Alarm Clock Report

Mini Project Report

Submitted in partial fulfillment of the requirements

For the degree of

Bachelor of Engineering (Computer Engineering)

by:

Vaishnavi Ganesh Bavkar [B06]	TU3F2223061
Avadhoot Vilas Chavan [B13]	TU3F2223068

Under the Guidance of

Ms. Bhakti Kaushal

Department of Computer Engineering

TERNA ENGINEERING COLLEGE

Nerul (W), Navi Mumbai 400706

(University of Mumbai)

(2023-2024)

Internal Approval Sheet

TERNA ENGINEERING COLLEGE, NERUL

Department of Computer Engineering

Academic Year 2023-2024

CERTIFICATE

This is to certify that project entitled “Flappy Bird Game” is a bonafide work of:

by:

Vaishnavi Ganesh Bavkar [B06]	TU3F2223061
Avadhoot Vilas Chavan [B13]	TU3F2223068

Submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the Bachelor of Engineering (Computer Engineering).

Guide **Head of Department** **Principal**

Table of Contents

Chapter 1
Introduction

- Alarm clocks play a fundamental role in ensuring individuals wake up on time for their daily activities. With the advancement of technology, alarm clocks have evolved from simple mechanical devices to sophisticated digital applications with various features and functionalities.

- The development of an alarm clock application with a graphical user interface (GUI) using Python presents an exciting opportunity to explore the capabilities of modern software development tools.

- The alarm clock is designed to provide users with an intuitive and user-friendly interface for setting alarms, customizing alarm tones, and managing their wake-up routines efficiently.

- By leveraging Python's Tkinter library for GUI development, we aim to demonstrate how Python can be used to create interactive and visually appealing applications.

- Through this project, we seek to provide insights into the process of designing, implementing, and testing GUI applications using Python, with a specific focus on the development of an alarm clock application as a practical example.

- Overall, the development of an alarm clock application with GUI using Python serves as an educational and practical exercise, highlighting the potential of Python as a versatile tool for GUI development and providing valuable insights into the process of software development in a real-world context.

Chapter 2

Software And Hardware Requirements

Hardware Requirements:

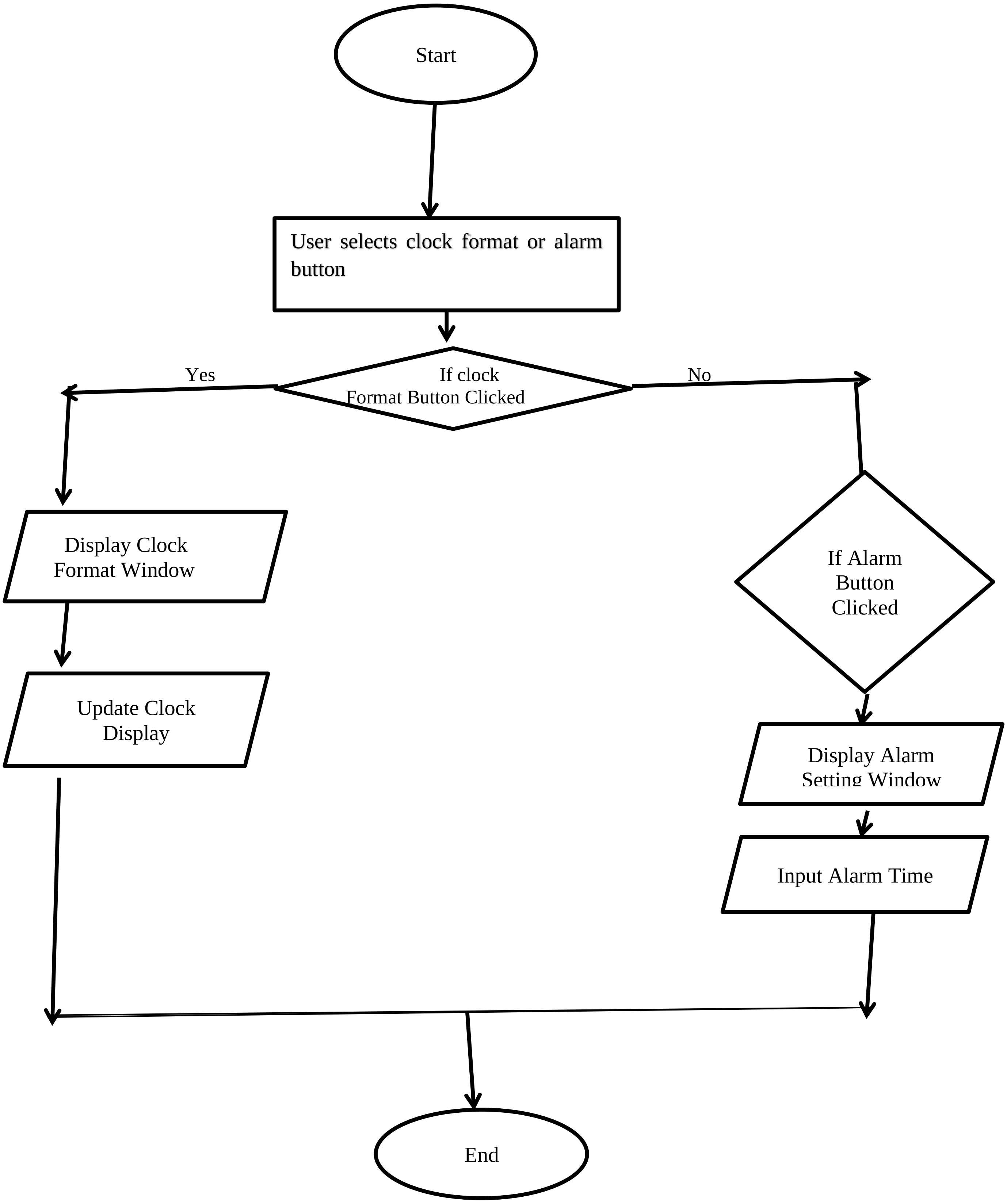
Processor Brand	: AMD
Processor Type	: Ryzen 7 PRO
Processor Speed	: 1.90 GHz
Processor Count	: 1
RAM Size	: 16 GB
Memory Technology	: DDR4
Computer Memory Type	: DDR4
Hard Drive Size	: 160 GB

