***University Institute of Engineering and Technology, Kanpur***



**INVENTORY MANAGEMENT SYSTEM**

**SUMMER TRAINING PROJECT**

**SUBMITTED BY**

**ASHWANI SHARMA(135)**

**ACKNOWLEDGEMENT**

It has indeed been a great privilege for me to have **Mr. KULDEEP SINGH**, Electronics and ICT Academy, Indian Institute of Technology, Kanpur, as my mentor for this project. I take this opportunity to express my utmost gratitude to him. I am indebted to him for his timely and valuable advice throughout the summer training program.

**Ashwani sharma**

**B-Tech IT (2016-2020)**

**University Institute of Engineering and Technology, Kanpur**

**TABEL OF CONTENTS**

**Topic Pg no**

1. Summer training Certificate 4

2. Project Letter 5

3. Abstract 6

4. Introduction 7

5. Methodology 8

6. Tools and software used 33

7. Result and application 34

8. Future Aspect 35

9. References 36

**ABSTRACT**

The main objectives of inventory management are operational and financial. The operational objectives mean that the materials and spares should be available in sufficient quantity so that work is not disrupted for want of inventory. The financial objective mean that investments in inventories should not remain idle and minimum working capital should be locked in it.

The followings are the objectives of inventory management:

1. To avoid both overstocking and under-stocking of inventory.

2. To keep materials cost under control so that they contribute in reducing cost of production and overall cost.

3. To eliminate duplication in ordering or replenishing stocks. This is possible with the help of centralising purchases.

4.To design proper organisation for inventory management. Clear cut accountability should be fixed at various levels of the organisation.

**INTRODUCTION**

The inventory management ensures that the company always has the required materials and products in hand while keeping the cost as low as possible. Inventory Management refers to the process of supervising and controlling the stock items of a company. Inventory Management Systems are used by firms that either sell a product or manufacture a product for the purpose of accounting all the tangible goods that allow for a sale of a finished product, or parts for making a product. This inventory management system can be used to store the details of the inventory, update the inventory based on the sale details, generate receipts for sales, generate sales and inventory reports periodically. This inventory management software has one module, Admin. Admin has the authority to add, update and delete an inventory. This inventory management software also has its own intelligently managed support system. This intelligent support system allows admin to view and manage various inventories.

**METHODOLOGY**

**1. Software Installation:** The first step of the project is to install the following software:

a) Python-python 3.7

b)Jupyter Notebook-pip install jupyter notebook

c)Pandas-pip install pandas

d)Numpy-pip install numpy

e)Matplotlib-pip install matplotlib

**2. Import module**: The first step after installation is to import the following modules to :

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

from matplotlib.pyplot import rcParams

rcParams['figure.figsize']=15,6

.

