Requirements

Hardware ::-

- Printer to print the project report.
- Compact Disc.
- Ram 4 GB.

Software ::-

- Operating System Windows 10 or any main stream operating system that support python programming .
- Python installed
- SQL installed
- WPS word for report presentation.

General Description

“Bill Generating and Inventory Management System is a software used to comfort the shopkeeper or any one running their commercial entity , which allows them to seamlessly manage their data of their products with their prices , and vendors and also allows them to generate the bill for the customer to manage their billing task comfortably. With all this It also allows shopkeepers to add new vendors and to maintain the record of dues and payments of each vendor.

Here are some features of this Bill management system -

- **Add new product with it’s price in records easily.**
- **Manage stock and vendor dues .**
- **Add a new Vendor seamlessly.**
- **Store add the records in SQL database.**
- **Basket to add number of products.**
- **The final bill gets printed with total cost including GST .**
- **GUI interface for easy and efficient interaction between user and program.**

Source Code

```python
import tkinter as tk
import os
import ttkbootstrap as ttk
import mysql.connector
import tkinter.messagebox

#Database connection --
conn = mysql.connector.connect(host='localhost',user='root',database='bill',passwd='krpabhi9876')
cur = conn.cursor()

# GUI --
window = ttk.Window(themename='minty')
window.title('Bill Generating and Inventry Management System --')
window.geometry('230x120')
window.maxsize(230, 120)
window.minsize(230, 120)
window.iconbitmap('Untitled.ico')

class Gui:
    head_var = tk.StringVar()
    heading = ttk.Label(textvariable=head_var,
                        font=('Algerian', 18, 'bold'))
    add_frame = ttk.Frame(master=window)
    save_button = tk.Button(master=window,
                            text='Save',
                            font=('ComicSansMs', 18))

    generate_frame = ttk.Frame(master=window)
    add_to_list_button = tk.Button(master=window,
                                   text='Add to basket',
                                   font=('ComicSansMs', 16))
    generate_button = tk.Button(master=window,
                                text='Generate',
                                font=('ComicSansMs', 16))
    basket_table = ttk.Treeview(master=window, columns=('product', 'quantity', 'cost'), show='headings')
    basket_table.heading('product', text='Product')
    basket_table.heading('quantity', text='Quantity')
    basket_table.heading('cost', text='Cost')

def add_func():

    def save_func():
        pro_name = name_var.get()
        pro_price = price_var.get()
        v_name = vendor_box.get()
        v_quant = vend_quant.get()
        v_cost = vendor_cost_var.get()
        if pro_name=="" or pro_price==0 or v_quant==0 or v_cost==0:
            tkinter.messagebox.showerror(title="Input error !", message="Invalid input pls retry !")
        else:
            query = 'insert into product values(%s,%s,%s,%s)'
            val = (f'{pro_name}',f'{pro_price}',f'{v_name}',f'{v_quant}')
            cur.execute(query,val)
            v_name = vendor_box.get()
            query = f"select price_due from vendor where v_name = '{v_name}';"
            cur.execute(query)
            due = cur.fetchone()[0] + vendor_cost_var.get()
            query1 = f"update vendor set price_due = {due} where v_name = '{v_name}';"
            cur.execute(query1)
            conn.commit()
            tkinter.messagebox.showinfo('Saving file to record ', f'{pro_name} added to record !')

    window.minsize(480, 400)
    add_radio['state'] = 'disabled'
    generate_radio['state'] = 'enabled'
    Gui.generate_frame.forget()
    Gui.generate_button.forget()
    Gui.add_to_list_button.forget()
    Gui.basket_table.forget()
    Gui.basket_table.delete(*Gui.basket_table.get_children())
    Gui.head_var.set('Add new product')
    Gui.heading.pack()
    Gui.add_frame.pack()
    pr_name = ttk.Label(master=Gui.add_frame,
                        text='Name of product :: ',
                        font=('ComicSansMs', 14, 'bold'))
    pr_name.grid(row=0, column=0, pady=5, padx=4)
    name_var = tk.StringVar()
    name_entry = ttk.Entry(master=Gui.add_frame,
                           textvariable=name_var,
                           font=('arial', 14))
    name_entry.grid(row=0, column=1, pady=5, padx=4)
    pr_price = ttk.Label(master=Gui.add_frame,
                         text='Price of product :: ',
                         font=('ComicSansMs', 14, 'bold'))
    pr_price.grid(row=1, column=0, pady=5, padx=4)
    price_var = tk.IntVar()
```

