```
"""
This file is a heavily modified version of https://github.com/PySimpleGUI/PySimpleGUI/blob/master/DemoPrograms/Demo_Desktop_Widget_Timer.py

The original code is licensed under the GNU Lesser General Public License v3.0 (https://github.com/PySimpleGUI/PySimpleGUI/blob/master/license.txt).
This means that we can use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the original code, as long as we include the original License and copyright notice, Disclose source, State changes, and the project is licensed under the Same license.
"""

import _thread
import asyncio
import split_watcher as sw
import PySimpleGUI as sg
import time


user_times = []
splitNumber = 0

def time_as_int():
    return int(round(time.time() * 100))
def format_time():
    return window['-TIMER-TEXT-'].update('{:02d}:{:02d}.{:02d}'.format((current_time // 100) // 60,
                                                                        (current_time // 100) % 60,
                                                                        current_time % 100))

def callOnSplit():
  user_times.append(current_time)
  currentF = open('current_run.csv', 'a+')
#   currentF.writelines(str(splitNumber) + ", " + str(user_times[-1]) + "\n")
  currentF.writelines(str(splitNumber) + ", " + str(current_time) + "\n")

  currentF.close()

def endSplitWatch():
  print("endSplitWatch")
#   splitWatch.exit_program = True
#   splitWatchThread.join()
  exit()

# ----------------  Create Form  ----------------
sg.theme('Black')

layout = [[sg.Text('')],
          [sg.Text('', size=(8, 2), font=('Helvetica', 20),
                justification='center', key='-TIMER-TEXT-'),
           sg.Text('', size=(8, 2), font=('Helvetica', 20),
                justification='center', key='-SPLIT-TEXT-')],
          [sg.Button('Pause', key='-RUN-PAUSE-', button_color=('white', '#001480')),
           sg.Button('Reset', button_color=('white', '#007339'), key='-RESET-'),
           sg.Exit(button_color=('white', 'firebrick4'), key='Exit'),
```

```
50            sg.Button('Split', key='-SPLIT-TIMER-', button_color=('white',
'#ff0000'))]]
51
52 window = sg.Window('Running Timer', layout,
53                     no_titlebar=False,
54                     auto_size_buttons=True,
55                     keep_on_top=True,
56                     grab_anywhere=True,
57                     element_padding=(0, 0),
58                     finalize=True,
59                     element_justification='c',
60                     right_click_menu=sg.MENU_RIGHT_CLICK_EDITME_EXIT,
61                     size = (500, 500),
62                     resizable = True)
63
64 splitWatch = sw.splitWatcher(callOnSplit, endSplitWatch, window)
65
66 def startSplitWatch():
67   asyncio.run(splitWatch.start())
68
69 _thread.start_new_thread(startSplitWatch, ())
70
71 current_time, paused_time, paused = 0, 0, True
72 start_time = time_as_int()
73
74 def format_time():
75     return window['-TIMER-TEXT-'].update('{:02d}:{:02d}.{:02d}'.format((current_time
// 100) // 60,
76                                                         (current_time // 100) % 60,
77                                                         current_time % 100))
78
79 def doEventWindows(event):
80     global paused_time
81     global current_time
82     global paused
83     global start_time
84     global splitNumber
85
86     match event:
87         case sg.WIN_CLOSED | 'Exit':
88             splitWatch.exit_program = True
89             del splitWatch
90             print('Exit')
91             return True
92         case '-RESET-':
93             paused_time = start_time = time_as_int()
94             current_time = 0
95             splitNumber = 0
96             window['-SPLIT-TEXT-'].update('{:02d}:{:02d}.
{:02d}'.format((current_time // 100) // 60,
97                                                                 (current_time // 100) %
60,
98                                                                 current_time % 100))
99             format_time()
100            print('Reset')
101            if not paused:
```

```
                doEventWindows('-RUN-PAUSE-')
            return False
        case '-RUN-PAUSE-':
            paused = not paused
            if paused:
                paused_time = time_as_int()
            else:
                start_time = start_time + time_as_int() - paused_time
            print('Run or Pause')
            return False
        case '-SPLIT-TIMER-':
            if (splitNumber == 0 and paused):
                doEventWindows('-RUN-PAUSE-')
                return False
            elif (paused):
                return False
            else:
                callOnSplit()
                splitNumber += 1
                window['-SPLIT-TEXT-'].update('{:02d}:{:02d}.
{:02d}'.format((current_time // 100) // 60,
                                                        (current_time // 100) %
60,
                                                        current_time % 100))
                # Change button's text
                window['-RUN-PAUSE-'].update('Run' if paused else 'Pause')
                print('Split')
            return False

def main():
  global paused_time
  global current_time
  global paused
  global start_time
  global splitNumber
  doEventWindows('-RESET-')

  while True:
  # --------- Read and update window --------
    if not paused:
        event, values = window.read(timeout=10)
        current_time = time_as_int() - start_time
        # print(event, values)
    else:
        event, values = window.read()
        print(event, values)
    # --------- Do Button Operations --------

    if doEventWindows(event):
        break

    format_time()

if __name__ == "__main__":
  main()
```