Functions used in Program

insert_command() : Opens insert window

i_command(): Inserts value given in text boxes

query_command() : opens query window

f_command(): Search for record by name

f1_command(): Search by quantity (give < or > or = or <= or >=) when searching

f2_command(): Search by price (give < or > or = or <= or >=) when searching

um_command(): Updates medicine name

ua_command(): Updates quantity

up_command(): Updates price

c_command(): Clears all inserted labels and values in text box

d_command(): Deletes table

needed_command() : Opens needed window

nm_command(): Add medicines which are needed(manually)

na_command(): Automatic search medicines which less than 5 in table

nc_command(): clears all labels,output and entries

t_command() : Opens new table window

nt_command(): Creates new table when table name given

a_command(): calculates amount of medicines when purchased

p_command(): insert medicines name , quantity , price in csv file and displays its records with minimum price for selling

b_command(): Prints bill and substract quantity when medicine is sell , in table

Program

```
from tkinter import *        #importing tkinter library
import mysql.connector as sql  #importing sql library
import csv                     #importing csv

root = Tk()    #creating window for main window

#<name>.geometry used to defne size of window
root.geometry("650x400")       #size of main window

# ['bg'] defines background color of window
root['bg'] = '#f8f7fc'         #background color of window

con = sql.connect(host = "localhost" , user = "root" , passwd = "yato" , database =
"med")   #connecting to database
cur = con.cursor()            #creating cursor for query execution

# .title used to set title of window
root.title("Medical Store Management")   #heading of main window

#Keywords used in program

#Entry used to create text boxes window
#Labels used to create labels in window
#<name>.gird used to set position of objects in window
#Button used to create button in window
#<name>.delete used to delete value inserted in text boxes
#text = set text of labels or buttons
#widht = set width of text boxes and buttons
#pady = set padding on y axis
#ipadx = set padding of buttons
#command = command given to buttons
#<name>.insert = inserts value in text box
```

```
#---------------creating new windows--------------

#-----creating insert window

def insert_command():
    i_window = Toplevel()     #new window for inserting data
    i_window.geometry("300x300")   #size of insert window

    i_window.title("Insert")       #insert window title
    i_window['bg'] = "#f8f7fc"      #window background color

    #creating text boxes

    #i = insert , m = medicines , #q = quantity , #p = price , #t = table
    im_name = Entry(i_window , width=30)
    im_name.grid(row = 1, column = 1 , pady = 10) #defining position of text box

    iq_name = Entry(i_window , width=30)
    iq_name.grid(row = 2 , column = 1 , pady = 10)

    ip_name = Entry(i_window , width=30)
    ip_name.grid(row = 3, column = 1 , pady = 10)

    it_name = Entry(i_window , width=30)
    it_name.grid(row = 4, column = 1 , pady=10)

    #creating labels
    im_label = Label(i_window , text="Medicine Name" , bg="#f8f7fc").grid(row= 1, column =0)
    #labelsposition

    iq_label = Label(i_window , text="Quantity" , bg="#f8f7fc").grid(row = 2, column = 0)
    ip_label = Label(i_window , text="Price" , bg="#f8f7fc").grid(row=3 , column = 0)
    it_label = Label(i_window , text="Table Name" , bg="#f8f7fc").grid(row = 4, column = 0)


    #functions for insert window

    def i_command():       #for inserting values
        i_add = """INSERT INTO {} Values
        ('{}' , {} , {})""".format(it_name.get(),im_name.get(),iq_name.get(),ip_name.get())
        #sql syntax for inserting

        cur.execute(i_add)           #command execution
        con.commit()                 #commiting changes


        i_complete = "Data inserted"      #on successfully inserting value

        i_print = Label(i_window , text=i_complete)   #output
```

```
        i_print.grid(row = 7 , column = 1 )

        #deleting values of text box
        im_name.delete(0 , END)
        iq_name.delete(0 , END)
        ip_name.delete(0 , END)

    #creating buttons
    i_insert = Button(i_window , text="Insert" , command = i_command)
    i_insert.grid(row=5 , column=0 , columnspan=2 , pady=10 , padx=10 , ipadx=100)

#----------creating query window

def query_command():
    q_window = Toplevel()   #new window
    q_window.geometry("730x350")

    q_window.title("Query")  #query window title
    q_window['bg'] = "#f8f7fc"


    #creating text boxes

    #q = query , u = update , p = price , t = table
    qm_name = Entry(q_window , width=30)
    qm_name.grid(row = 1, column = 1 , pady = 10)

    qq_name = Entry(q_window , width=30)
    qq_name.grid(row = 2 , column = 1 , pady = 10)

    qp_name = Entry(q_window , width=30)
    qp_name.grid(row = 3, column = 1 , pady = 10)

    qu_name = Entry(q_window , width=30)
    qu_name.grid(row = 1, column = 3 , pady = 10)

    quq_name = Entry(q_window , width=30)
    quq_name.grid(row = 2 , column = 3 , pady = 10)

    qup_name = Entry(q_window , width=30)
    qup_name.grid(row = 3, column = 3 , pady = 10)

    qt_name = Entry(q_window , width=30)
    qt_name.grid(row = 4, column = 1 , pady=10)

    #creating labels
    qm_label = Label(q_window , text="Medicine Name" , bg="#f8f7fc").grid(row= 1, column = 0)
    qq_label = Label(q_window , text="Quantity" , bg="#f8f7fc").grid(row = 2, column = 0)
    qp_label = Label(q_window , text="Price" , bg="#f8f7fc").grid(row=3 , column = 0)
    qt_label = Label(q_window , text="Table Name" , bg="#f8f7fc").grid(row = 4, column = 0)
```

```
    qu_label = Label(q_window , text="Update Medicine Name" , bg="#f8f7fc").grid(row= 1,
column = 2 , padx=10)
    quq_label = Label(q_window , text="Add Stock" , bg="#f8f7fc").grid(row = 2, column = 2 ,
padx=10)
    qup_label = Label(q_window , text="Change Price" , bg="#f8f7fc").grid(row=3 , column = 2
, padx=10)

    #functions for query window
    def f_command():                        #defining functions
        global qr_output

        f_text = """SELECT * FROM {}
        WHERE Medicine like '{}'""".format(qt_name.get() , qm_name.get())

        cur.execute(f_text)
        o = cur.fetchall()

        q_output = ' '
        for i in o:
            q_output += str(i) + '\n'

        qr_output = Label(q_window , text=q_output)  #give record in form of label
        qr_output.grid(row = 7 , column=1)

        con.commit()

    def f1_command():
        global qr_output

        f_text = """SELECT * FROM {}
        WHERE Quantity {}""".format(qt_name.get() , qq_name.get())

        cur.execute(f_text)
        o = cur.fetchall()

        q_output = ' '
        for i in o:
            q_output += str(i) + '\n'

        qr_output = Label(q_window , text=q_output)
        qr_output.grid(row = 7 , column=1)

        con.commit()

    def f2_command():
        global qr_output

        f_text = """SELECT * FROM {}
        WHERE Price {}""".format(qt_name.get() , qp_name.get())
```

```
    cur.execute(f_text)
    o = cur.fetchall()

    q_output = ' '
    for i in o:
        q_output += str(i) + '\n'

    qr_output = Label(q_window , text=q_output)
    qr_output.grid(row = 7 , column=1)

    con.commit()

def um_command():          #functions of updation
    global qr_output

    u_text = """UPDATE {}
    SET MEDICINE = '{}'
    WHERE MEDICINE LIKE '{}'""".format(qt_name.get() , qu_name.get() , qm_name.get())

    cur.execute(u_text)
    con.commit()

    qr_output = Label(q_window , text="Updated")
    qr_output.grid(row = 7 , column=1)

def ua_command():
    global qr_output

    u_text = """UPDATE {}
    SET Quantity = {}
    WHERE Medicine = '{}'""".format(qt_name.get() , quq_name.get() , qm_name.get())

    cur.execute(u_text)
    con.commit()

    qr_output = Label(q_window , text="Updated")
    qr_output.grid(row = 7 , column=1)

def up_command():
    global qr_output

    u_text = """UPDATE {}
    SET Price = {}
    WHERE MEDICINE LIKE '{}'""".format(qt_name.get() , qup_name.get() , qm_name.get())

    cur.execute(u_text)
    con.commit()

    qr_output = Label(q_window , text="Updated")
    qr_output.grid(row = 7 , column=1)
```

```
#clear text boxes and label
def c_command():
    qm_name.delete(0 , END)
    qq_name.delete(0 , END)
    qp_name.delete(0 , END)
    qu_name.delete(0 , END)
    quq_name.delete(0 , END)
    qup_name.delete(0 , END)
    qr_output.destroy()

def d_command():  #function for table deletion
    global qr_output

    d_text = "DROP TABLE {}".format(qt_name.get())

    cur.execute(d_text)

    qr_output = Label(q_window , text="Successfully Deleted")
    qr_output.grid(row = 7 , column=1)

    con.commit()

#creating buttons for query

#f = find #c = clear #d = delete
f_button = Button(q_window , text="Find from name" , command= f_command)
f_button.grid(row=5 , column=0 , ipadx=40 , pady=30)

f1_button = Button(q_window , text="Find in quantity" , command= f1_command)
f1_button.grid(row=5 , column=1 , ipadx=40 , pady=30)

f2_button = Button(q_window , text="Find in price" , command= f2_command)
f2_button.grid(row=5 , column=2 , ipadx=40 , pady=30)

um_button = Button(q_window , text="Update Name" , command=um_command)
um_button.grid(row=6 , column=0 , ipadx=40)

ua_button = Button(q_window , text="Add Stock" , command=ua_command)
ua_button.grid(row=6 , column=1 , ipadx=40)

up_button = Button(q_window , text="Change Price" , command=up_command)
up_button.grid(row=6 , column=2 , ipadx=40)

c_button = Button(q_window , text="Clear" , command=c_command)
c_button.grid(row=5 , column=3 , ipadx=50 , pady=30)

d_button = Button(q_window , text="Delete Table" , command=d_command)
d_button.grid(row=6 , column=3 , ipadx=50)
```

```
#-------creating needed window

def needed_command():
    n_window = Toplevel()
    n_window.geometry("500x400")
    n_window.title("Needed Medicines")
    n_window['bg']="#f8f7fc"

    #creating boxes

    #n = needed
    nm_name = Entry(n_window , width=30)
    nm_name.grid(row = 1, column = 1 , pady = 10)

    nt_name = Entry(n_window , width=30)
    nt_name.grid(row=2 , column=1)

    #creating labels
    nm_label = Label(n_window , text="Medicine Name" , bg="#f8f7fc")
    nm_label.grid(row = 1, column = 0 , pady = 10)

    nt_label = Label(n_window , text="Table name(for automatic search)" , bg="#f8f7fc")
    nt_label.grid(row=2 , column=0)

    na_label = Label(n_window , text="Left less then 5" , bg="#f8f7fc")
    na_label.grid(row=3 , column=0)

    #functions for needed window
    def nm_command():
        global no
        nf = open("Needed.txt" , "a")
        nf.write(nm_name.get() + '\n')

        no = Label(n_window , text="Successfully Added")
        no.grid(row=4 , column=1)

    def na_command():    #automatic search medicines
        global qr_output
        na_text = """SELECT Medicine FROM {}
        WHERE QUANTITY < 5""".format(nt_name.get())

        cur.execute(na_text)
        o = cur.fetchall()

        q_output = ' '

        for i in o:
            q_output += str(i) + '\n'

        qr_output = Label(n_window , text=q_output)
```

```
        qr_output.grid(row = 4 , column=1)

        con.commit()

    def nc_command():
        global no
        qr_output.destroy()
        no.destroy()

    #creating buttons

    nm_button = Button(n_window , text="ADD" , command=nm_command)
    nm_button.grid(row=1 , column=2 , padx=10 , ipadx=20)

    na_button = Button(n_window , text="Automatic Search" , command=na_command)
    na_button.grid(row=3 , column=1 , pady=10 , ipadx=20)

    nc_button = Button(n_window , text="Clear" , command=nc_command)
    nc_button.grid(row=2 , column=2 , pady=10 , ipadx=20)

#-------creating new table window

def t_command():
    t_window = Toplevel()
    t_window.geometry("400x300")
    t_window.title("New table")
    t_window['bg'] = "#f8f7fc"

    #text box for table window
    t_name = Entry(t_window , width=40)
    t_name.grid(row=0 , column=1)

    #label for table window
    t_label = Label(t_window , text="Table Name" , bg="#f8f7fc").grid(row=0 , column=0)

    #function for table window

    def nt_command():
        nt_text = """CREATE TABLE {}(
            Medicine VARCHAR(20),
            Quantity INT,
            Price INT)""".format(t_name.get())

        cur.execute(nt_text)

        nt_created = Label(t_window , text="New table created")
        nt_created.grid(row=2 , column=0 , pady=20)

        con.commit
```

