

Python Minor Project

Problem Statement- Create A Countdown Timer Using Python

Features To Include

Reset/ Stop

Pause /Resume

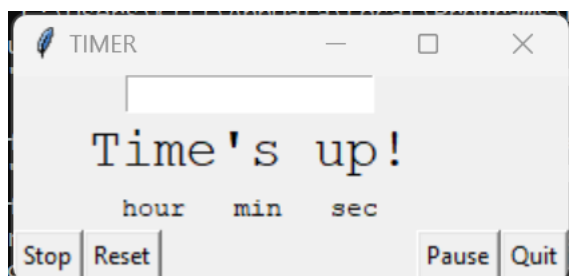
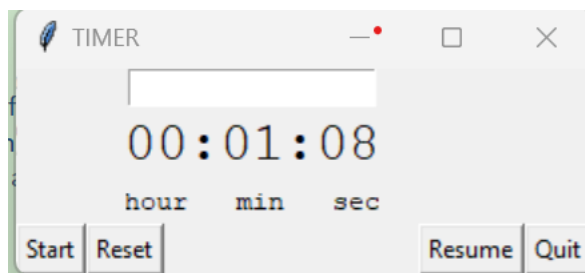
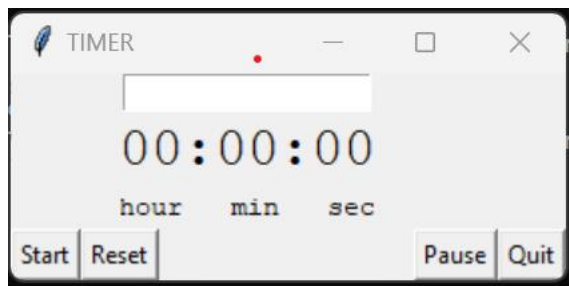
I Arbaz Hasan Roll no. 2004260 of ETC Branch KIIT University I have completed my ML Project (Create A Countdown Timer Using Python) Using Tkinter Library in Python with features Reset/Stop and Pause/Resume.

Software required-

Python 3.9

Visual studio

Outputs-



Codes-

```
import tkinter as tk
import tkinter.messagebox
import time
```

```
class Application(tk.Frame):
    def __init__(self, master, *args, **kwargs):
        tk.Frame.__init__(self, master, *args, **kwargs)
        self.master = master
        self.running = False
        self.time = 0
        self.hours = 0
        self.mins = 0
        self.secs = 0
        self.build_interface()

    def build_interface(self):
        self.time_entry = tk.Entry(self)
        self.time_entry.grid(row=0, column=1)

        self.clock = tk.Label(self, text="00:00:00", font=("Courier", 20), width=10)
        self.clock.grid(row=1, column=1, sticky="S")

        self.time_label = tk.Label(self, text="hour  min  sec", font=("Courier", 10),
width=15)
        self.time_label.grid(row=2, column=1, sticky="N")

        self.power_button = tk.Button(self, text="Start", command=lambda: self.start())
        self.power_button.grid(row=3, column=0, sticky="NE")

        self.reset_button = tk.Button(self, text="Reset", command=lambda: self.reset())
        self.reset_button.grid(row=3, column=1, sticky="NW")

        self.quit_button = tk.Button(self, text="Quit", command=lambda: self.quit())
        self.quit_button.grid(row=3, column=3, sticky="NE")

        self.pause_button = tk.Button(self, text="Pause", command=lambda:
self.pause())
        self.pause_button.grid(row = 3,column=2, sticky = "NW")

        self.master.bind("<Return>", lambda x: self.start())
        self.time_entry.bind("<Key>", lambda v: self.update())
```

```

def calculate(self):
    """time calculation"""
    self.hours = self.time // 3600
    self.mins = (self.time // 60) % 60
    self.secs = self.time % 60
    return "{:02d}:{:02d}:{:02d}".format(self.hours, self.mins, self.secs)

def update(self):
    """validation"""
    self.time = int(self.time_entry.get())
    try:
        self.clock.configure(text=self.calculate())
    except:
        self.clock.configure(text="00:00:00")

def timer(self):
    """display time"""
    if self.running:
        if self.time <= 0:
            self.clock.configure(text="Time's up!")
        else:
            self.clock.configure(text=self.calculate())
            self.time -= 1
            self.after(1000, self.timer)

def start(self):
    """start timer"""
    try:
        self.time = int(self.time_entry.get())
        self.time_entry.delete(0, 'end')
    except:
        self.time = self.time
    self.power_button.configure(text="Stop", command=lambda: self.stop())
    self.master.bind("<Return>", lambda x: self.stop())
    self.running = True
    self.timer()

def stop(self):
    """Stop timer"""
    self.power_button.configure(text="Start", command=lambda: self.start())
    self.master.bind("<Return>", lambda x: self.start())
    self.running = False

def reset(self):
    """Resets the timer to 0."""
    self.power_button.configure(text="Start", command=lambda: self.start())

```

```

self.master.bind("<Return>", lambda x: self.start())
self.running = False
self.time = 0
self.clock["text"] = "00:00:00"

def quit(self):
    """quit the window"""
    if tk.messagebox.askokcancel("Quit", "Do you want to quit?"):
        root.destroy()

def pause(self):
    """Pause timer"""
    self.pause_button.configure(text="Resume", command=lambda: self.resume())
    self.master.bind("<Return>", lambda x: self.resume())
    if self.running == True:
        self.running = False
    self.timer()

def resume(self):
    """Resume timer"""
    self.pause_button.configure(text="Pause", command=lambda: self.pause())
    self.master.bind("<Return>", lambda x: self.pause())
    if self.running == False:
        self.running = True
    self.timer()

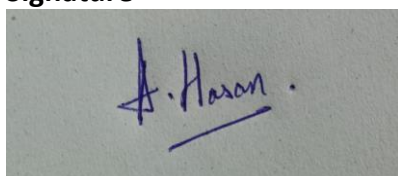
if __name__ == "__main__":
    """Main loop of timer"""
    root = tk.Tk()
    root.title("TIMER")
    Application(root).pack(side="top", fill="both", expand=True)
    root.mainloop()

```

Conclusion-

Using Tkinter Library in Python with features Reset/Stop and Pause/Resume completed this project

Signature-



A. Hasan

