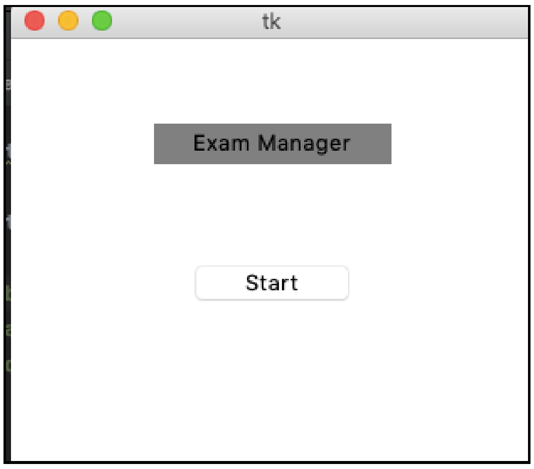
Criterion C: Product Development

Product structure:

For the realisation and graphic interface of the program I will be using a Tkinter tool. I use widgets like labels (for displaying text, e. g time, date, table name), buttons (like “end all” and ”enter new subject”), entries (to create a table with exams). I also use the datetime library for displaying current date and time in the program. For the alarms I use the pygame library so it has not only visual but also a sound stimuli that indicates the 5 minute mark. For adding the inserted exams into the database and displaying them I use the sqlite3 library.

Starting window:

Button “Start” - takes the user to the main window.



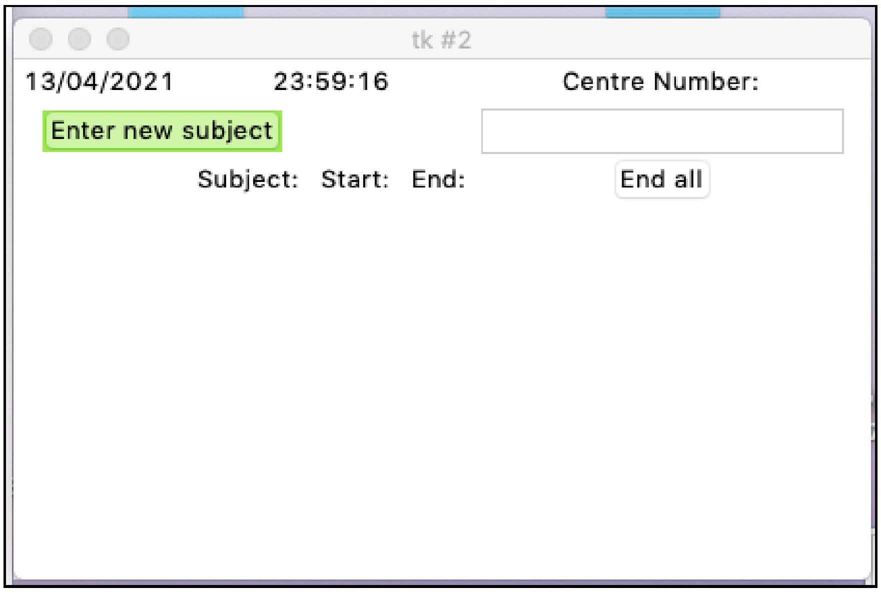
```
from tkinter import *

start = Tk()

l1 = Label(start, text="Exam Manager", width=15, bg='grey', fg='black')

b1 = Button(start, text="Start", width=10, bg='green', fg='black', common = startexam)<-- function “startexam” takes us to the main window

l1.pack(side=TOP, pady=50)

b1.pack(side=TOP, pady=10)

start.mainloop()
```

We can already see that to create the starting page I use labels - the title, and a button - “start” to open the main window. The button has function “startexam” which takes as to the main window.

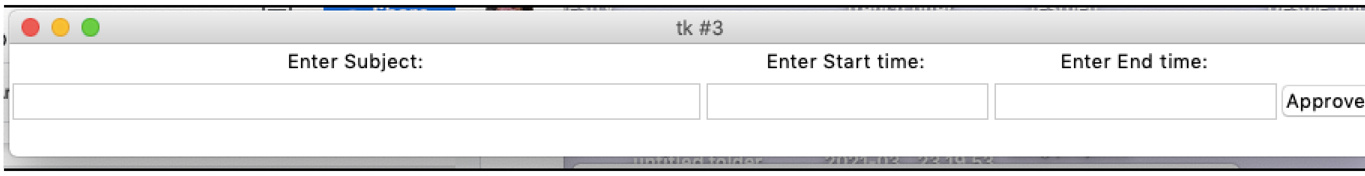
Main window:

This window appears after pressing the “start” button in the previous window.