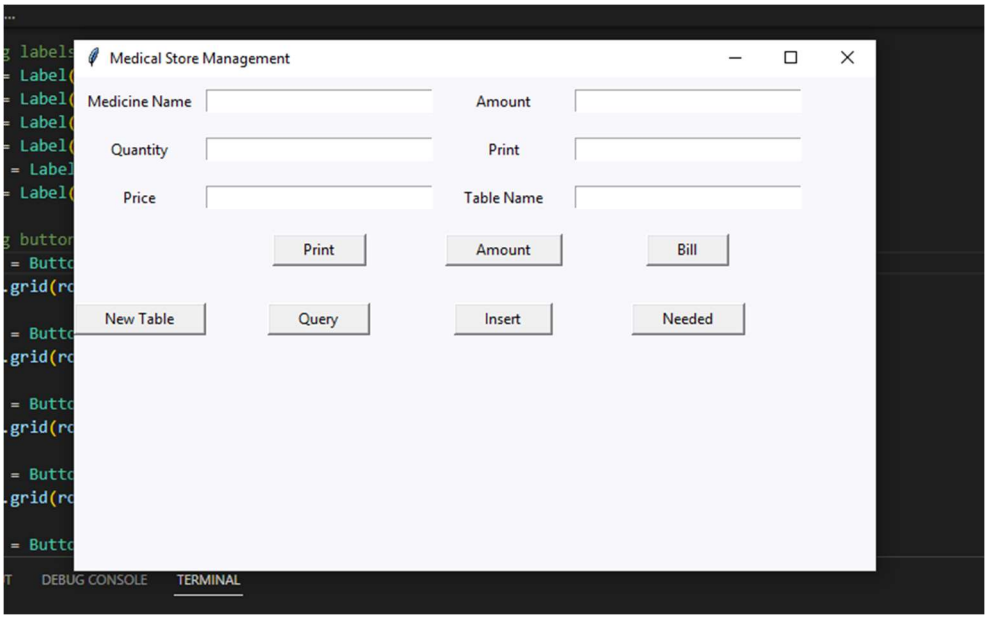
```
    #buttons for new table window

    nt_button = Button(t_window , text="Create" , command=nt_command)
    nt_button.grid(row=1 , column=1 , pady=10)


#creating functions for main window

def a_command():
    import csv
    cr = open("bill.csv" , "r")
    crr = csv.reader(cr)

    amount = 0
    for rec in crr:
        amount = amount + (int(rec[1])*int(rec[2]))

    a_name.delete(0 , END)
    a_name.insert(0 , amount)


def p_command():
    import csv
    pfc = open("bill.csv" , "a" , newline='\n')

    pfr = csv.writer(pfc)

    global qr_output

    f_text = """SELECT Medicine,0.1*Price + Price FROM {}
        WHERE Medicine like '{}'""".format(t_name.get() , m_name.get())

    cur.execute(f_text)
    o = cur.fetchall()

    q_output = ' '
    for i in o:
        q_output += str(i) + '\n'

    qr_output = Label(root , text=q_output)
    qr_output.grid(row = 6 , column=1)

    con.commit()

    a_name.delete(0 , END)

    a_name.insert(0 , int(q_name.get())*int(p_name.get()))


    pr_name.delete(0 , END)
```

```
    csvr = [m_name.get() , q_name.get() , p_name.get()]
    pfr.writerow(csvr)

    m_name.delete(0 , END)
    q_name.delete(0 , END)
    p_name.delete(0 , END)

    pr_name.insert(0 , "Added")

def b_command():
    import csv
    pc = open("bill.csv" , "r")
    cr = csv.reader(pc)


    for rec in cr:
        b_text = """UPDATE {}
        SET QUANTITY = Quantity - {}
        WHERE MEDICINE LIKE '{}'""".format(t_name.get() , rec[1] , rec[0])

        cur.execute(b_text)
        con.commit()

    qr_output.destroy()

    pc.close()

    pw = open("bill.csv" , "w")
    pw.truncate()
    pw.close()

#creating text boxes for main window
m_name = Entry(root , width=30)
m_name.grid(row = 1, column = 1 , pady = 10)

q_name = Entry(root , width=30)
q_name.grid(row = 2 , column = 1 , pady = 10)

p_name = Entry(root , width=30)
p_name.grid(row = 3, column = 1 , pady = 10)

a_name = Entry(root , width=30)
a_name.grid(row = 1, column = 3 , pady=10)

pr_name = Entry(root , width=30)      #pr = print
pr_name.grid(row = 2, column = 3 , pady=10)

t_name = Entry(root , width=30)
t_name.grid(row = 3, column = 3 , pady=10)
```

```
#creating labels
m_label = Label(root , text="Medicine Name" , bg="#f8f7fc").grid(row= 1, column = 0)
q_label = Label(root , text="Quantity" , bg="#f8f7fc").grid(row = 2, column = 0)
p_label = Label(root , text="Price" , bg="#f8f7fc").grid(row=3 , column = 0)
a_label = Label(root , text="Amount" , bg="#f8f7fc").grid(row=1 , column = 2)
pr_label = Label(root , text = "Print" , bg="#f8f7fc").grid(row=2 , column=2)
t_label = Label(root , text="Table Name" , bg="#f8f7fc").grid(row = 3, column = 2)

#creating buttons
a_button = Button(root , text="Amount" , command = a_command)
a_button.grid(row = 4 , column = 2 , pady=10 , padx = 10 , ipadx = 20)

p_button = Button(root , text="Print" , command = p_command)
p_button.grid(row = 4 , column = 1 , pady=10 , padx = 10 , ipadx = 20)

i_button = Button(root , text = "Insert" , command = insert_command)
i_button.grid(row = 5 , column = 2 , ipadx = 20 , pady=20)

b_button = Button(root , text="Bill" , command = b_command)
b_button.grid(row= 4 , column = 3 , ipadx=20)

q_button = Button(root , text="Query" , command= query_command)
q_button.grid(row = 5 , column = 1 , ipadx=20 , pady=20)

n_button = Button(root , text="Needed" , command= needed_command)
n_button.grid(row = 5 , column = 3 , ipadx=20 , pady=20)

t_button = Button(root , text="New Table" , command= t_command)
t_button.grid(row = 5 , column = 0 , ipadx=20 , pady=20)

root.mainloop()  #infinite loop for window
```

