VIT®
BHOPAL

NAME – SUNDARAM KUMAR

REG. NO-25BCE11211

FACULTY NAME – PAVITHRA
RATHINAVEL

SLOT-A11+A12+A13

COURSE CODE – CSE1021

# PROJECT REPORT

Stopwatch and Alarm Clock Console Application

## Introduction

This project is a Python-based console application that provides two essential time-related utilities: Stopwatch and Alarm Clock. The program uses simple menu-driven interaction to help users measure time or set alarms.

### Objectives

- Build a functional console application using Python.

- Implement an accurate stopwatch with millisecond precision.

- Develop a custom alarm clock using system time.

- Practice input validation and modular coding.

### Project Description

The application provides:

1. Stopwatch

2. Alarm

3. Quit

It uses Python modules like time, datetime, and sys to manage time and user interaction.

### Features

- Stopwatch with minute, second, and millisecond display.

- Alarm clock with validated user inputs.

- User-friendly interactive menu.

- Graceful exit on interrupt.

### Technology Stack

- Language: Python

- Modules: time, datetime, sys

- Interface: Console based

# Algorithm

## Stopwatch:

1.Wait for user input to start timing.

2.Wait for user input to stop timing.

3.Calculate and display elapsed time.

## Alarm:

1. Ask user for hour, minute, second.

2. Validate input.

3. Continuously check system time.

4. Trigger alert at set time.

# Challenges Faced

- Accurate time measurement.

- Handling interruptions safely.

- Preventing high CPU usage.

- Input validation.

# Learning & Key Takeaways

- Working with time modules.

- Building modular functions.

- Managing infinite loops.

- Graceful exception handling.

# Future Enhancements

- Add sound alerts.

- Add countdown timer.

- Add GUI using Tkinter.

- Add lap functionality.

- Save logs of timings.

## Instructions to Run

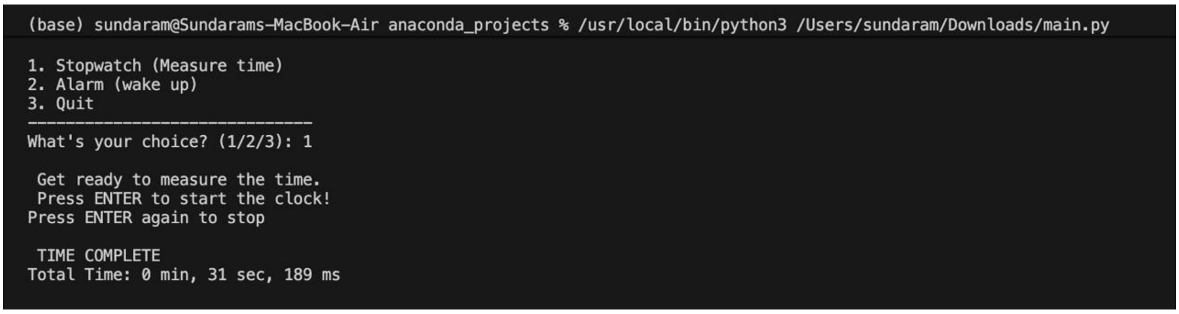
1. Ensure Python 3 is installed.
2. Run: python main.py
3. Follow the on-screen menu.

## Testing Instructions

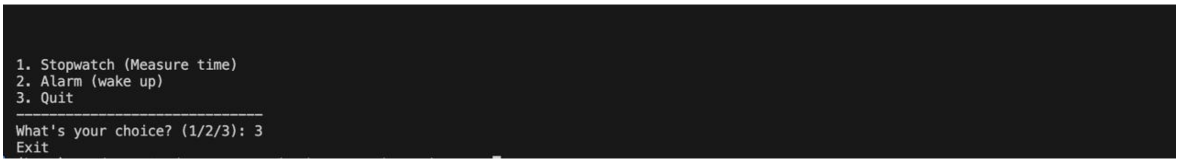
- Test stopwatch accuracy.
- Validate alarm triggering.
- Check input validation.
- Test Ctrl+C functionality.

## Results

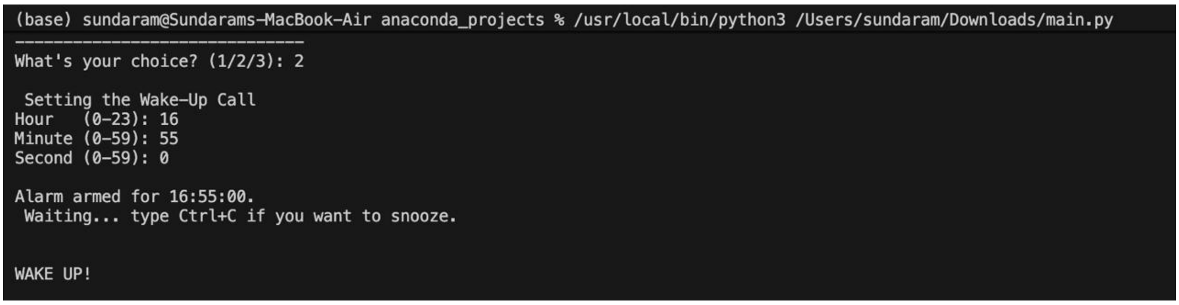
The result of this project is exact and correct. The output of this came correct as expected. Below is the Screenshot of output of project:



```
(base) sundaram@Sundarams-MacBook-Air anaconda_projects % /usr/local/bin/python3 /Users/sundaram/Downloads/main.py

1. Stopwatch (Measure time)
2. Alarm (wake up)
3. Quit
------------------------------
What's your choice? (1/2/3): 1

 Get ready to measure the time.
 Press ENTER to start the clock!
Press ENTER again to stop

 TIME COMPLETE
Total Time: 0 min, 31 sec, 189 ms
```



```
1. Stopwatch (Measure time)
2. Alarm (wake up)
3. Quit
------------------------------
What's your choice? (1/2/3): 3
Exit
```



```
(base) sundaram@Sundarams-MacBook-Air anaconda_projects % /usr/local/bin/python3 /Users/sundaram/Downloads/main.py
------------------------------
What's your choice? (1/2/3): 2

 Setting the Wake-Up Call
Hour   (0-23): 16
Minute (0-59): 55
Second (0-59): 0

Alarm armed for 16:55:00.
 Waiting... type Ctrl+C if you want to snooze.


WAKE UP!
```

## Scope of the Project

Useful for beginners learning Python time manipulation and console utilities.

## Target Users

- Students

- Beginners in Python

- Developers needing time utilities

## References

- Python Documentation: https://docs.python.org/

## Conclusion

This project successfully demonstrates the use of Python's time-handling capabilities through the implementation of a functional stopwatch and alarm clock. By combining input validation, looping structures, and system time monitoring, the application provides a simple yet effective solution for basic time-tracking tasks. The project enhances understanding of modular programming and user interaction in console-based applications. With potential for future upgrades such as sound alerts, a graphical interface, and additional timer features, this application serves as a strong foundation for more advanced time-management tools.