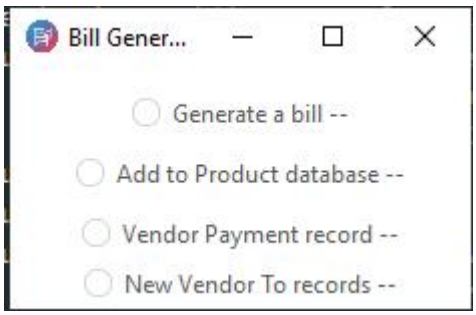
```
    price_entry = ttk.Entry(master=Gui.add_frame,
                            textvariable=price_var,
                            font=('arial', 14))
    price_entry.grid(row=1,column=1,padx=5,pady=4)
    vendor_lable = ttk.Label(master=Gui.add_frame,
                            text='Vendor :: ',
                            font=('ComicSansMs', 14, 'bold'))
    vendor_lable.grid(row=2, column=0, pady=5, padx=4)
    query = 'select * from vendor;'
    cur.execute(query)
    li = cur.fetchall()
    vendors = []
    for i in li:
        vendors.append(i[0])
    vendor_box = ttk.Combobox(master=Gui.add_frame,
                              values=vendors,
                              state='readonly',
                              font=('arial', 14))
    vendor_box.grid(row=2, column=1, padx=5, pady=4)
    vendor_quant = ttk.Label(master=Gui.add_frame,
                            text='Stock Added :: ',
                            font=('ComicSansMs', 14, 'bold'))
    vendor_quant.grid(row=3, column=0, pady=5, padx=4)
    vend_quant = tk.IntVar()
    vendor_quant_entry = ttk.Entry(master=Gui.add_frame,
                                   textvariable=vend_quant,
                                   font=('arial', 14))
    vendor_quant_entry.grid(row=3, column=1, pady=5, padx=4)
    vendor_cost_var = tk.IntVar()
    vendor_cost_Lable = ttk.Label(master=Gui.add_frame,
                                  text='Cost of stock added :: ',
                                  font=('ComicSansMs', 14, 'bold'))
    vendor_cost_Lable.grid(row=4,column=0,padx=5,pady=4)
    vendor_cost_Entry = ttk.Entry(master=Gui.add_frame,
                                   textvariable=vendor_cost_var,
                                   font=('arial', 14))
    vendor_cost_Entry.grid(row=4,column=1,padx=5,pady=4)
    Gui.save_button['command'] = save_func
    Gui.save_button.pack()


def generate_func():

    def generate():
        basket = Gui.basket_table.get_children()
        bill_string = 'Product  -   Quantity  -   Cost\n'
        cost_total = 0
        for i in basket:
            bill_basket = Gui.basket_table.item(i)['values']

            query = f"select quant_stock from product where pro_name = '{bill_basket[0]}';"
            cur.execute(query)
            stock = cur.fetchone()[0]- bill_basket[1]
            if stock <=0:
                query1 = f"delete from product where pro_name='{bill_basket[0]}';"
                cur.execute(query1)
                conn.commit()
            elif stock>0:
                query1 = f"update product set quant_stock = {stock} where pro_name = '{bill_basket[0]}';"
                cur.execute(query1)
                conn.commit()

            cost_total += bill_basket[2]
            bill_string = bill_string + f'{bill_basket[0]}    -    {bill_basket[1]}     -     {bill_basket[2]}\n'
        bill_string = bill_string + f'\nGoods and service Tax (G.S.T) 18% = {int(cost_total*0.18)}\nTotal Payable Amount=
        {cost_total + int(cost_total*0.18)}'
        bill = open('bill.txt', 'w')
        bill.write(bill_string)
        bill.close()
        os.startfile('bill.txt', 'open')
        query = "select pro_name,quant_stock from product;"
        cur.execute(query)
        stock_det = cur.fetchall()
        det = 'Stock Left \n '
        for i in stock_det:
            det+= f"{i[0]} - {i[1]}\n"
        tkinter.messagebox.showinfo(title='Stock',message=det)


    def basket_func(name, quantity):
        price = int(price_dict[name])*int(quantity)
        Gui.basket_table.insert(parent='', index=tk.END, values=(name, f'    {quantity}', f'    {price}'))
```

```
    # Data --
    query = 'select * from product'
    cur.execute(query)
    mydata = cur.fetchall()
    products = []
    price_dict = {}
    for i in mydata:
        products.append(i[0])
        price_dict[i[0]] = i[1]


    window.minsize(650, 550)
    add_radio['state'] = 'enabled'
    generate_radio['state'] = 'disabled'
    Gui.head_var.set('Generate bill ')
    Gui.heading.pack()
    Gui.add_frame.forget()
    Gui.save_button.forget()
    Gui.generate_frame.pack()
    product_label = ttk.Label(master=Gui.generate_frame, text='Product :: ', font=('ComicSansMs', 16, 'bold'))
    product_label.grid(row=0, column=0)
    product_box = ttk.Combobox(master=Gui.generate_frame,
                               values=products,
                               state='readonly')
    product_box.grid(row=0, column=1, padx=10, pady=10)
    product_label = ttk.Label(master=Gui.generate_frame, text='Quantity :: ', font=('ComicSansMs', 16, 'bold'))
    product_label.grid(row=2, column=0)
    quantity_spin = ttk.Spinbox(master=Gui.generate_frame,
                                from_=1,
                                to=100,
                                state='readonly')
    quantity_spin.grid(row=2, column=1, padx=10, pady=10)
    Gui.add_to_list_button.pack(pady=10)
    Gui.add_to_list_button['command'] = lambda: basket_func(product_box.get(), quantity_spin.get())
    Gui.basket_table.pack()
    Gui.generate_button.pack(pady=10)
    Gui.generate_button['command'] = generate

def vend_func():
    newWindow = ttk.Toplevel(window)
    newWindow.title("Vendor payment Mangement ---")
    newWindow.geometry("490x250")
    newWindow.iconbitmap('bitfield.ico')

    def update():
        v_name = vendor_box.get()
        query = f"select price_due from vendor where v_name = '{v_name}';"
        cur.execute(query)
        due = cur.fetchone()[0] - vendor_cost_var.get()
        query1 = f"update vendor set price_due = {due} where v_name = '{v_name}';"
        cur.execute(query1)
        conn.commit()
        vend_due_var.set(due)
        tkinter.messagebox.showinfo(title="Change Executed....",message="Chnges are executed....")

    def fetch():
        v_name = vendor_box.get()
        query = f"select price_due from vendor where v_name = '{v_name}';"
        cur.execute(query)
        due = cur.fetchone()[0]
        vend_due_var.set(due)

    vendor_lable = ttk.Label(master=newWindow,
                             text='Vendor :: ',
                             font=('ComicSansMs', 14, 'bold'))
    vendor_lable.grid(row=0, column=0, pady=5, padx=4)
    query = 'select * from vendor;'
    cur.execute(query)
    li = cur.fetchall()
    vendors = []
    for i in li:
        vendors.append(i[0])
    vendor_box = ttk.Combobox(master=newWindow,
                              values=vendors,
                              state='readonly',
                              font=('arial', 14))
    vendor_box.grid(row=0, column=1, padx=5, pady=4)
    vendor_cost_var = tk.IntVar()
    vend_due_var = tk.IntVar()
    vend_due_label = ttk.Label(master=newWindow,
                               textvariable=vend_due_var,
                               font=('ComicSansMs', 14, 'bold'))
    vend_due_text = ttk.Label(master=newWindow,
                              text='Amount Due :: ',
                              font=('ComicSansMs', 14, 'bold'))
```