Output

Main Window-



Medicine Name – <Enter Medicine Name>

Amount - <Automatically gives amount>

Quantity - <Enter Quantity>

Print – write selling info in csv file

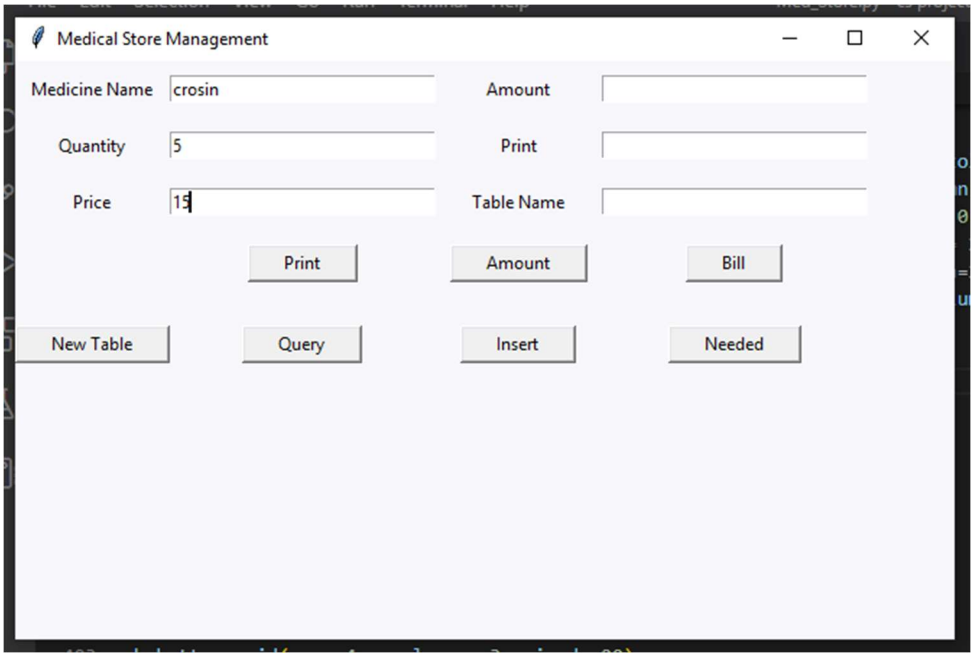
Price – <Enter Price>

Table Name - <Enter current table>

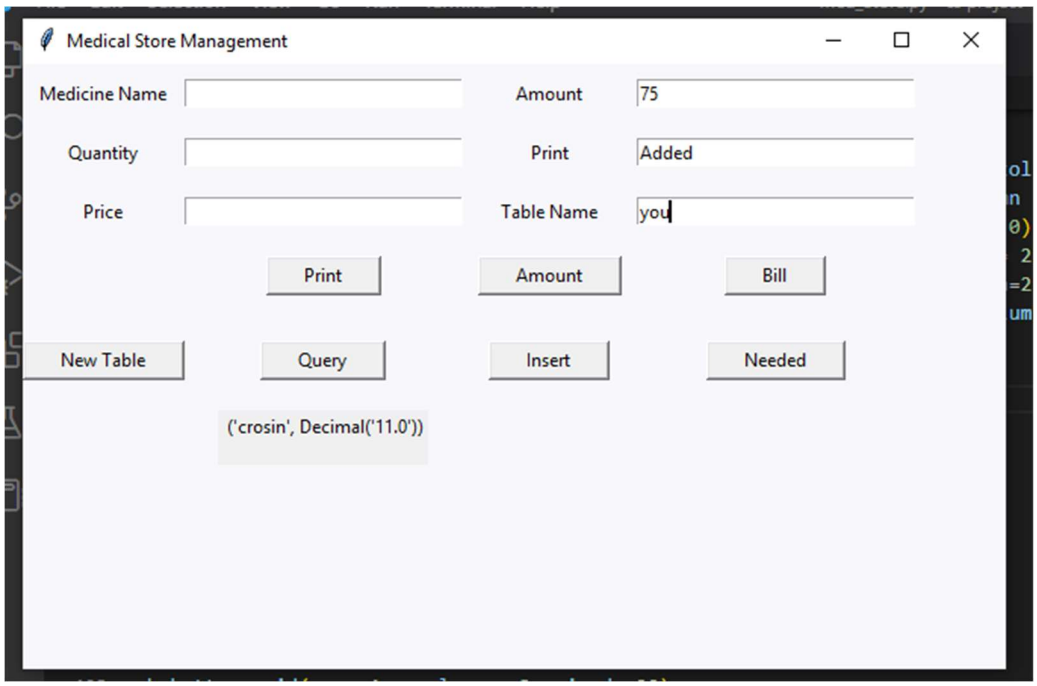
Print Button - <Enter medicine information in csv file when purchased>

Amount Button - <Calculates amount of purchased medicines>

Bill Button - <Subtracts Quantity of Purchased Medicine from main table >



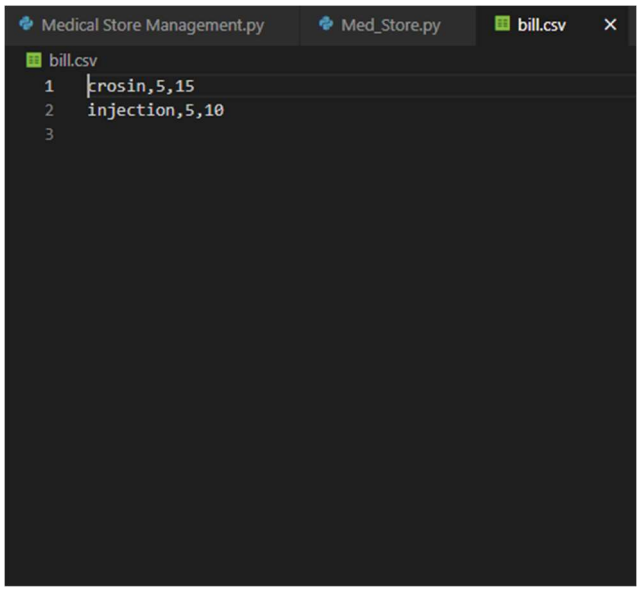
Print clicked



Amount = 5*15 = 75

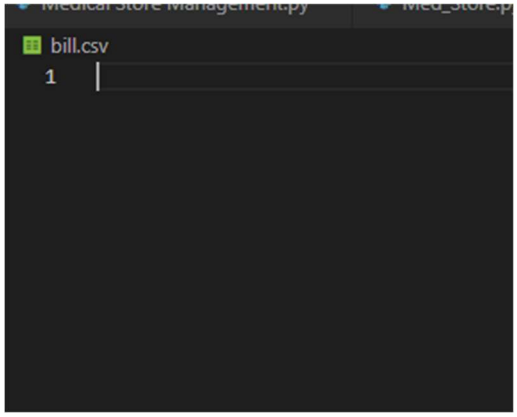
Medicine name and minimum selling price shown below query button