TABLE1:PRODUCT

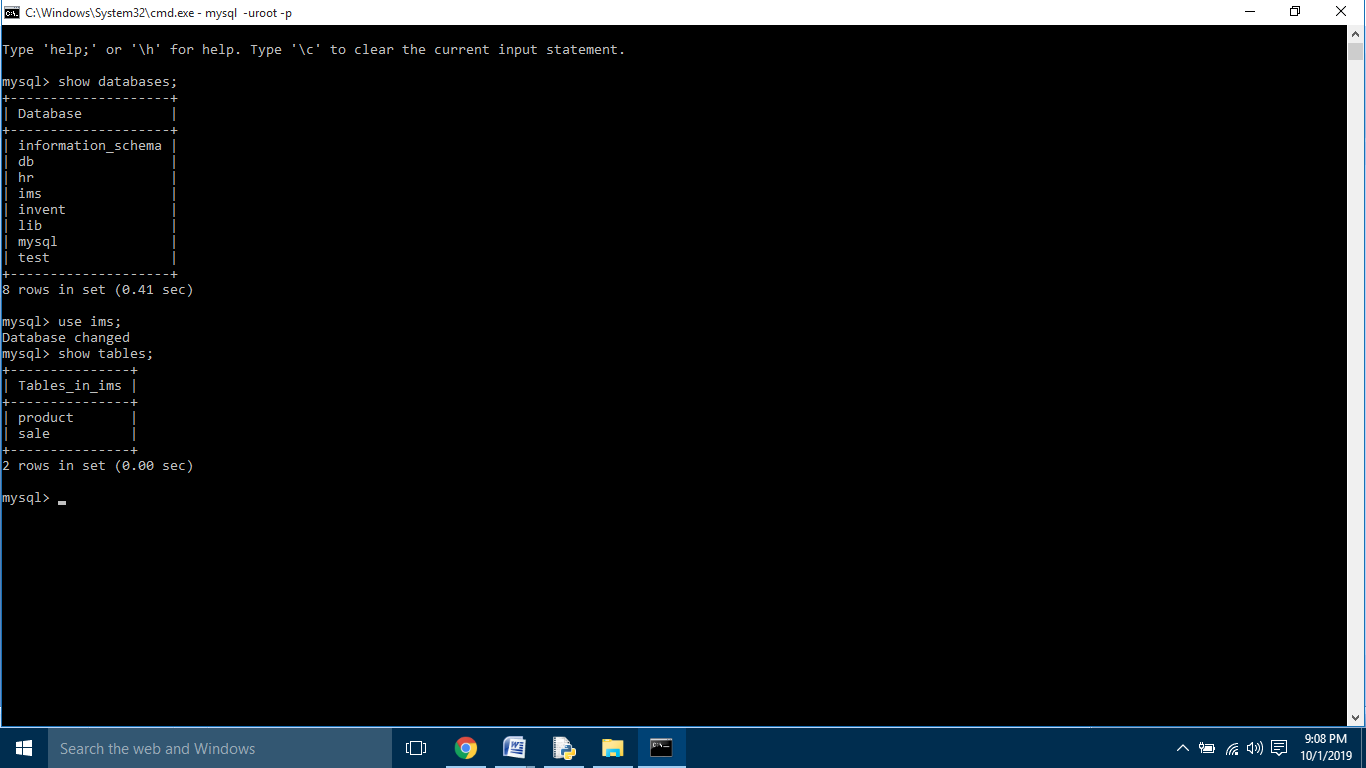
TABLE2:SALES

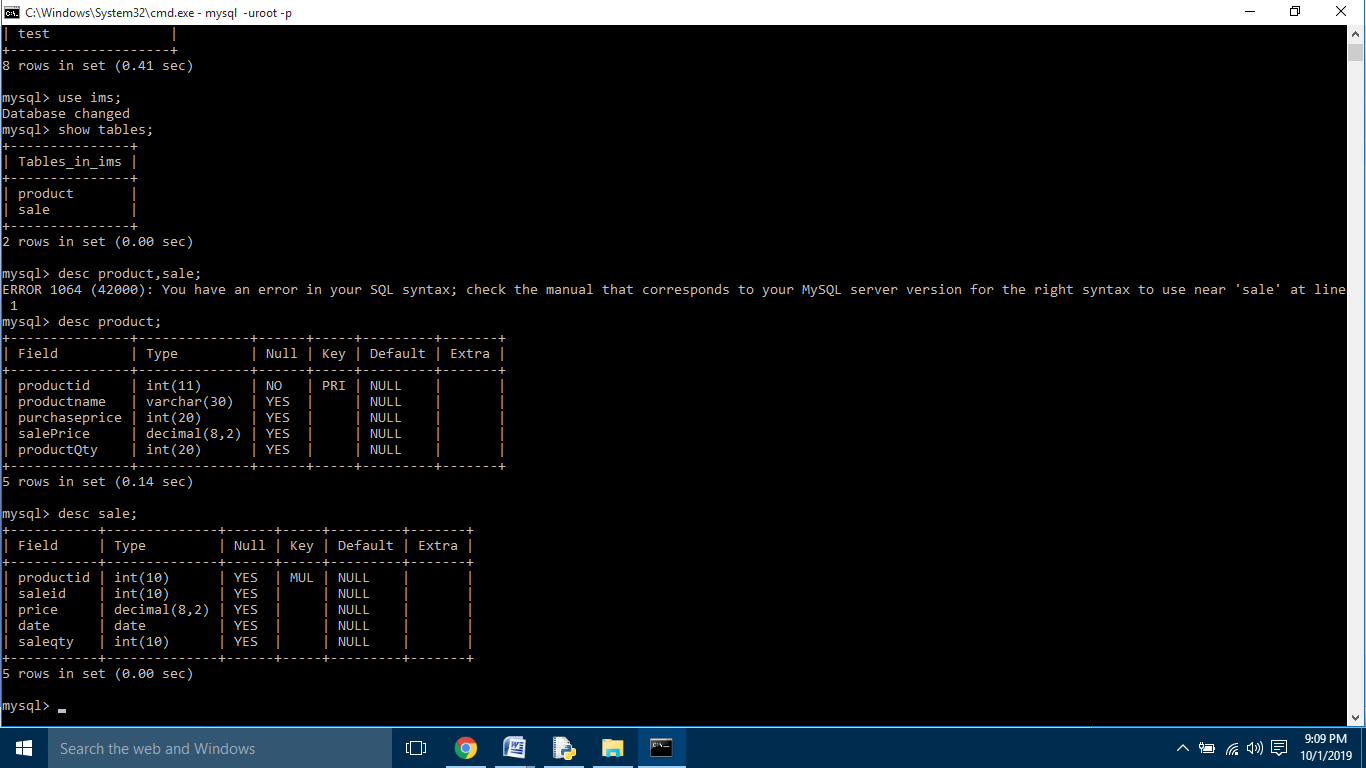
Command to start mysql server using terminal:

MySQL queries used to create both the tables:

1. create table product(productid varchar(20) primary key,product\_namevarchar(20),product\_price varchar(20),saleprice\_varchar(20),product\_quantity varchar(20));

ii)create table sale(sale\_id varchar(20),product\_id varchar(20),sale\_price varchar(20),sale\_quantity varchar(20) ON DELETE CASECADE);





**4. Connecting Database to GUI window**

Next we have to connect the guis to the Pymysql database using PHP so that upon entering the correct details data inserted in tables whereas on entering incorrect details it shows and error message.