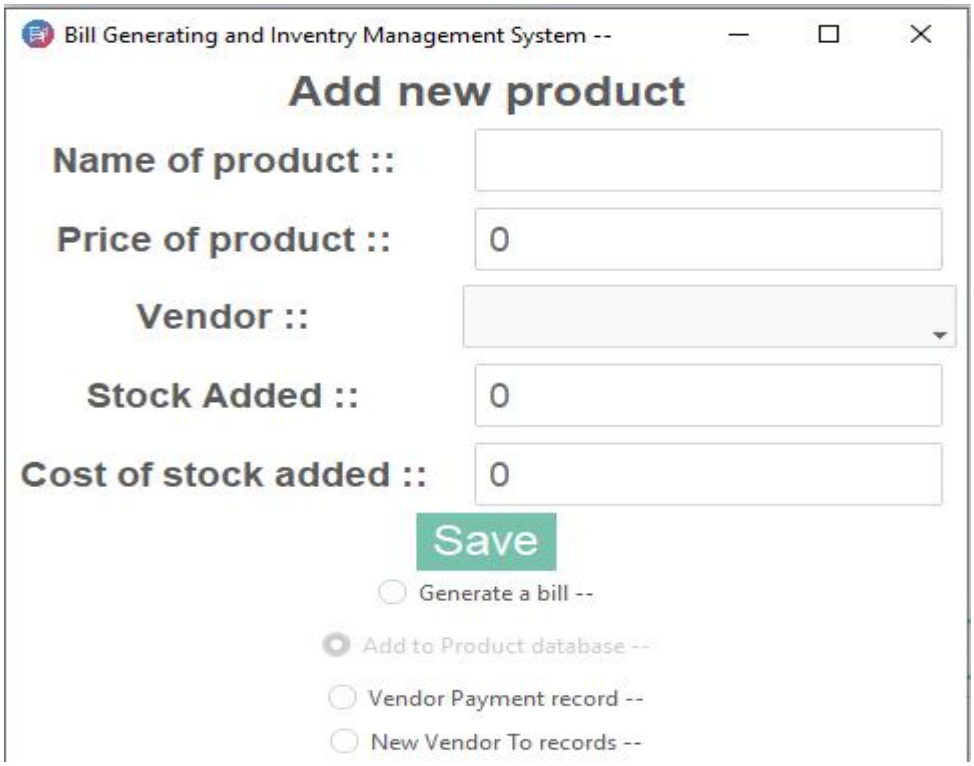
```
    vend_due_label.grid(row=2,column=1,padx=5,pady=4)
    vendor_cost_Lable = ttk.Label(master=newWindow,
                                  text='Price paid to Vendor :: ',
                                  font=('ComicSansMs', 14, 'bold'))
    vendor_cost_Lable.grid(row=3,column=0,padx=5,pady=4)
    vendor_cost_Entry = ttk.Entry(master=newWindow,
                                    textvariable=vendor_cost_var,
                                    font=('arial', 14))
    vendor_cost_Entry.grid(row=3,column=1,padx=5,pady=4)
    fetch_button = tk.Button(master=newWindow,
                             text='Fetch Due',
                             font=('ComicSansMs', 18),
                             command=fetch)
    update_button = tk.Button(master=newWindow,
                             text='Update',
                             font=('ComicSansMs', 18),
                             command=update)
    fetch_button.grid(row=4,column=0,padx=5,pady=4)
    update_button.grid(row=4,column=1,padx=5,pady=4)

def vend_new_func():
    def save():
        query = f"insert into vendor value('{vend_var.get()}',{due_var.get()});"
        cur.execute(query)
        conn.commit()
        tkinter.messagebox.showinfo(title="Record added ---",message=f"Vendor {vend_var.get()} added to record !!")

    newWindow1 = ttk.Toplevel(window)
    newWindow1.title("Add New Vendor ---")
    newWindow1.geometry("490x250")
    newWindow1.iconbitmap('Untitled.ico')
    vendor_lable = ttk.Label(master=newWindow1,
                             text='Vendor Name :: ',
                             font=('ComicSansMs', 14, 'bold'))
    vendor_lable.grid(row=0, column=0, pady=5, padx=4)
    vend_var = tk.StringVar()
    vendor_box = ttk.Entry(master=newWindow1,
                               textvariable=vend_var,
                               font=('arial', 14))
    vendor_box.grid(row=0, column=1, padx=5, pady=4)
    due_lable = ttk.Label(master=newWindow1,
                          text='Payment Due (if any) :: ',
                          font=('ComicSansMs', 14, 'bold'))
    due_lable.grid(row=1, column=0, pady=5, padx=4)
    due_var = tk.IntVar()
    due_box = ttk.Entry(master=newWindow1,
                             textvariable=due_var,
                             font=('arial', 14))
    due_box.grid(row=1, column=1, padx=5, pady=4)
    update_button = tk.Button(master=newWindow1,
                             text='Update',
                             font=('ComicSansMs', 18),
                             command=save)
    update_button.grid(row=2,column=1,padx=5,pady=4)


radio_var = tk.IntVar(value=4)
generate_radio = ttk.Radiobutton(text='Generate a bill --',
                                 value=0,
                                 variable=radio_var,
                                 command=generate_func)
add_radio = ttk.Radiobutton(text='Add to Product database --',
                            value=1,
                            variable=radio_var,
                            command=add_func)
vend_calc = ttk.Radiobutton(text='Vendor Payment record --',
                                 value=2,
                                 variable=radio_var,
                                 command=vend_func)
vend_new = ttk.Radiobutton(text='New Vendor To records --',
                                 value=3,
                                 variable=radio_var,
                                 command=vend_new_func)
vend_new.pack(side='bottom', pady=5)
vend_calc.pack(side='bottom', pady=5)
add_radio.pack(side='bottom', pady=10)
generate_radio.pack(side='bottom', pady=5)
window.mainloop()
conn.close()
```