Minimum selling price for crossing is 11



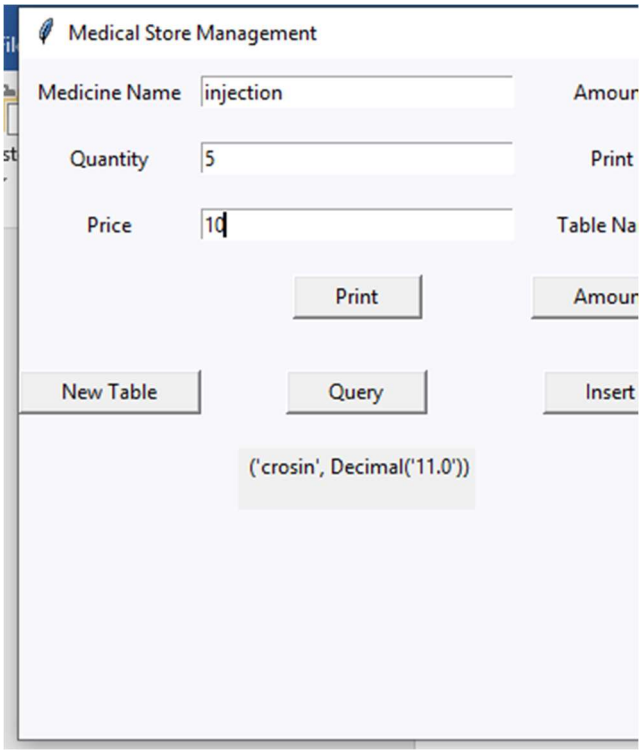
When Print Clicked , Record inserted in csv file bill.csv

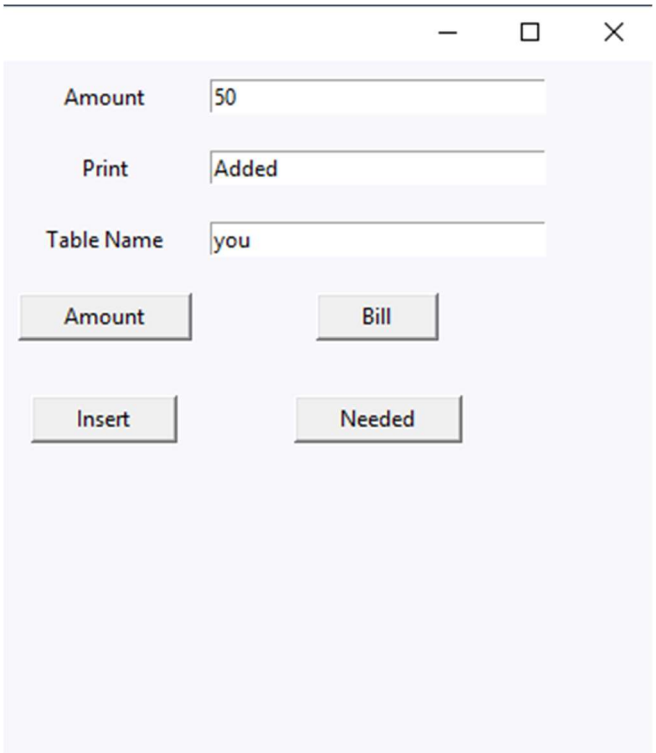
When Bill clicked

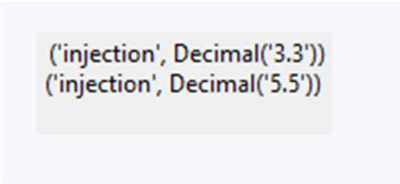


Prints Bill and clear csv file while subtracting quantity purchased from main table

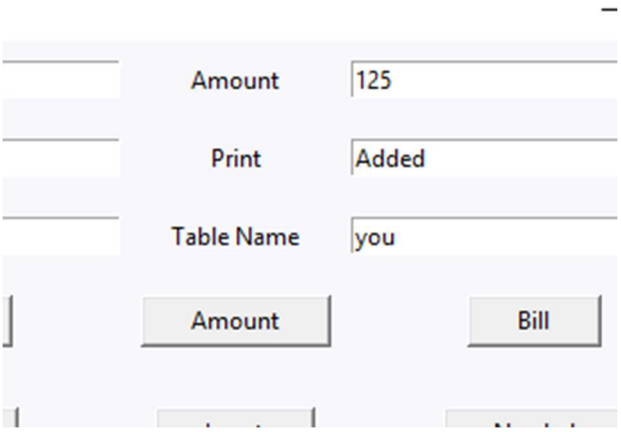
New Entries







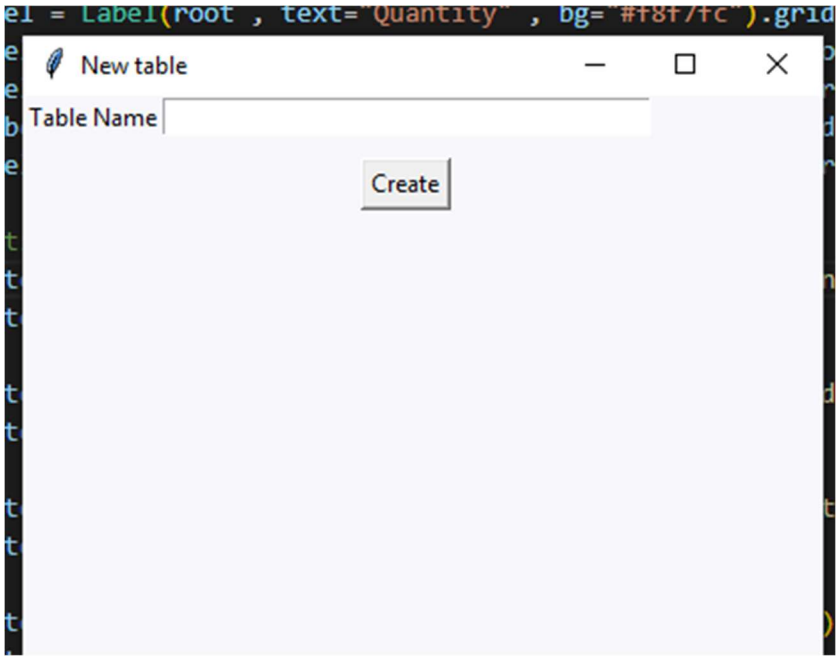
Minimum selling price of injection is 3.3 & 5.5 with margin of 10%



Amount Button – When click it gives total amount as 125

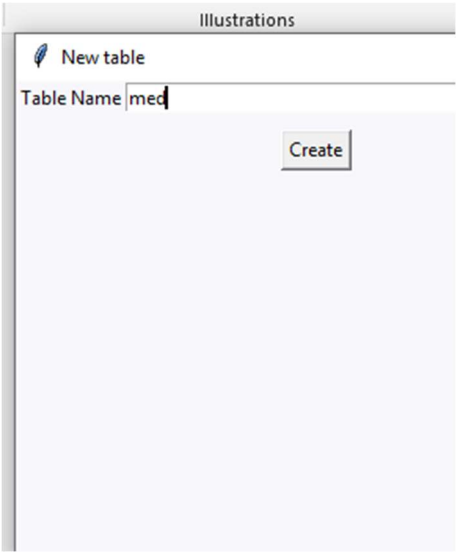
Amount of crosin + injection i.e., 5*15+5*10 = 125

