui()
```