Output

First Window:

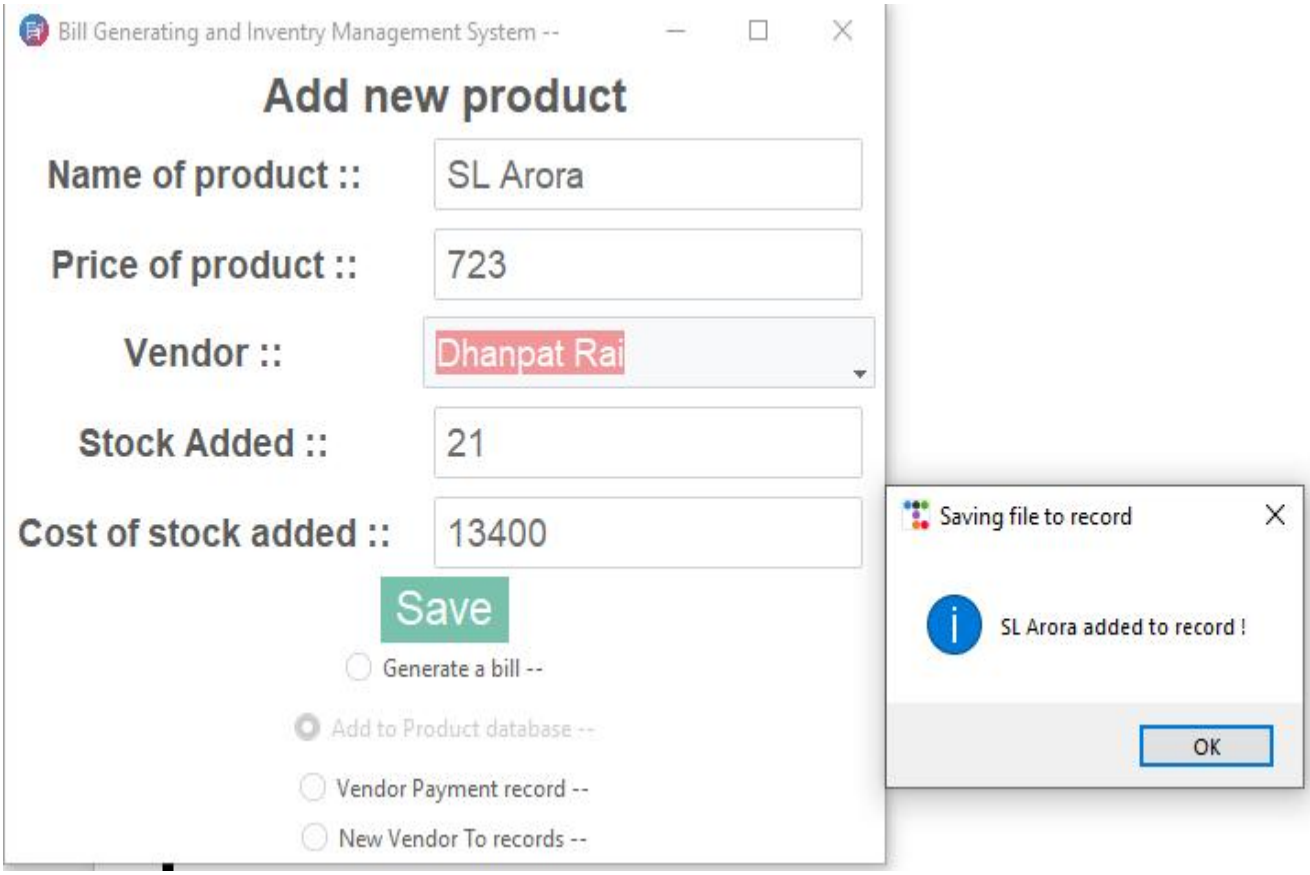


Adding product to records :