Software Requirements:

Operating system :	Windows 11 Pro
Application server:	Visual Studio Code

Chapter 3
Flowchart

- Process of the program in detail:


Start
User selects clock format or alarm button
If clock Format Button Clicked
Yes
No
Display Clock Format Window
Update Clock Display
If Alarm Button Clicked
Display Alarm Setting Window
Input Alarm Time
End

Chapter 4
Functions Used

Function	Description
tick(val)	This function is called when the user clicks on either the 12-hour or 24-hour clock button or the alarm button. It opens a new window (Toplevel) and calls one of the following functions based on the value of val: time24(), time12(), or alarm().
time24()	This function updates the label displaying the current time in 24-hour format (HH:MM:SS). It uses time.strftime() to get the current time in the desired format and updates the label text accordingly. It then schedules itself to run again after 1 second using after() method.
time12()	Similar to time24(), this function updates the label displaying the current time in 12-hour format (HH:MM:SS AM/PM). It also uses time.strftime() to get the current time in the desired format and updates the label text accordingly.
alarm()	This function is called when the user clicks on the alarm button. It opens a new window (Toplevel) with input fields for specifying the hour and minute of the alarm. It also creates a "Start" button that is bound to the alarm_begin function using the <Button-1> event.
alarm_begin(event)	This function is called when the user clicks the "Start" button after entering the desired alarm time. It retrieves the hour and minute input from the entry widgets (e1 and e2), converts them to integers, and enters into a loop checking if the current time matches the specified alarm time. Once the alarm time is reached, it plays the alarm sound using pygame.mixer.Sound and displays a message box using messagebox.showinfo().