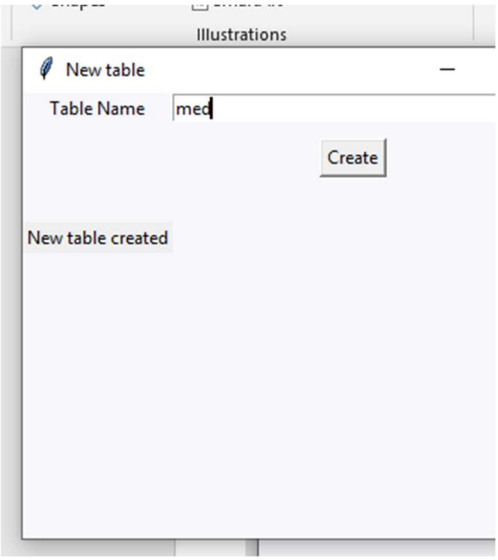
New Table Window –



Creates new table

When Create Button Clicked

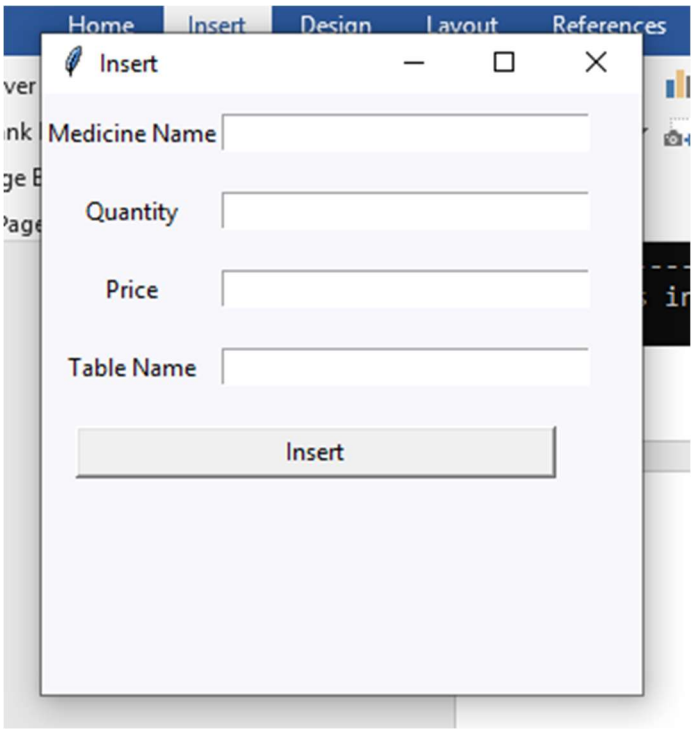




New Table med created

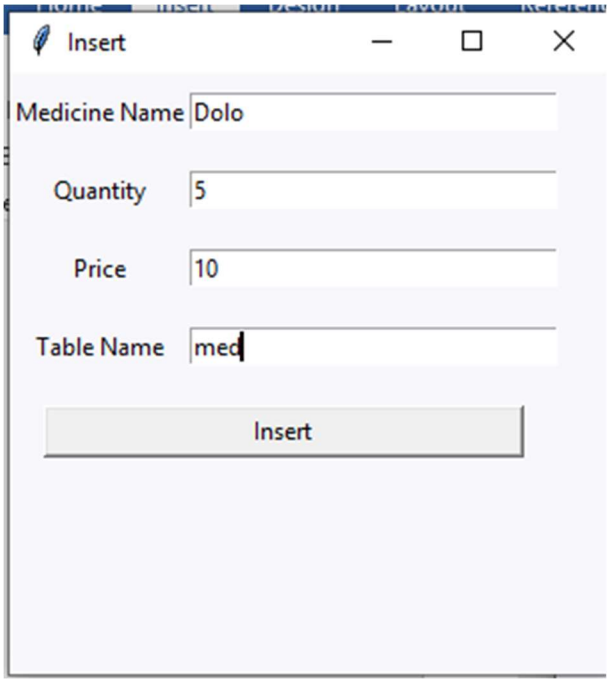
```
Database changed
mysql> show tables;
+---------------+
| Tables_in_med |
+---------------+
| boi           |
| med           |
| sales         |
| you           |
+---------------+
4 rows in set (0.08 sec)
```