-- Before :

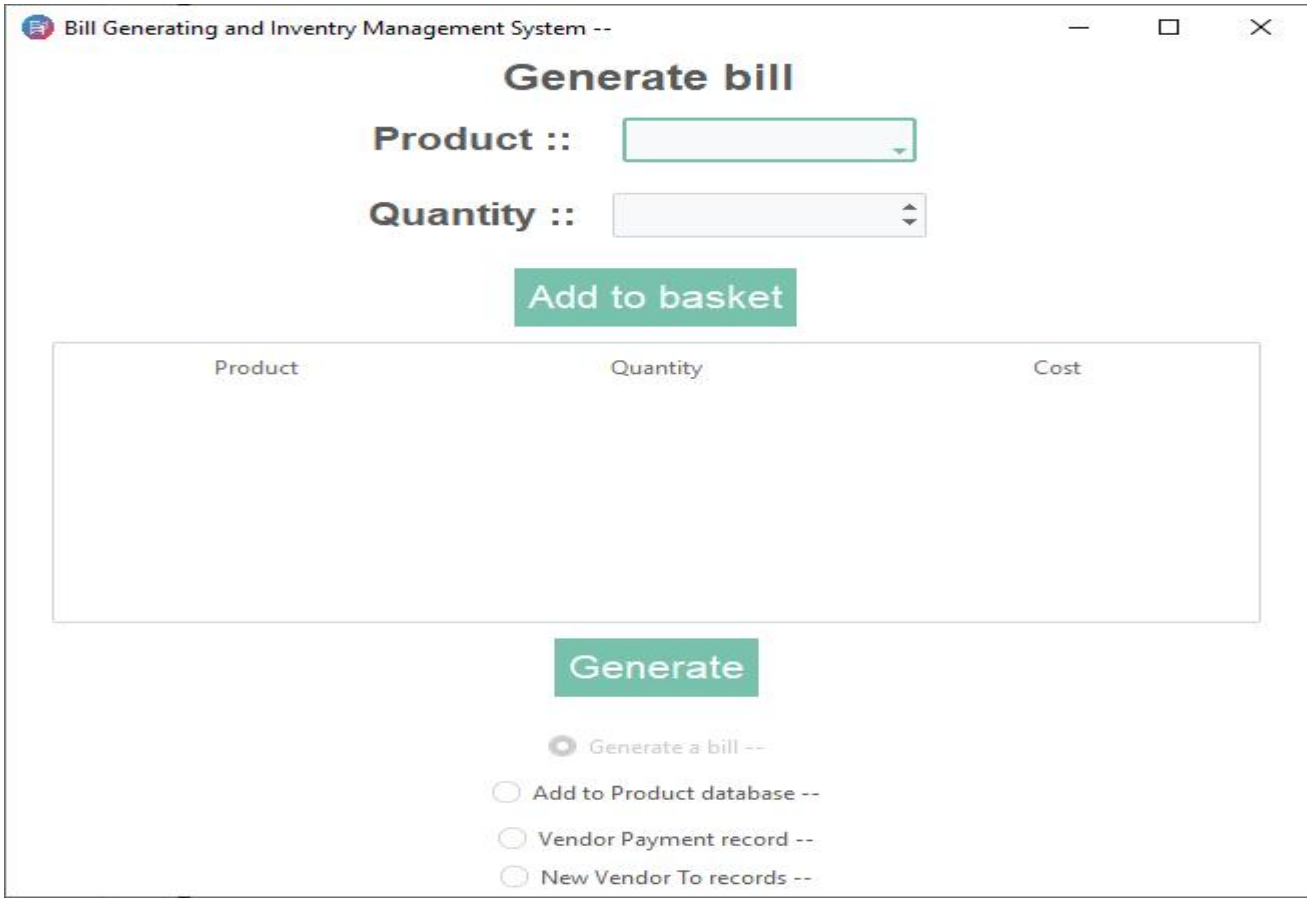
```
mysql> select * from product;
+-------------------+-------+--------------+-------------+
| pro_name          | price | vendor       | quant_stock |
+-------------------+-------+--------------+-------------+
| RS Aggarwal       |   397 | S-Chand      |          13 |
| HC Verma          |   543 | Dhanpat Rai  |           8 |
| DC Pandey         |   432 | Oxford       |          11 |
| Pradeep Chemistry |   479 | S-Chand      |          13 |
| Jee 5 year PYQ    |   326 | Cambridge    |          12 |
| PW Module         |   342 | RS Limited   |          21 |
| RD Sharma         |   457 | Dhanpat Rai  |          26 |
| Sample Papers     |   128 | Surya Prints |          19 |
| Physics Galaxy    |   670 | S-Chand      |           9 |
+-------------------+-------+--------------+-------------+
9 rows in set (0.02 sec)
```

