**Project Report**

**Title:** Simple Stop Watch

**Submitted by:** Ishan.Umarjikar

**Student Roll No:** 61

**Student EID:** 2303111003  
**Submitted to:** Mr. R. S. Khamitkar  
**Department:** Electronics and Computer Engineering  
**Date:** 17/04/25

**1. Project Overview**

Simple Stopwatch with Elapsed Time is a lightweight desktop application made using Python and PyQt5 libary. It allows us to start, stop, and reset a stopwatch with millisecond precision. The app features a good GUI and displays the total elapsed time in a popup when stopped, making it best for basic time tracking tasks like study sessions, workouts, or productivity challenges.

**2. Problem Statement**

There is a need for a minimal, easy-to-use desktop stopwatch that allows users to track time and perform basic time control operations (start, stop, reset), and view the total elapsed time after stopping the stopwatch.. This project solves that by offering a clean, responsive stopwatch built with Python and PyQt5, making it ideal for timing tasks like workouts, study sessions, or productivity sprints.

**3. Technology Stack**

* **Programming Language:** Python 3.x
* **Libraries Used**: PyQt5,sys
* **IDE:** Visual Studio Code(VScode)

**4. Implementation**

The implementation follows these steps:

* **Time Tracking**:
* The stopwatch starts at 00:00:00.00 and keeps updating every 10 milliseconds.
* The time is managed by QTime and shown in the format HH:MM:SS.MS.
* **UI (User Interface)**:
* A large **display** shows the current time.
* Three **buttons** are provided:
  1. **Start**: Starts the stopwatch and begins updating the time.
  2. **Stop**: Pauses the stopwatch and shows a pop-up with the final elapsed time.
  3. **Reset**: Resets the time to 00:00:00.00 so you can start over.
* **How It Works**:
* When you click **Start**, the stopwatch begins, and the time updates every 10 milliseconds.
* When you press **Stop**, the timer stops, and a message box shows how much time has passed.
* Pressing **Reset** clears the time and sets it back to zero.

**5. Code Snippet**

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QLabel, QPushButton, QVBoxLayout, QHBoxLayout, QMessageBox

from PyQt5.QtCore import QTimer, QTime, Qt

*class* SimpleStopwatch(QWidget):

*def* \_\_init\_\_(*self*):

        super().\_\_init\_\_()

*self*.setWindowTitle("Simple Stopwatch")

        #this is the  Stopwatch logic

*self*.time = QTime(0, 0, 0, 0)

*self*.timer = QTimer()

        # the is ui Elements

*self*.label = QLabel("00:00:00.00")

*self*.label.setAlignment(Qt.AlignCenter)

*self*.label.setStyleSheet("font-size: 60px; background-color: lightblue; padding: 20px; border-radius: 10px;")

*self*.start\_btn = QPushButton("Start")

*self*.stop\_btn = QPushButton("Stop")

*self*.reset\_btn = QPushButton("Reset"

        # Connect buttons to methods

*self*.start\_btn.clicked.connect(*self*.start)

*self*.stop\_btn.clicked.connect(*self*.stop)

*self*.reset\_btn.clicked.connect(*self*.reset)

*self*.timer.timeout.connect(*self*.update\_time)

        # Layout

        layout = QVBoxLayout()

        layout.addWidget(*self*.label)

        btn\_layout = QHBoxLayout()

        btn\_layout.addWidget(*self*.start\_btn)

        btn\_layout.addWidget(*self*.stop\_btn)

        btn\_layout.addWidget(*self*.reset\_btn)

        layout.addLayout(btn\_layout)

*self*.setLayout(layout)

*def* start(*self*):

*self*.timer.start(10)

*def* stop(*self*):

*self*.timer.stop()

*self*.show\_elapsed\_time()  # this shows  the final time when stopped

*def* reset(*self*):

*self*.timer.stop()

*self*.time = QTime(0, 0, 0, 0)

*self*.label.setText("00:00:00.00")

*def* update\_time(*self*):

*self*.time = *self*.time.addMSecs(10)

        h, m, s, ms = *self*.time.hour(), *self*.time.minute(), *self*.time.second(), *self*.time.msec() // 10

*self*.label.setText(*f*"{h*:02*}:{m*:02*}:{s*:02*}.{ms*:02*}")

*def* show\_elapsed\_time(*self*):

        h = *self*.time.hour()

        m = *self*.time.minute()

        s = *self*.time.second()

        ms = *self*.time.msec() // 10

        elapsed = *f*"Elapsed Time: {h*:02*}:{m*:02*}:{s*:02*}.{ms*:02*}"

        QMessageBox.information(*self*, "Time Info", elapsed)

if \_\_name\_\_ == "\_\_main\_\_":

    app = QApplication(sys.argv)