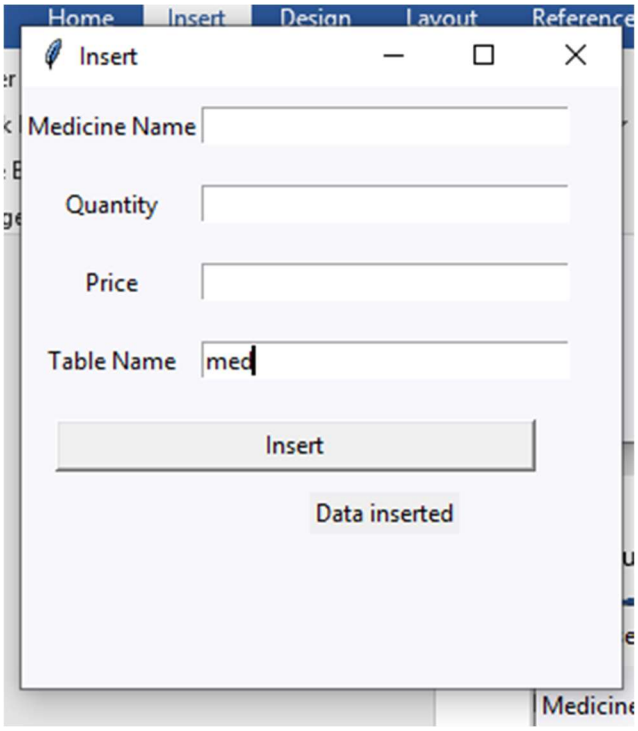
Insert Button -



Opens Insert window

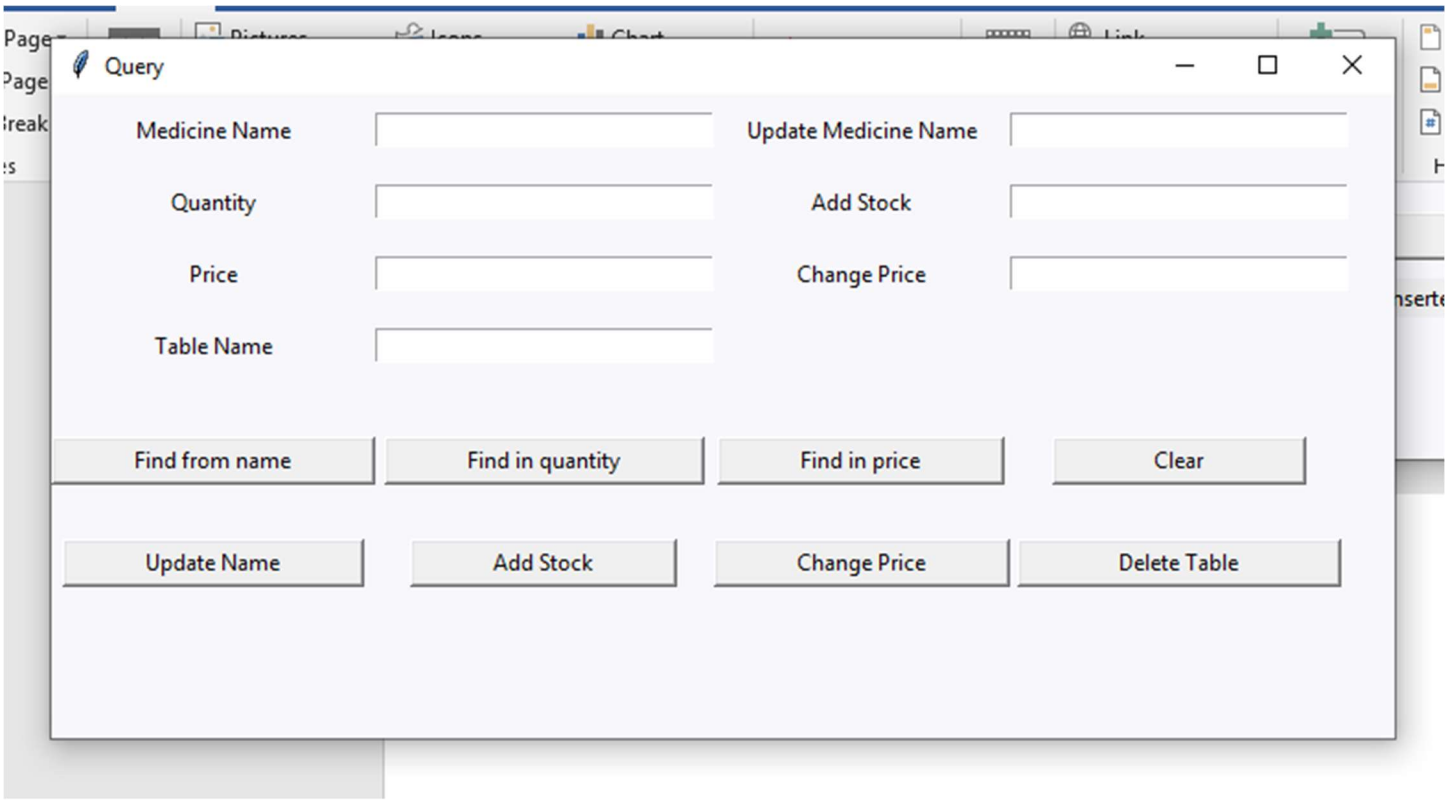
Insert Button –





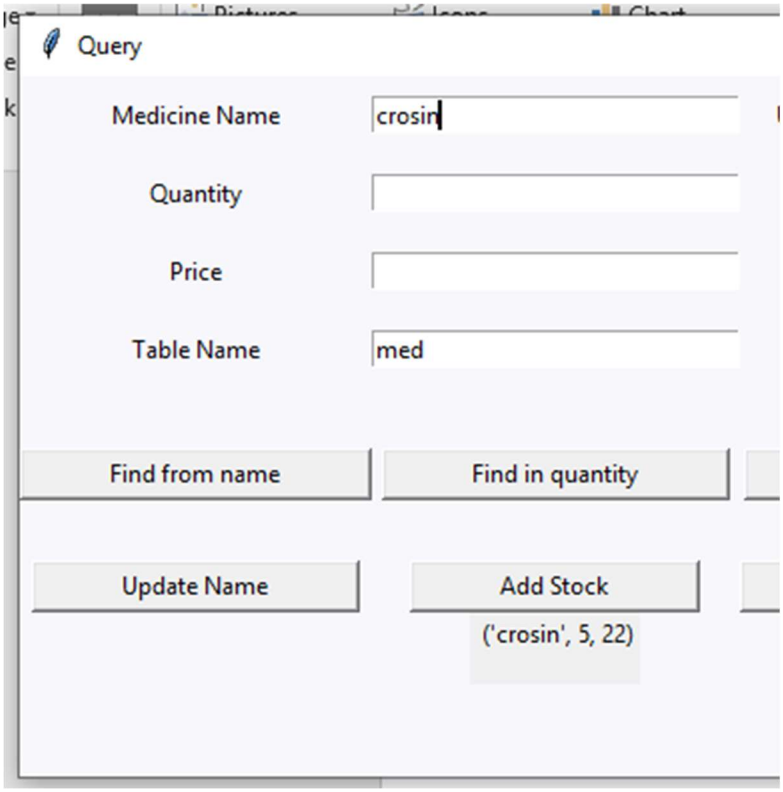
Data Insert is shown , hence successfully inserted in table med

Query Button –



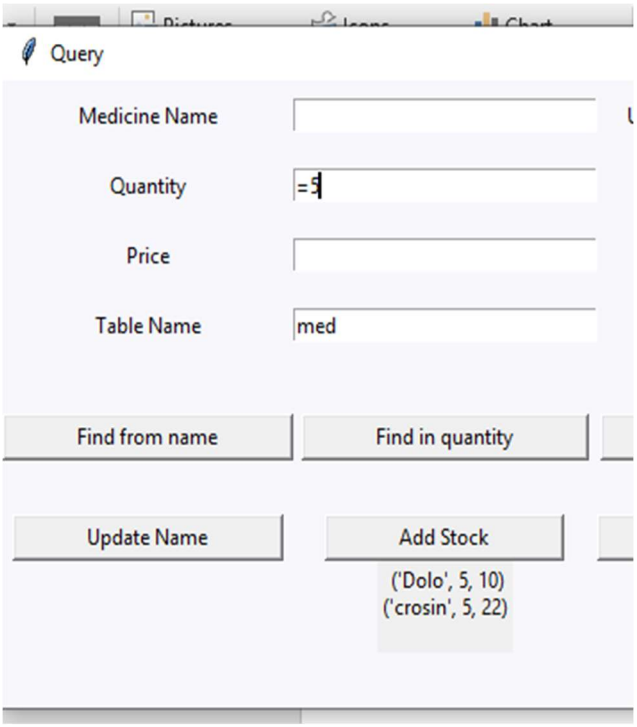
Opens Query Window

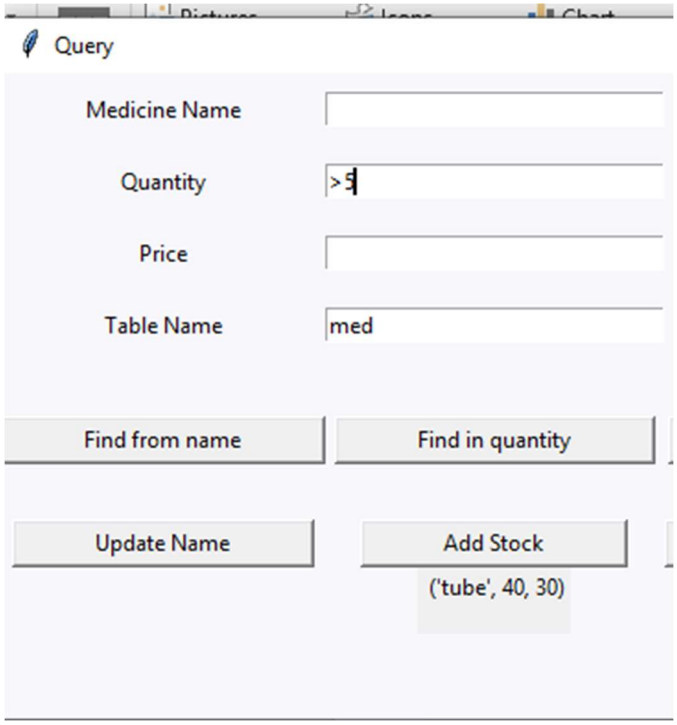
Using Find from name :



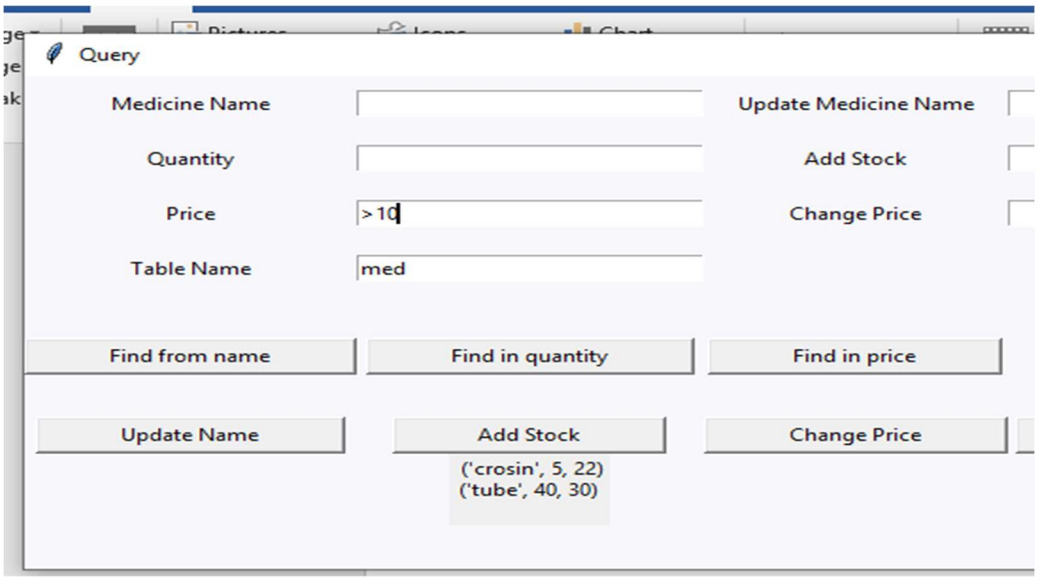
Output is shown below Add Stocks Button

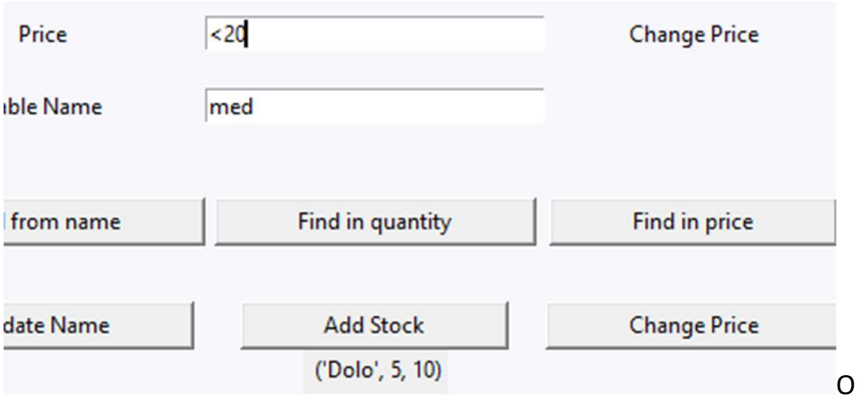
Using Find in Quantity Button:





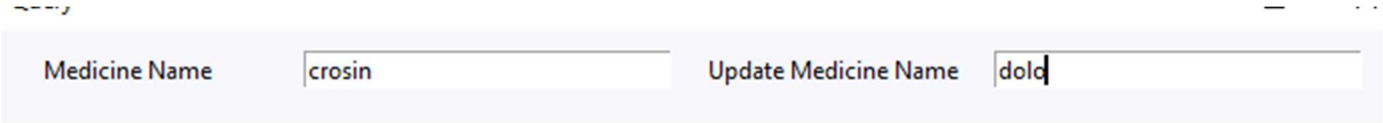
Using Find from Price Button:





Output

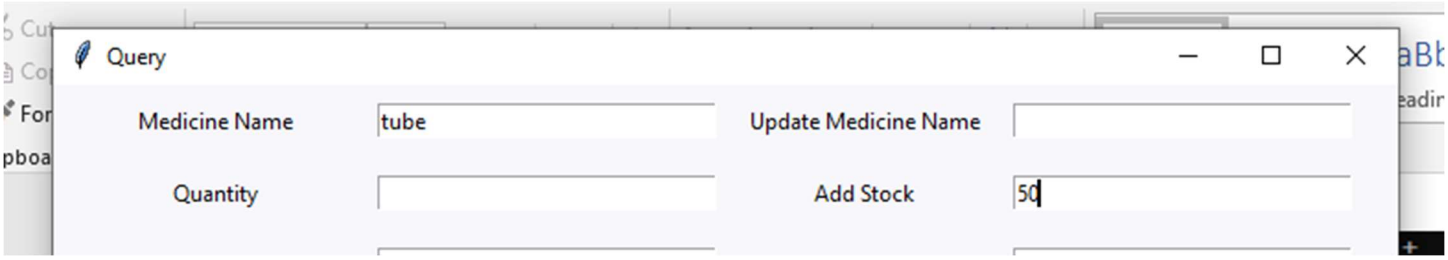
Using Update Name Button:



Name is updated from crosin to dolo

```
+---------------+
4 rows in set (0.08 sec)

mysql> select * from med;
+----------+----------+-------+
| Medicine | Quantity | Price |
+----------+----------+-------+
| Dolo     |        5 |    10 |
| crosin   |        5 |    22 |
| tube     |       40 |    30 |
+----------+----------+-------+
3 rows in set (0.00 sec)

mysql> select * from med;
+----------+----------+-------+
| Medicine | Quantity | Price |
+----------+----------+-------+
| Dolo     |        5 |    10 |
| dolo     |        5 |    22 |
| tube     |       40 |    30 |
+----------+----------+-------+
3 rows in set (0.00 sec)
```

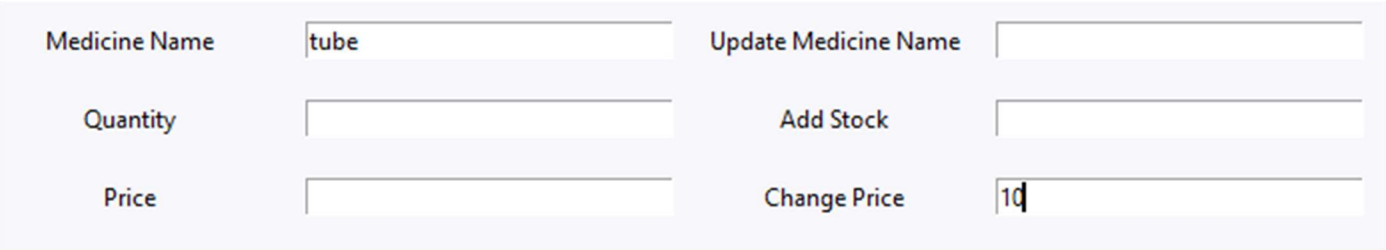
Add Stock Button:




```
mysql> select * from med;
+----------+----------+-------+
| Medicine | Quantity | Price |
+----------+----------+-------+
| Dolo     |        5 |    10 |
| dolo     |        5 |    22 |
| tube     |       50 |    30 |
+----------+----------+-------+
3 rows in set (0.00 sec)
```

Changes stock from 0 to 50 when Add Stock is clicked

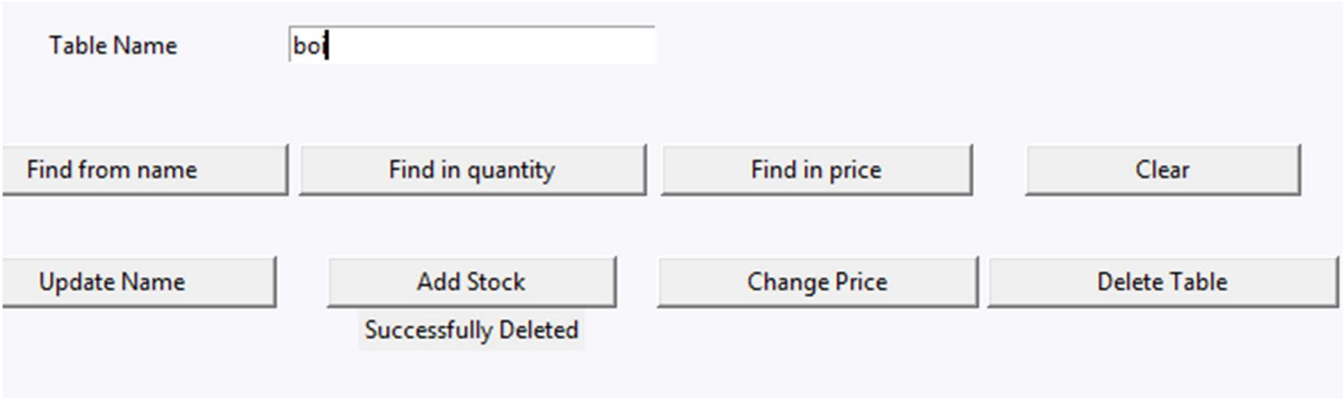
Change Price Button:




```
mysql> select * from med;
+----------+----------+-------+
| Medicine | Quantity | Price |
+----------+----------+-------+
| Dolo     |        5 |    10 |
| dolo     |        5 |    22 |
| tube     |       50 |    10 |
+----------+----------+-------+
3 rows in set (0.00 sec)
```

Changes tube price from 30 to 10

Delete table Buttons:

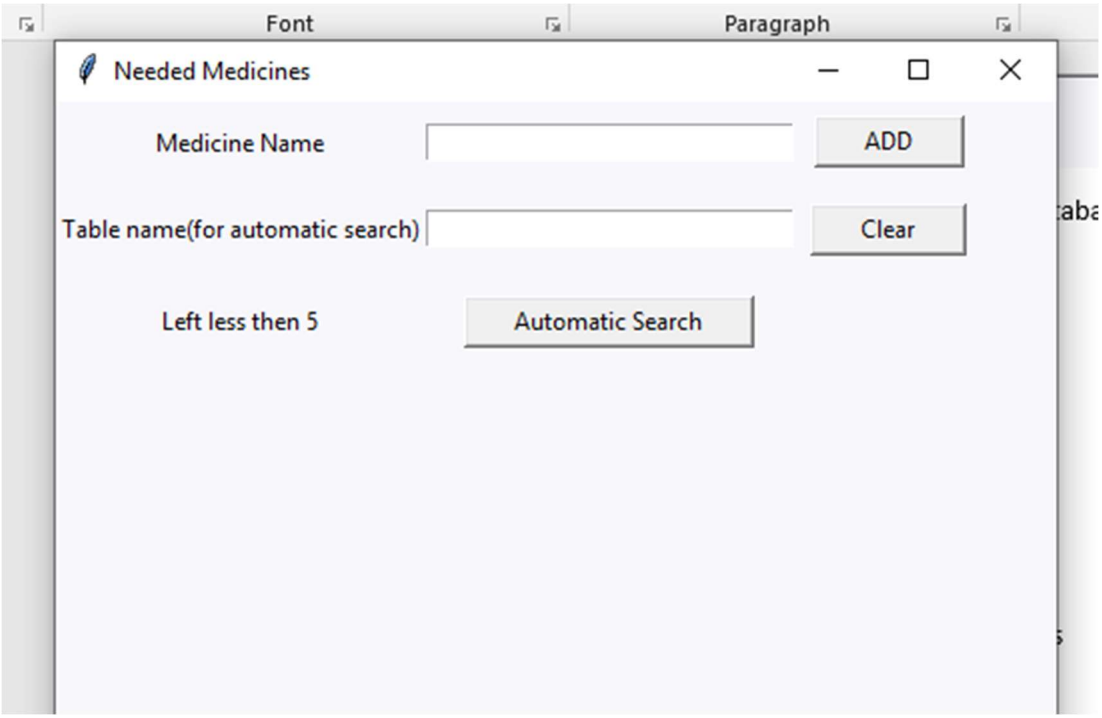



```
mysql> show tables;
+---------------+
| Tables_in_med |
+---------------+
| med           |
| sales         |
| you           |
+---------------+
3 rows in set (0.00 sec
```