Chapter 5

Program

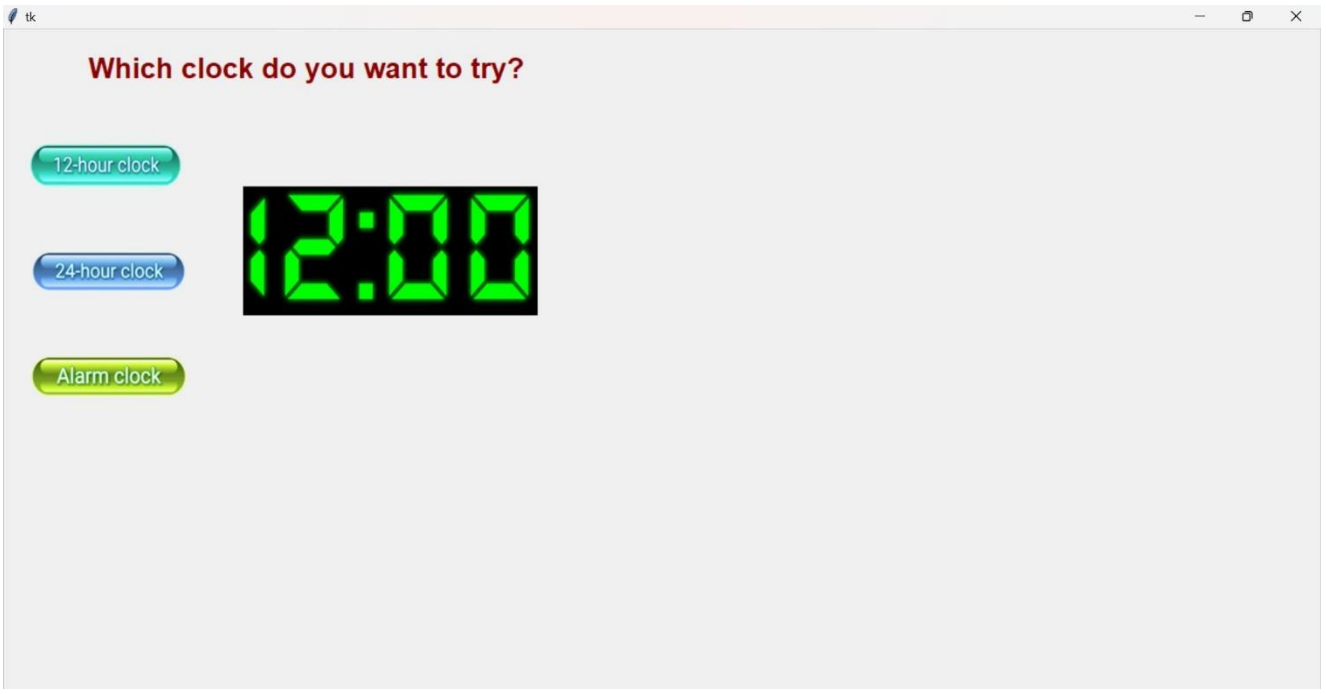
```
import sys
from tkinter import *
from tkinter import messagebox
import time
import datetime
import pygame
pygame.init()
pygame.mixer.init()
from PIL import ImageTk, Image


def tick(val):
    global clock
    window=Toplevel()
    window.geometry("300x130")
    clock=Label(window, font = ('Helvetica', 30, "bold"), fg="light green", bg="black", bd=16,
relief=SUNKEN)
    clock.place(x=18, y=30)
    if val=="btn_24":
        time24()
    elif val=="btn_12":
        time12()
    else:
        alarm()

def time24():
    global clock
    ts=time.strftime("%H:%M:%S")
    clock["text"]=ts
    clock.after(1000, time24)

def time12():
    global clock
    ts=time.strftime("%I:%M:%S:%p")
    clock["text"]=ts
    clock.after(1000, time12)

def alarm():
    global e1, e2
    window=Toplevel()
    window.geometry("350x200")
    hours=Label(window, text="At what hour do you want is to ring?\nPlease enter in military
time.", font = ('Helvetica', 12, "bold"))
```

```
    hours.place(x=30, y=20)
    e1 = Entry(window, relief=GROOVE)
    e1.place(x=30, y=70)
    minutes=Label(window, text="At what minute do you want it to ring?", font = ('Helvetica', 12,
"bold"))
    minutes.place(x=30, y=100)
    e2 = Entry(window, relief=GROOVE)
    e2.place(x=30, y=130)
    begin=Button(window, text="Start", relief=GROOVE)
    begin.place(x=200, y=150)
    begin.bind("<Button-1>", alarm_begin)

def alarm_begin(event):
    global e1, e2
    h=e1.get()
    m=e2.get()
    while(1):
        if (int(h)==datetime.datetime.now().hour and int(m)==datetime.datetime.now().minute):
          sound = pygame.mixer.Sound("Alarm_sound.mp3")
          sound.play()
          messagebox.showinfo("Alarm","Time's up!")
          break




root=Tk()

root.geometry("550x400")
img1=PhotoImage(file="12.png")
img2=PhotoImage(file="24.png")
img3=PhotoImage(file="alarm.png")
text=Label(root, text="Which clock do you want to try?", font = ('Helvetica', 20, "bold"), fg="dark
red")
text.place(x=80, y=20)

btn1=Button(root,        image=img1,        borderwidth=0,        command=lambda:
tick("btn_12")).place(anchor=SW,x=0, y=200)
btn2=Button(root,        image=img2,        borderwidth=0,        command=lambda:
tick("btn_24")).place(anchor=SW,x=0, y=300)
btn3=Button(root, image=img3, borderwidth=0, command=alarm).place(anchor=SW,x=0,y=400)


photo=ImageTk.PhotoImage(Image.open("clock.png"))

img_label=Label(root, image=photo)
img_label.place(x=230, y=150)

root.mainloop()
```