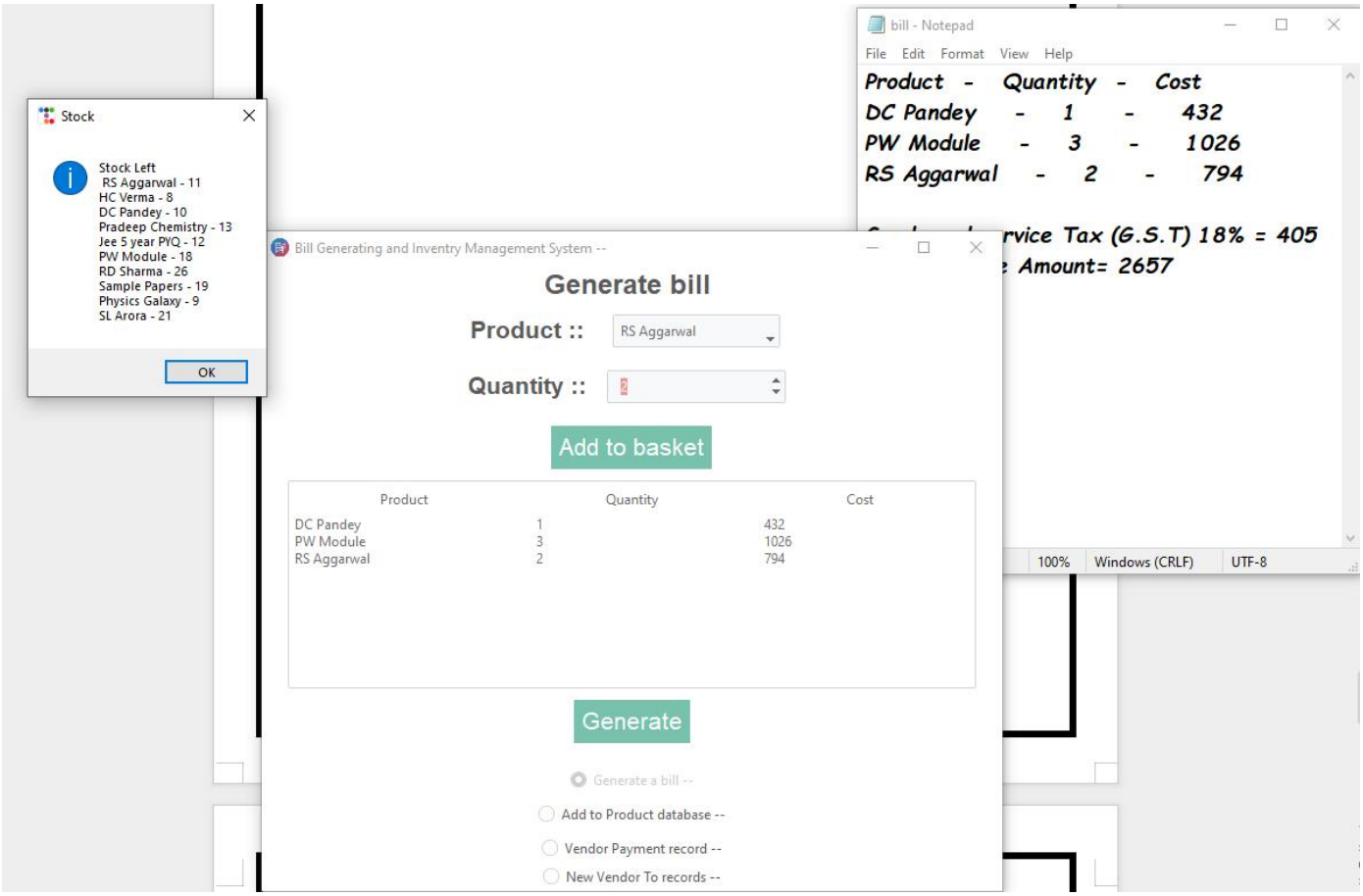


-- After :

```
mysql> select * from product;
+--------------------+-------+--------------+-------------+
| pro_name           | price | vendor       | quant_stock |
+--------------------+-------+--------------+-------------+
| RS Aggarwal        |   397 | S-Chand      |          13 |
| HC Verma           |   543 | Dhanpat Rai  |           8 |
| DC Pandey          |   432 | Oxford       |          11 |
| Pradeep Chemistry  |   479 | S-Chand      |          13 |
| Jee 5 year PYQ     |   326 | Cambridge    |          12 |
| PW Module          |   342 | RS Limited   |          21 |
| RD Sharma          |   457 | Dhanpat Rai  |          26 |
| Sample Papers      |   128 | Surya Prints |          19 |
| Physics Galaxy     |   670 | S-Chand      |           9 |
| SL Arora           |   723 | Dhanpat Rai  |          21 |
+--------------------+-------+--------------+-------------+
10 rows in set (0.00 sec)
```