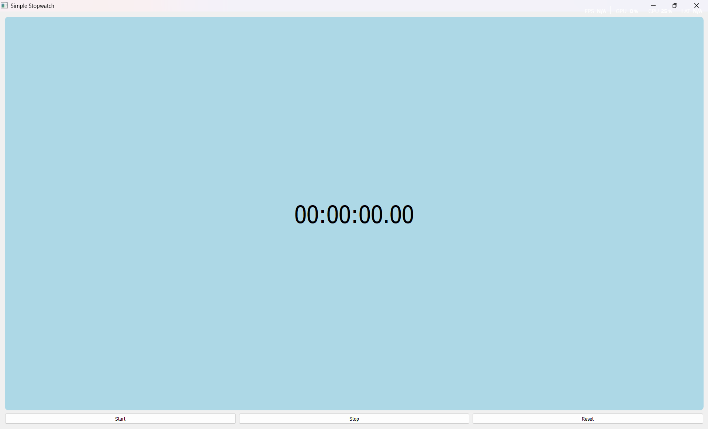
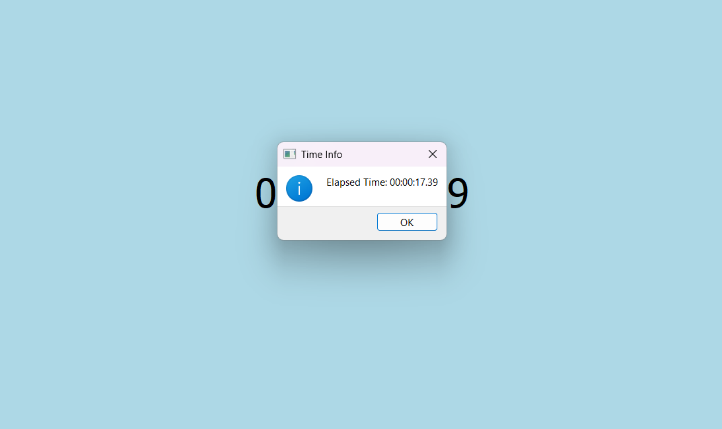
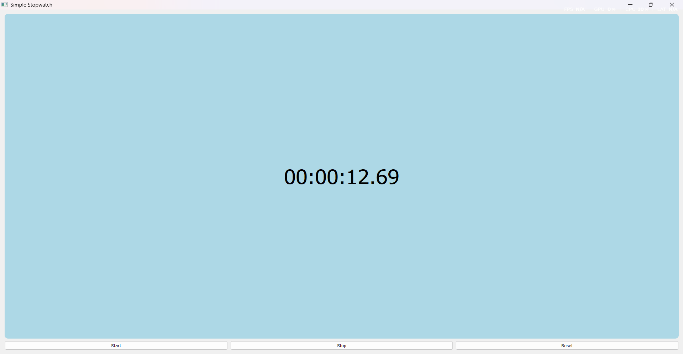
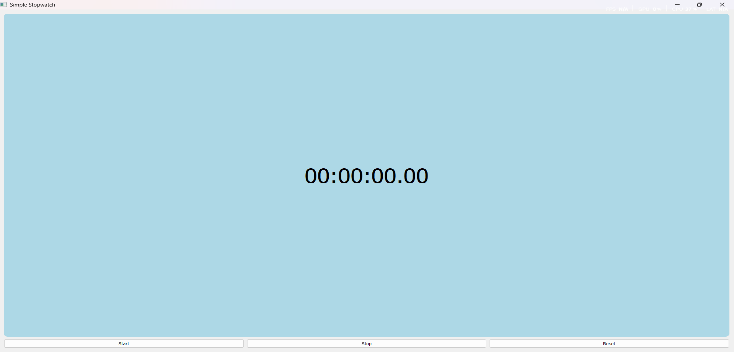
    window = SimpleStopwatch()

    window.show()

    sys.exit(app.exec\_())

**6. Sample Test Cases**

| Input  Idle no actions | Output  Started the stopwatch |
| --- | --- |



Reset the stop watch

StopWatch is stopped and time is also Elapsed

**7. Challenges Faced**

The main challenges were understanding how to use PyQt5 components, making the timer work in sync, and ensuring buttons interacted correctly without any problems during the working of the code. Designing a clean UI and handling edge cases like multiple button presses was hard to understand. Debugging and making sure the stopwatch displayed the time correctly were key tasks

**8. Results and Observations**

* The stopwatch successfully starts, stops, and resets with accurate time tracking down to hundredths of a second.
* The displayed time updates in real-time and remains consistent without noticeable lag.
* The GUI is clean and user-friendly, with clear buttons and a large, readable timer display.
* The use of QMessageBox effectively shows the final elapsed time when stopped.
* All buttons work as expected, even with repeated use—no crashes or glitches observed.

**Overall:** The project meets all intended goals and provides a smooth user experience.

Project Submission

Repository Link; https://github.com/darkmando990/Python\_Project\_61.git

**9. Signatures**

**Student Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Guide Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Submission

Repository Link

**9. Signatures**

**Student Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Guide Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_