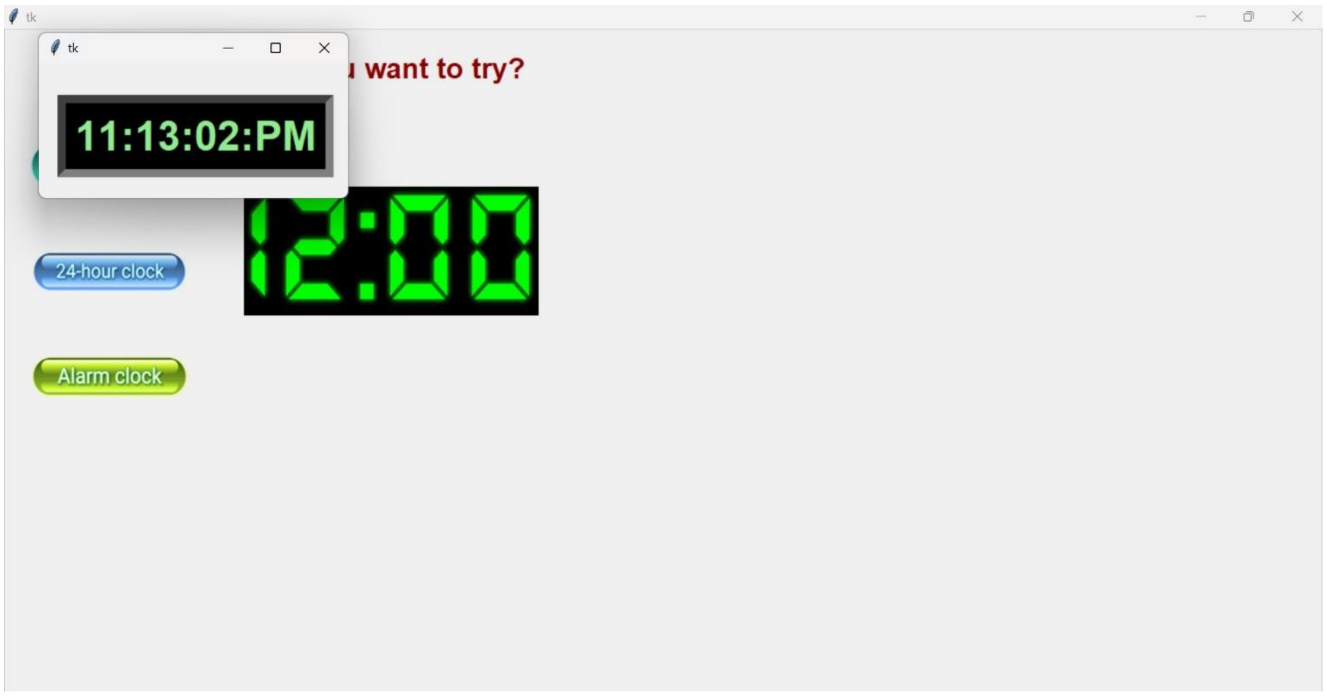
Chapter 6

Output Snapshots

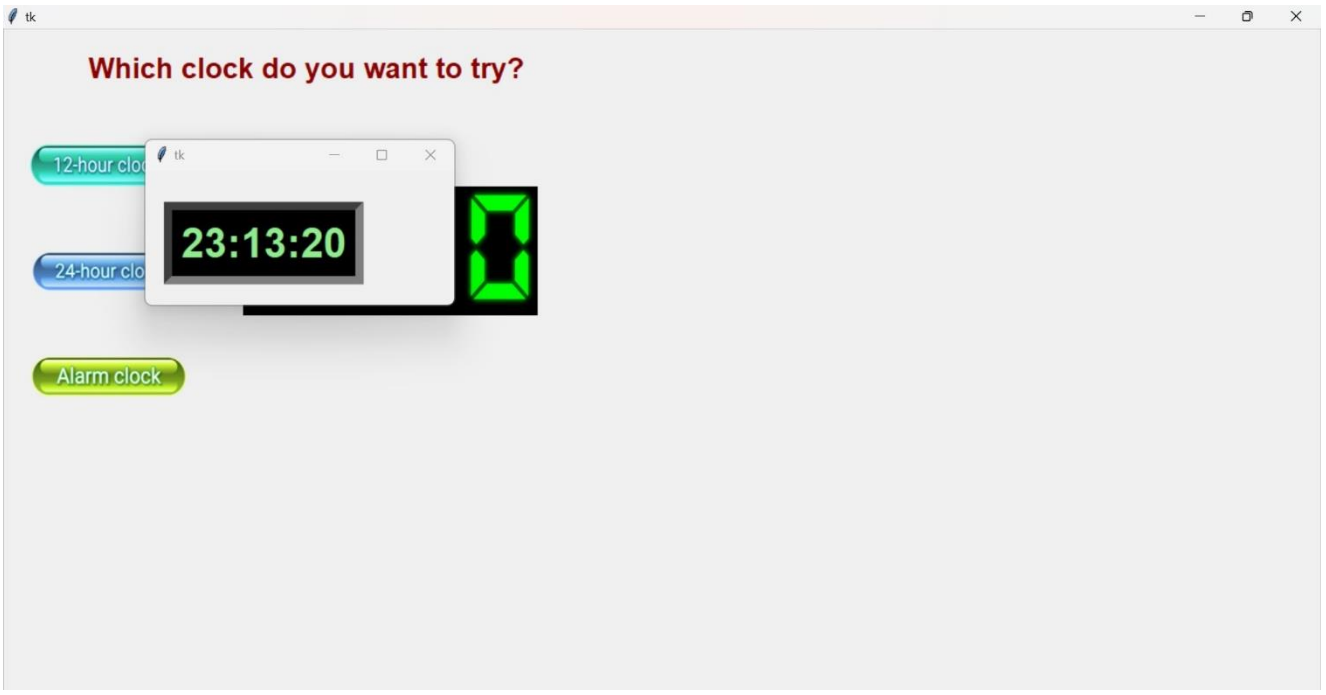