“Enter new subject” - a button that allows to create a new line in the table



Button “Enter new subject” opens the entry window where the user enter the necessary data about new exam:



It contains tkinter’s labels, buttons, entries that are visible through a grid method that allows us to visualise the functionality like table, rows and columns.

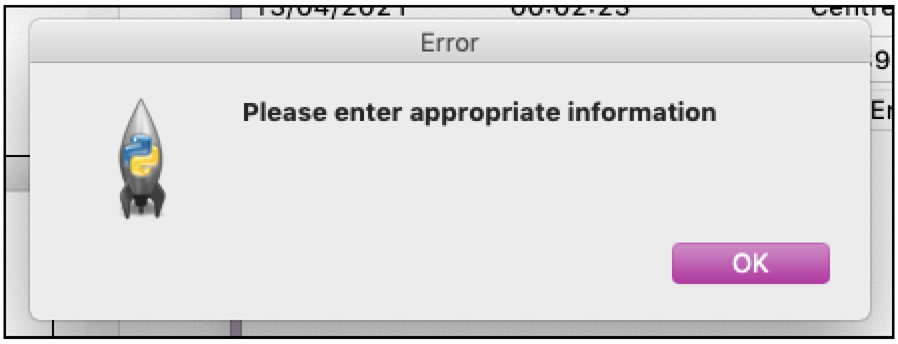
```
Label(root2, text="Subject:").grid(row=3, column=3, columnspan=1)

Label(root2, text="Start:").grid(row=3, column=4, columnspan=1)

Label(root2, text="End:").grid(row=3, column=5, columnspan=1)

a = Entry(root2, width=50)

a.grid(row=5, column=1, columnspan=1)

b = Entry(root2, width=20)

b.grid(row=5, column=4, columnspan=1)

c = Entry(root2, width=20)

c.grid(row=5, column=5, columnspan=1)

Button(root2, text="approve", command=check).grid(row=8, column=8, columnspan=8)

root2.mainloop()
```

"command=check" - function that checks the data inserted like:

The only appropriate information in the window should be:
-only string characters in the entries
-start time must be smaller than end time
-the time has to be in appropriate format (no 24:00 or 12:61)
If the information is correct the approve button is pressed which inserts the data(subject, time) to database and from there it demonstrates it to the table

```
def check(a,b,c):


  subject = a.get()
  st = b.get()
  ed = c.get()
  time_re = re.compile(r'^([0-1]?[0-9]|2[0-3]):[0-5][0-9]$')


  def is_time_format(s):
     return bool(time_re.match(s))
  if subject != '' and bool(time_re.match(st)) == True and bool(time_re.match(ed)) == True:
     answer = mb.askyesno(
        title = "New Exam",
        message="Insert new exam?")
     if answer:
        with sqlite3.connect('Sasha.db') as connection:
           cursor = connection.cursor()
           cursor.execute("Insert into ExamManager(Subject, start, end, flag) values(?, ?, ?, ?)",(a.get(), b.get(), c.get(), 0))
           connection.commit()
           recreate_table()
           root2.destroy()
  else:
     mb.showerror(
        "Error",
        "Please enter appropriate information")
```

