Advanced stop watch

```
# Advanced Stopwatch in Python with Tkinter
import tkinter as tk
from tkinter import ttk, messagebox
import time, csv, sqlite3, math

# Database setup for sessions
conn = sqlite3.connect("stopwatch_sessions.db")
c = conn.cursor()
c.execute('''CREATE TABLE IF NOT EXISTS sessions
             (id INTEGER PRIMARY KEY, mode TEXT, duration TEXT, laps TEXT, date TEXT)''')
conn.commit()

class StopwatchApp:
    def _init_(self, root):
        self.root = root
        self.root.title("Advanced Stopwatch")
        self.root.geometry("500x400")
        self.is_running = False
        self.start_time = 0
        self.elapsed = 0
        self.laps = []
        self.mode = "Stopwatch"  # or "Countdown"
        self.countdown_from = 300  # default 5 minutes
        self.theme = "light"

        self.build_ui()
        self.update_timer()

        self.root.bind('<space>', lambda e: self.toggle_start())
        self.root.bind('<l>', lambda e: self.lap())
        self.root.bind('<r>', lambda e: self.reset())

    def build_ui(self):
        self.timer_label = ttk.Label(self.root, text="00:00:00", font=("Helvetica", 40))
        self.timer_label.pack(pady=10)

        self.lap_box = tk.Listbox(self.root, height=5)
        self.lap_box.pack(pady=5)

        btn_frame = ttk.Frame(self.root)
        btn_frame.pack()
        ttk.Button(btn_frame, text="Start/Stop", command=self.toggle_start).grid(row=0, column=0,
padx=5)
        ttk.Button(btn_frame, text="Lap", command=self.lap).grid(row=0, column=1, padx=5)
        ttk.Button(btn_frame, text="Reset", command=self.reset).grid(row=0, column=2, padx=5)
        ttk.Button(btn_frame, text="Save", command=self.save_session).grid(row=0, column=3, padx=5)

        mode_frame = ttk.Frame(self.root)
        mode_frame.pack(pady=5)
        ttk.Button(mode_frame, text="Toggle Mode", command=self.toggle_mode).grid(row=0,
column=0, padx=5)
```

```
        ttk.Button(mode_frame, text="Theme", command=self.toggle_theme).grid(row=0, column=1,
padx=5)

        # Canvas for analog view
        self.canvas = tk.Canvas(self.root, width=200, height=200, bg="white")
        self.canvas.pack(pady=10)

    def update_timer(self):
        if self.is_running:
            if self.mode == "Stopwatch":
                self.elapsed = time.time() - self.start_time
            elif self.mode == "Countdown":
                self.elapsed = self.countdown_from - (time.time() - self.start_time)
                if self.elapsed <= 0:
                    self.elapsed = 0
                    self.is_running = False
                    messagebox.showinfo("Time Up", "Countdown finished")

        self.display_time()
        self.draw_analog_clock()
        self.root.after(100, self.update_timer)

    def display_time(self):
        mins, secs = divmod(int(self.elapsed), 60)
        hours, mins = divmod(mins, 60)
        self.timer_label.config(text=f"{hours:02}:{mins:02}:{secs:02}")

    def draw_analog_clock(self):
        self.canvas.delete("all")
        x, y, r = 100, 100, 90
        self.canvas.create_oval(x-r, y-r, x+r, y+r)
        angle = (self.elapsed % 60) * 6
        rad = math.radians(angle)
        x2 = x + 70 * math.sin(rad)
        y2 = y - 70 * math.cos(rad)
        self.canvas.create_line(x, y, x2, y2, width=3, fill="blue")

    def toggle_start(self):
        if not self.is_running:
            self.start_time = time.time() - self.elapsed if self.mode == "Stopwatch" else time.time()
            self.is_running = True
        else:
            self.is_running = False

    def reset(self):
        self.is_running = False
        self.elapsed = 0
        self.laps = []
        self.lap_box.delete(0, tk.END)

    def lap(self):
        if self.is_running and self.mode == "Stopwatch":
            lap_time = time.strftime('%H:%M:%S', time.gmtime(self.elapsed))
            self.laps.append(lap_time)
            self.lap_box.insert(tk.END, f"Lap {len(self.laps)}: {lap_time}")

    def toggle_mode(self):
        self.mode = "Countdown" if self.mode == "Stopwatch" else "Stopwatch"
        self.reset()

    def toggle_theme(self):
        self.theme = "dark" if self.theme == "light" else "light"
        bg = "black" if self.theme == "dark" else "white"
        fg = "white" if self.theme == "dark" else "black"
        self.root.configure(bg=bg)
        self.canvas.configure(bg=bg)
        self.timer_label.configure(background=bg, foreground=fg)

    def save_session(self):
        date = time.strftime("%Y-%m-%d %H:%M:%S")
        duration = time.strftime('%H:%M:%S', time.gmtime(self.elapsed))
        laps = ", ".join(self.laps)
        with open("stopwatch_sessions.csv", "a", newline="") as f:
            writer = csv.writer(f)
            writer.writerow([self.mode, duration, laps, date])
        c.execute("INSERT INTO sessions (mode, duration, laps, date) VALUES (?, ?, ?, ?)",
                  (self.mode, duration, laps, date))
        conn.commit()
        messagebox.showinfo("Saved", "Session saved to CSV and database.")

root = tk.Tk()
app = StopwatchApp(root)
root.mainloop()
```