📘 Alarm Clock Project Report

Author: Rashi Mishra

---

1. Introduction

The Alarm Clock project is a simple yet effective Python application created using the Tkinter library. It allows a user to set a specific time, and the program triggers an alarm when the system time matches the user-set time. This project demonstrates concepts like GUI development, time handling, threading, and system-based sound alerts.

---

2. Objectives

- To create a functional alarm clock using Python.
- To implement a user-friendly graphical interface.
- To demonstrate the use of system time and event-driven programming.
- To use multi-threading for smooth execution without freezing the UI.

---

3. Technologies Used

- **Python 3**
- **Tkinter** – for GUI
- **Datetime module** – to fetch system time
- **Threading module** – to run alarm check in background
- **Winsound** (Windows-only) – to play alarm sound

---

4. System Requirements

- Works on Windows, Linux, and macOS
- Python 3.x installed
- No external libraries needed

5. Project Description

The Alarm Clock application displays a live clock and allows the user to input a time in HH:MM or HH:MM:SS format. The app continuously checks for a match between system time and the alarm time. Once the time matches, it triggers a sound alert.

Key Functionalities:

- Display current time
- Accept time input from the user
- Validate time format
- Trigger alarm with sound
- Allow stopping the alarm

6. Working Process

1. The program launches a Tkinter window showing:
    - A live digital clock
    - An input field for alarm time
    - Set Alarm & Stop Alarm buttons
2. When the user sets the alarm:
    - The input time is validated.
    - A new background thread starts checking time every second.
3. When system time == alarm time:
    - Alarm alert is displayed
    - A beep sound plays repeatedly
4. User may stop the alarm using the Stop button.

7. Flowchart

Start → Display Live Time → User Enters Alarm Time → Validate Time →
Start Background Thread → Check Time Every Second →
If Time == Alarm Time → Play Sound → Stop Alarm → End

8. Applications

- Basic daily-use alarm
- Can be extended into task reminders
- Used as a learning project for GUI programming

---

9. Advantages

- Lightweight and fast
- Cross-platform support
- Beginner-friendly code structure

---

10. Future Enhancements

- Support for custom sound files (MP3/WAV)
- Multiple alarms
- Snooze feature
- Dark/Light mode interface
- Alarm history logs

---

11. Conclusion

The Alarm Clock project is an excellent example of combining GUI programming with system-level operations in Python. It provides hands-on experience in event-driven programming, threading, and time-based automation. This project is ideal for beginners looking to practice Tkinter and core Python concepts.