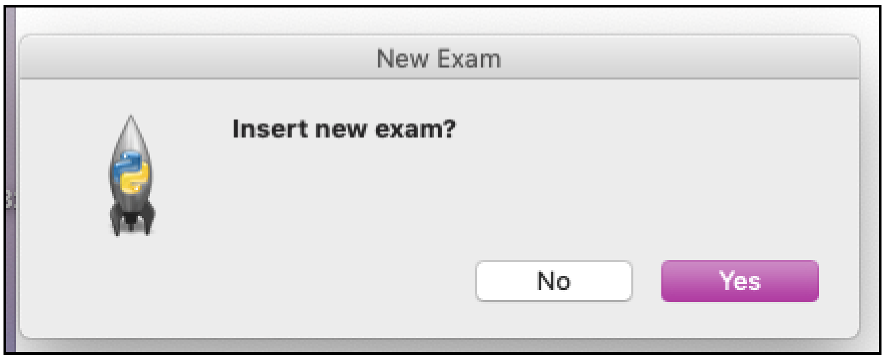
ERROR CHECK


Error

Please enter appropriate information

This is another question window that appears upon pressing "approve button":

We create a seperate function "check" that uses the tkinter method and asks for approval of insertion. After pressing "yes" button it sends the information to the database sqlite and displays it in the table that is called ExamManager and contains three fields - Subject, Start, End.


New Exam

Insert new exam?

```
def check():


    answer = mb.askyesno(
        title="New Exam",
        message="Insert new exam?")


    if answer:


        with sqlite3.connect('Sasha.db') as connection:
            cursor = connection.cursor()


            cursor.execute("Insert into ExamManager(Subject, start, end, flag) values(?, ?, ?, ?)",(a.get(), b.get(), c.get(), 0))
            connection.commit()


            recreate_table()


            root2.destroy()
```

Label “finished” appears in two options:

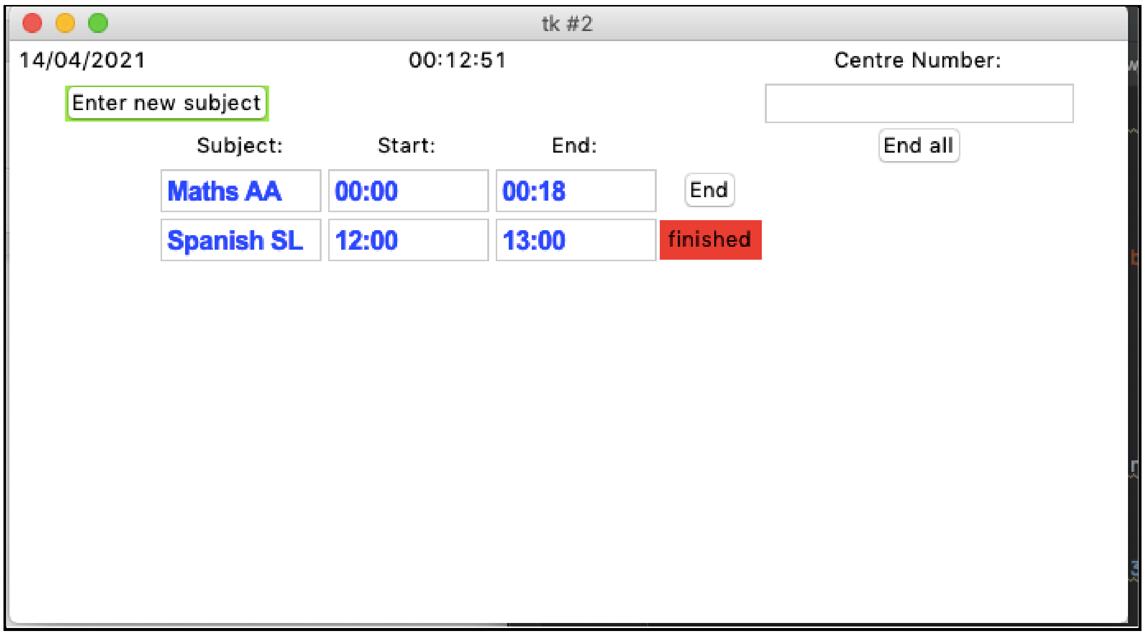
-when button “end” is pressed and the exam is finished manually;

```
def end_all ():
   for i in range(len(data)):
        Label(text="finished", bg='red').grid(row=4+i, column=6, columnspan=1,)
   #root.destroy()
   finishexam(root)
```

-when the real time matches the end time of the exam meaning the exam is finished automatically:

```
if now in times:
   index = [i for i in range(len(times)) if now == times[i]]
   for i in index:
      Label(text="finished", bg="#4db6eb").grid(row=4+i, column=6, columnspan=1, )
```

