TIMER USING PYTHON

PYHON PROGRAM THAT FUNCTIONS AS A TIMER .THIS TIMER HAVE THE FOLLOWING FEATURES.

1. COUNTDOWN TIMER
2. STOPWATCH
3. 3. USER INERFACE
4. ACCURACY’

CODE :

```
import time
import threading
import tkinter as tk

class Timer:
    def __init__(self, master):
        self.master = master
        master.title("Timer")

        self.is_running = False
        self.start_time = 0
        self.elapsed_time = 0

        self.label = tk.Label(master, font=("Helvetica", 48))
        self.label.pack(pady=20)

        self.start_button = tk.Button(master, text="Start",
command=self.start)
        self.start_button.pack(pady=10)

        self.stop_button = tk.Button(master, text="Stop",
command=self.stop)
        self.stop_button.pack(pady=10)

        self.reset_button = tk.Button(master, text="Reset",
command=self.reset)
        self.reset_button.pack(pady=10)

        self.countdown_entry = tk.Entry(master)
        self.countdown_entry.pack(pady=10)

        self.countdown_button = tk.Button(master, text="Start
Countdown", command=self.start_countdown)
        self.countdown_button.pack(pady=10)

        self.update_time()
```

```
    def start(self):
        if not self.is_running:
            self.is_running = True
            self.start_time = time.time() - self.elapsed_time

    def stop(self):
        if self.is_running:
            self.is_running = False
            self.elapsed_time = time.time() - self.start_time

    def reset(self):
        self.is_running = False
        self.start_time = 0
        self.elapsed_time = 0
        self.label.config(text="00:00:00")

    def start_countdown(self):
        try:
            self.countdown_time = int(self.countdown_entry.get())
            self.is_running = True
            self.countdown_thread =
threading.Thread(target=self.countdown)
            self.countdown_thread.start()
        except ValueError:
            pass

    def countdown(self):
        while self.countdown_time > 0 and self.is_running:
            self.countdown_time -= 1
            self.label.config(text=f"{self.countdown_time}")
            time.sleep(1)
        self.is_running = False
        if self.countdown_time == 0:
            self.label.config(text="Time's up!")

    def update_time(self):
        if self.is_running:
            current_time = time.time() - self.start_time
            self.label.config(text=self.format_time(current_time))
        self.label.after(10, self.update_time)

    def format_time(self, time_in_seconds):
        hours = int(time_in_seconds // 3600)
        minutes = int((time_in_seconds % 3600) // 60)
        seconds = int(time_in_seconds % 60)
```

```
        return f"{hours:02d}:{minutes:02d}:{seconds:02d}"

root = tk.Tk()
timer = Timer(root)
root.mainloop()
```