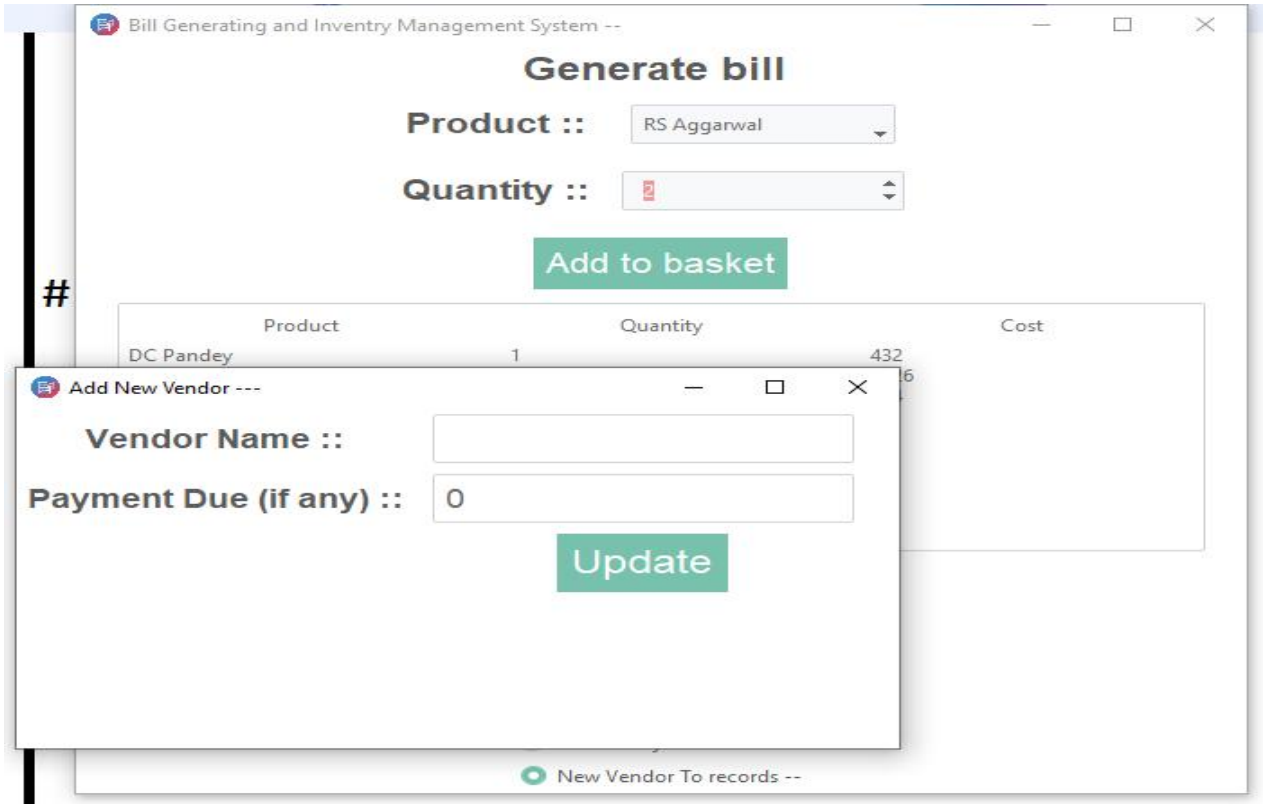
Generating Bill:





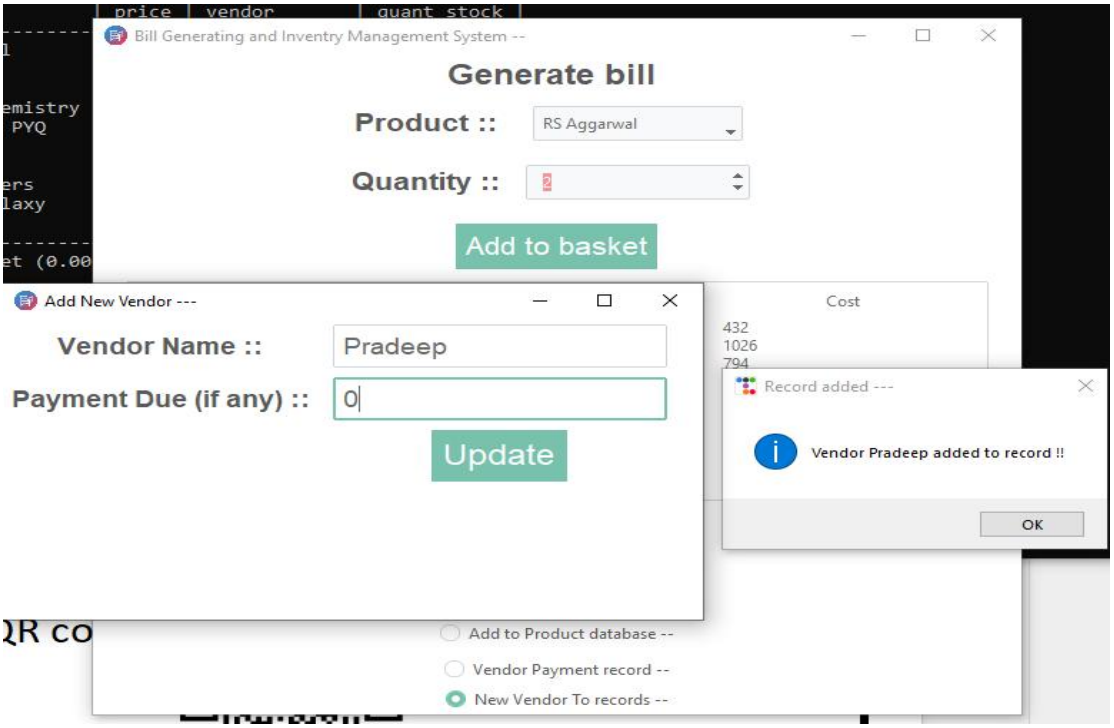
--" You can add products in basket , and after pressing Generate bill will be generate and left stock will be Shown , also the stock in database server will be automatically changed ."