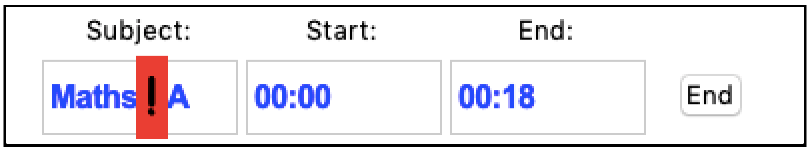
```
label.config(text=now)
label.after(1000, clock)
```



14/04/2021   00:12:51   Centre Number:

Subject:	Start:	End:	
Maths AA	00:00	00:18	End
Spanish SL	12:00	13:00	finished

The warning icon and alarm sound appear 5 minutes before the end of each exam:



Subject:	Start:	End:	
Maths ! A	00:00	00:18	End

```
if alarm in times:
   pg.mixer.music.play()
   index = [i for i in range(len(times)) if alarm == times[i]]
   for i in index:
       a = Label(text="!", bg='red', font=("Comic Sans MS", 24, "bold"))
       a.grid(row=4 + i, column=1, columnspan=4)
# the statement above accounts for displaying the exclamation point warning
if now in times:
   index = [i for i in range(len(times)) if now == times[i]]
   for i in index:
       Label(text="finished", bg="red").grid(row=4+i, column=6, columnspan=1, )
```

```
Label(text=date.today().strftime("%d/%m/%Y")).grid(row=1, column=2, columnspan=1)
    label = Label(text="")
    label.grid(row=1, column=3, columnspan=4)
    Label(text="Centre Number:").grid(row=1, column=7, columnspan=2)
    Entry().grid(row=2, column=7, columnspan=2)
now = time.strftime("%H:%M:%S")
label.config(text=now)
        label.after(1000, clock)
```

This is how the time is displayed in the mail window:

```
Label(text=date.today().strftime("%d/%m/%Y")).grid(row=1, column=2, columnspan=1)
```

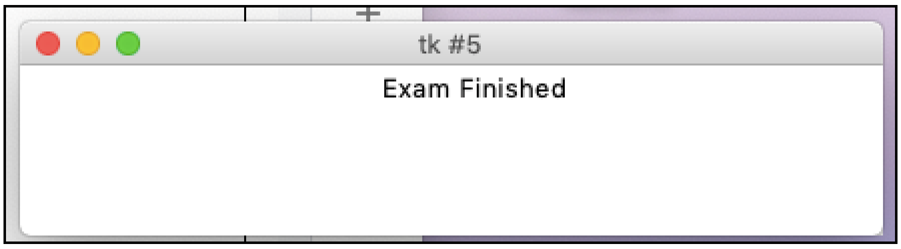
We create the function “recreate_table " that after we add a new subject to the table, it connects to the database and then recreates the updated table.

```
def recreate_table():

   with sqlite3.connect('Sasha.db') as connection:

      cursor = connection.cursor()

      cursor.execute("select * from ExamManager")

      data = [row for row in cursor.fetchall()]

   butt = []

   for i in range(len(data)):

      b = Button(text="End")

      butt.append(b)

   for i in range(len(data)):

      for j in range(len(data[0])):

         if j != len(data[0]) - 1:

            e = Entry(root, width=10, fg='blue', font=('Arial', 16, 'bold'))

            e.grid(row=4 + i, column=3 + j, sticky='nsew', columnspan=1)

            e.insert(END, data[i][j])

         else:

            butt[i].grid(row=4 + i, column=3 + len(data[0]) - 1, columnspan=1)

   for i in range(len(butt)):

      butt[i].config(command=lambda i=i: end_32(4 + i))
```

This is what is displayed when the “end all” button is pressed. It finishes all exams that are inserted in the table at the moment:

end all:

```
def finishexam(root):
   #global root
   root.destroy()
   end = Tk()
   f_top = Frame(end)
   f_bot = Frame(end)
   a = Label(text="Exam Finished",width=50)
   a.grid(row=1, column=1, columnspan=1)
   end.mainloop()
```



Exam Finished

Word count: 466