1.Interface:



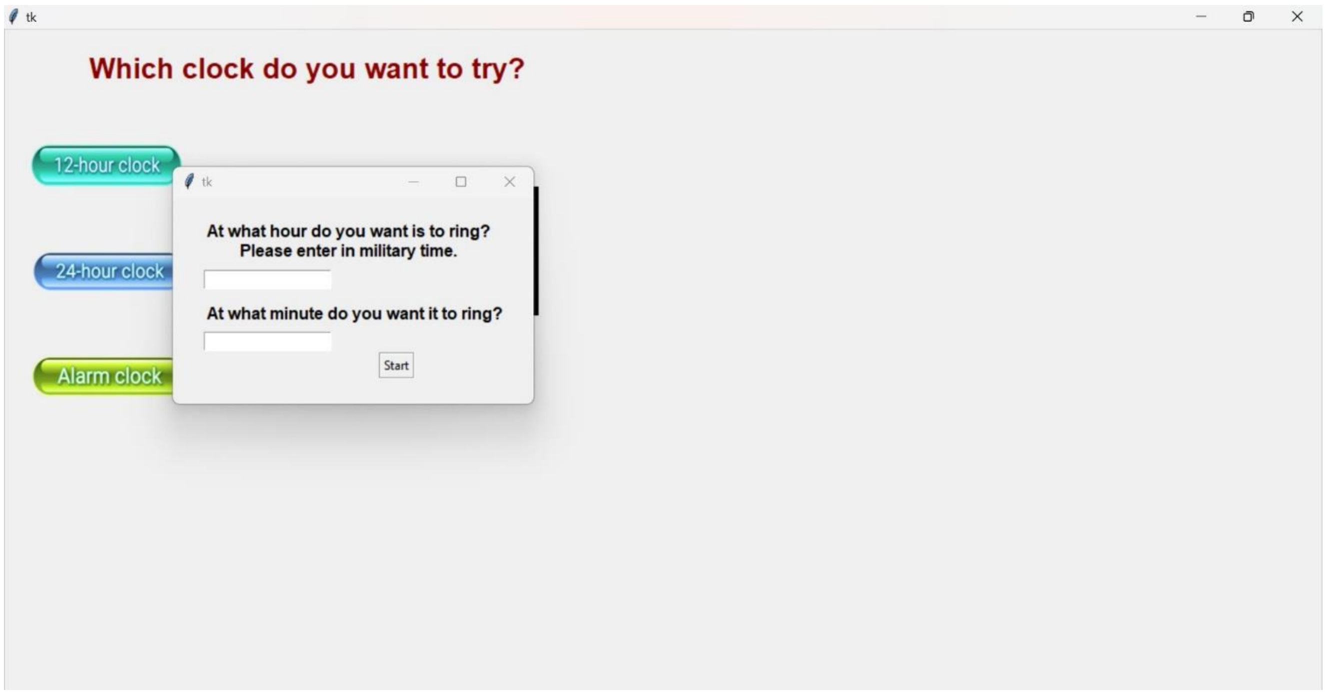
2.When 12 hour format appears:



3.When 24 hour format appears:



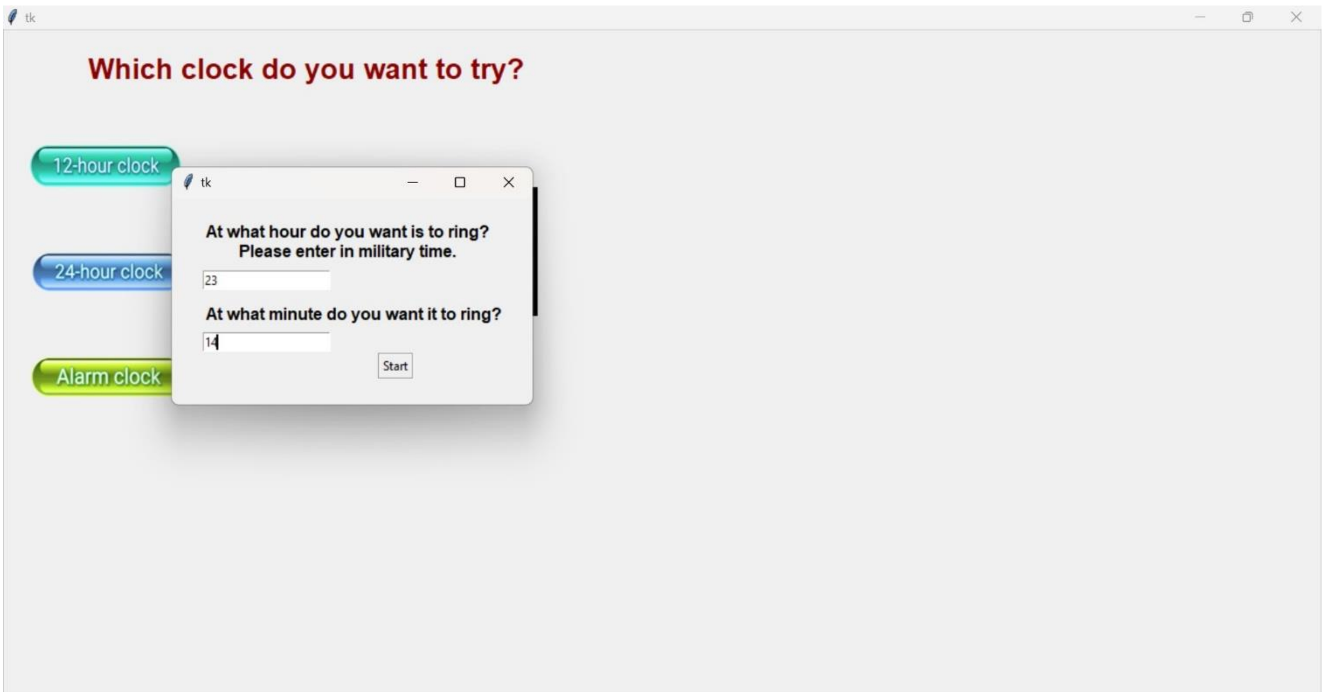
4.Alarm should be set using military time format (24 hour format):



At what hour do you want is to ring?
Please enter in military time.