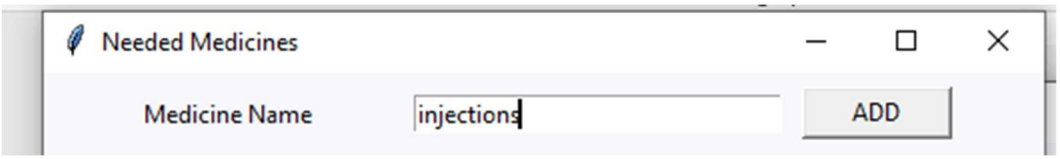
Deletes table from database

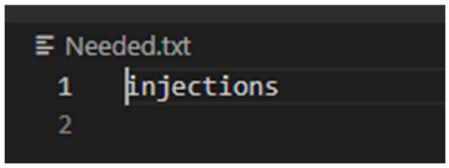
Clear Button: Clear all inserted values in text box and labels

Needed Button:

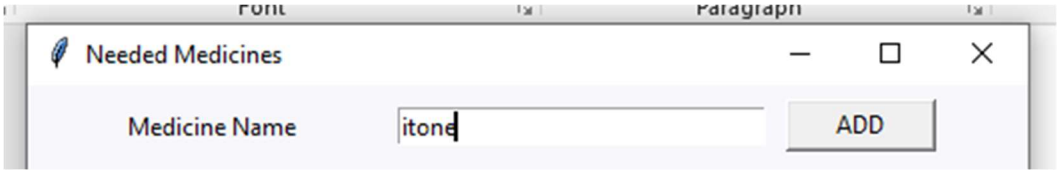


Needed – Add Button:

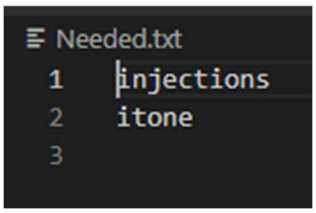




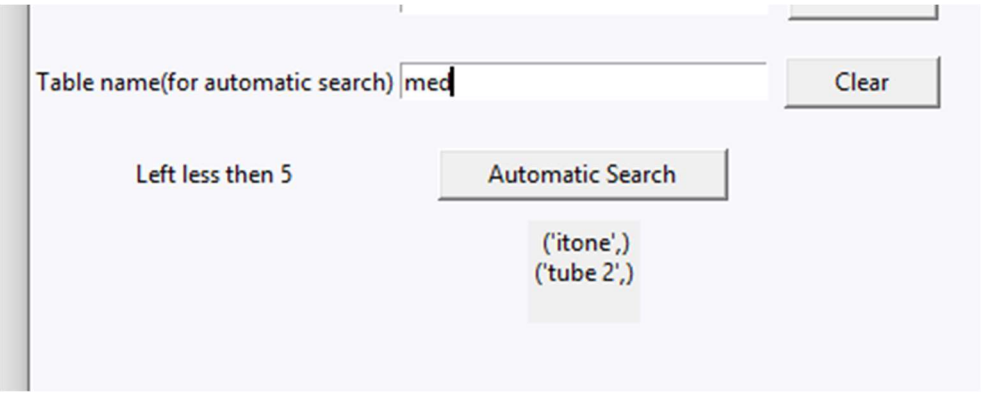
Adds given medicine in text file



Second Entry



Automatic Button : Gives medicine left less than 5



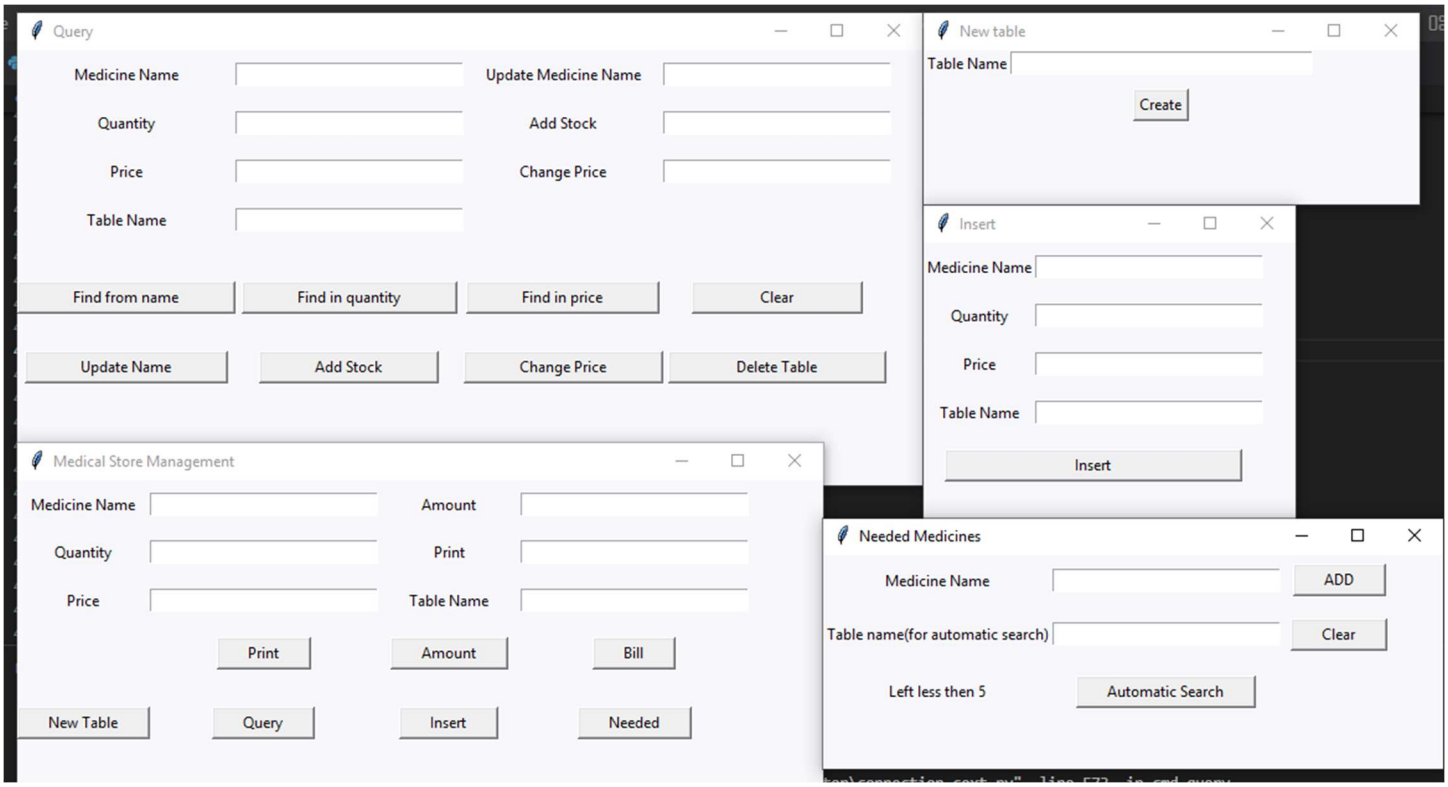
These are left less than 5 in table med

```
mysql> select * from med;
+----------+----------+-------+
| Medicine | Quantity | Price |
+----------+----------+-------+
| Dolo     |        5 |    10 |
| dolo     |        5 |    22 |
| tube     |       50 |    10 |
| itone    |        3 |     2 |
| tube 2   |        1 |     1 |
+----------+----------+-------+
5 rows in set (0.00 sec)

mysql>
```

Clearly we can see itone and tube 2 is left less than 5 in table med

Clear Button: Clear all entries and lables

All Windows of Program



This Program is help full in managing Medical store.

Program uses SQL (sequel) as main structure for its queries , data insertion , table creation , deleting tables and programs which can be useful such as automatic amount calculation , automatically providing information of medicines left less than 5 , manually adding medicines which are needed and other shown above.

Program provides minimum value for selling medicines (which is set 10% here) and gives total amount of medicine purchased by customer.

Program can also automatically subtracts quantity of medicines which are purchased.

Program also includes use of csv and text files.

Bibliography

- Youtube.com/watch/Tkinter Course – Create Graphics User Interface in Python Tutorial
  https://www.youtube.com/watch?v=YXPyB4XeYLA&t=5344s

- Stackoverflow.com