Add new Vendor to Database ::



-- Before:

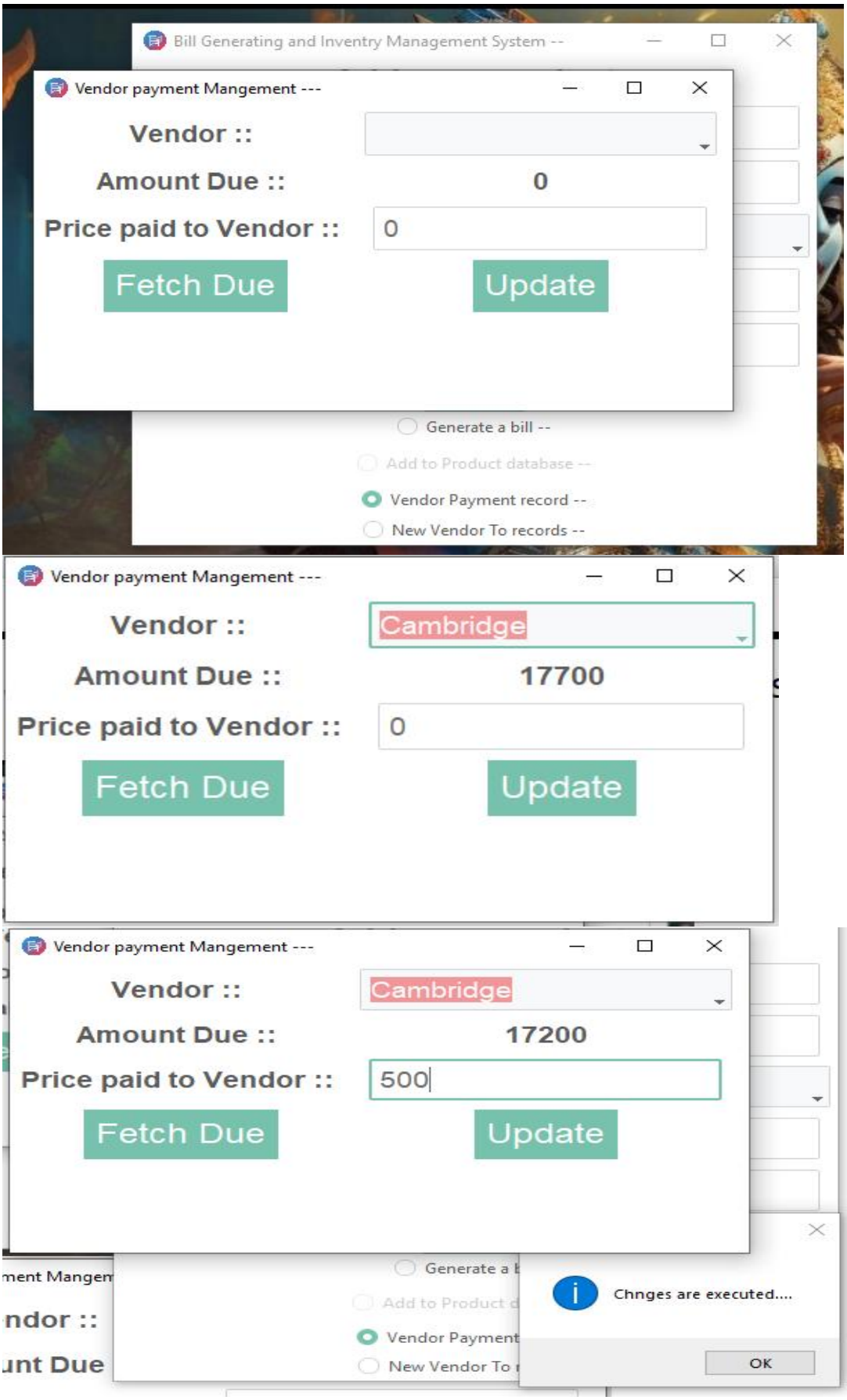
```
mysql> select * from vendor;
+--------------+-----------+
| v_name       | price_due |
+--------------+-----------+
| RS Limited   |      9600 |
| Dhanpat Rai  |     24700 |
| S-Chand      |     25500 |
| Oxford       |     10000 |
| Cambridge    |     17700 |
| Surya Prints |      1746 |
+--------------+-----------+
6 rows in set (0.00 sec)
```



-- After:

```
mysql> select * from vendor;
+--------------+-----------+
| v_name       | price_due |
+--------------+-----------+
| RS Limited   |      9600 |
| Dhanpat Rai  |     24700 |
| S-Chand      |     25500 |
| Oxford       |     10000 |
| Cambridge    |     17700 |
| Surya Prints |      1746 |
| Pradeep      |         0 |
+--------------+-----------+
7 rows in set (0.00 sec)
```

Dues and payment management of Vendors::



Limitations and scope of improvements

- This program do not indicate the unit for the pricing like per dozen , per KG and more , this can also be added to make it more helpful in real life practical scenarios.
- With increase in Number of vendors and products it will be tough to find desired result so something like searching and iteration among list can make it more easy to use in actual scenarios.