At what minute do you want it to ring?

5.When the alarm is set:


Which clock do you want to try?

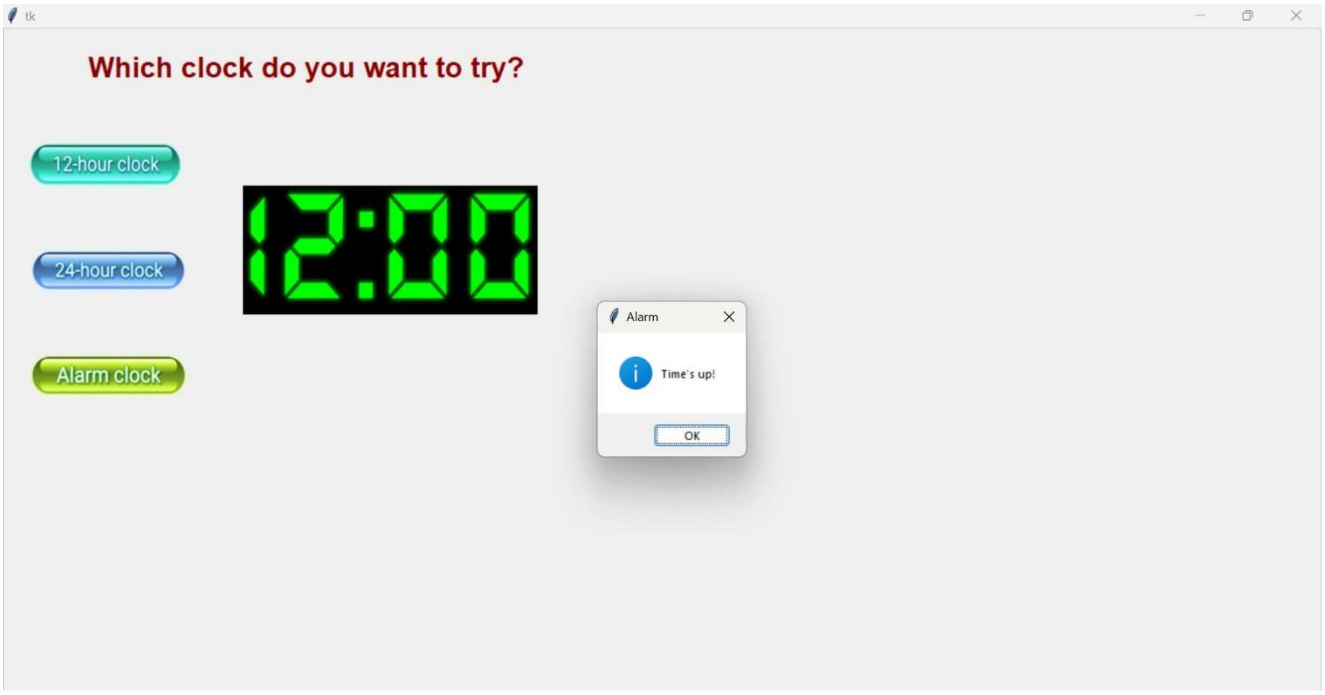
At what hour do you want is to ring?
Please enter in military time.

```
23
```

At what minute do you want it to ring?

```
14
```

6.When the alarm rings:


Which clock do you want to try?

```
12:00
```

Time's up!

Chapter 7

Conclusion

- In conclusion, the development of an alarm clock application using Python with a graphical user interface (GUI) showcases the versatility and power of Python in creating interactive and functional applications.

- The project has provided valuable insights into best practices for GUI design, user experience considerations, and error handling techniques.

- Looking ahead, there are several opportunities for further improvement and expansion of the alarm clock.

- Overall, the development of the alarm clock using Python with GUI serves as a testament to the versatility and accessibility of Python for GUI application development. By leveraging Python's rich ecosystem of libraries and tools, developers can create innovative and user-friendly applications that cater to diverse needs and preferences.

Chapter 8
Reference

- https://www.geeksforgeeks.org/creat-an-alarm-clock-using-tkinter/
- https://youtu.be/nYVgO6HR8bw?feature=shared