**Python code:**

conn= pymysql.connect(host='localhost',user='root',password='',db='ims')

a=conn.cursor()

a.execute("insert into product values('"+pid+"','"+pn+"','"+pp+"','"+sp+"','"+pq+"')")

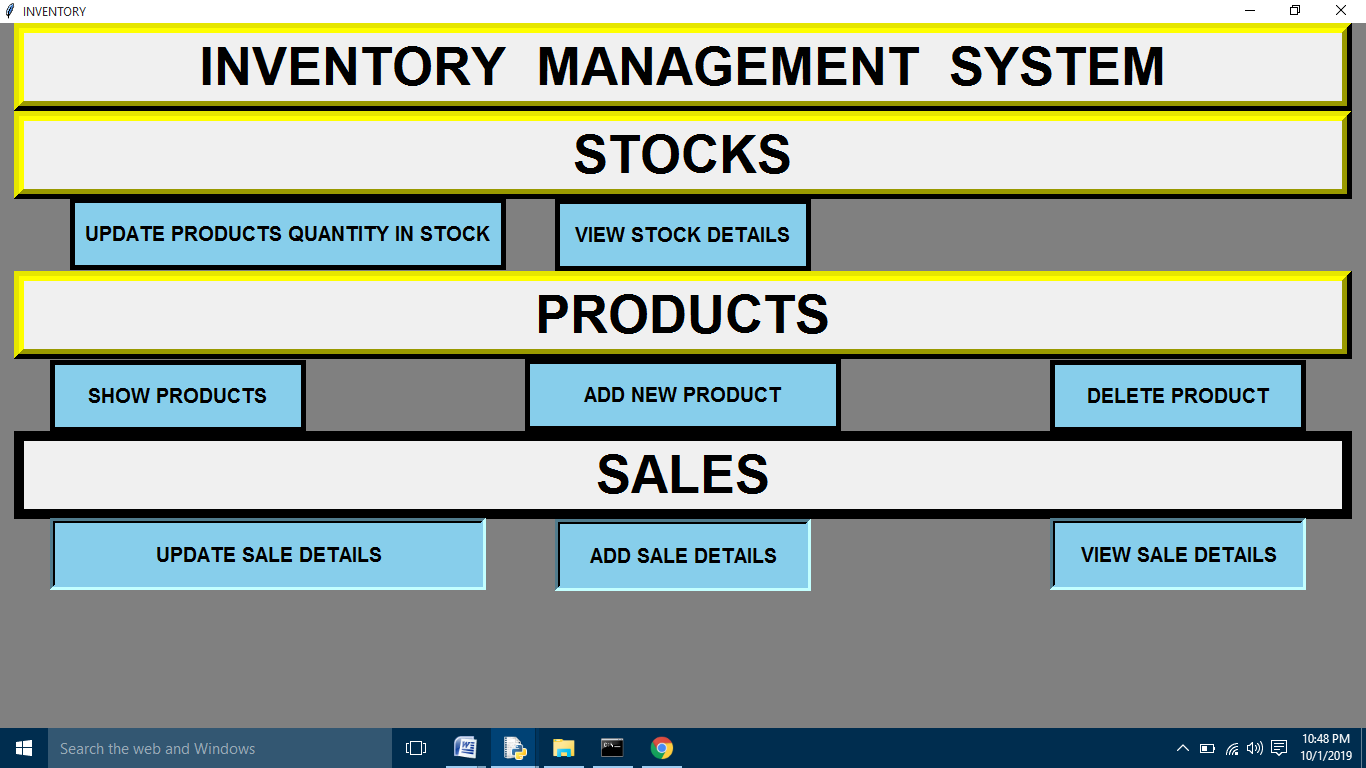
conn.commit()

messagebox.showinfo('submitted')

except:

conn.rollback()

conn.close()



**5. Creating Stock gui windows:** Next we create stock gui windows through which we can update product quantity in stock and also view stock details.

**Code to update stock quantity:**

def stock\_1():

def stock\_insert():

pid=product\_id.get()

pq=product\_quantity.get()

try:

conn= pymysql.connect(host='localhost',user='root',password='',db="ims")

a=conn.cursor()

a.execute("update product set productQty='"+pq+"' where productid='"+pid+"'")

conn.commit()

messagebox.showinfo('submitted')

except:

conn.rollback()

messagebox.showinfo('Not submit')

conn.close()

top=Toplevel()

top.title('NEW Window')

Mframe=Frame(top,bg="yellow",width=1000,height=800,relief='raise',bd=10)

Mframe.pack(padx=50,pady=20)

lb2=Label(Mframe,text="Product\_id",width=15)

lb2.grid(row=1,column=0,padx=10,pady=10)

product\_id=StringVar()

tb2=Entry(Mframe,textvariable=product\_id)

tb2.grid(row=1,column=1)

lb2=Label(Mframe,text="Product\_Quantity",width=15)

lb2.grid(row=2,column=0,padx=10,pady=10)

product\_quantity=StringVar()

tb2=Entry(Mframe,textvariable=product\_quantity)

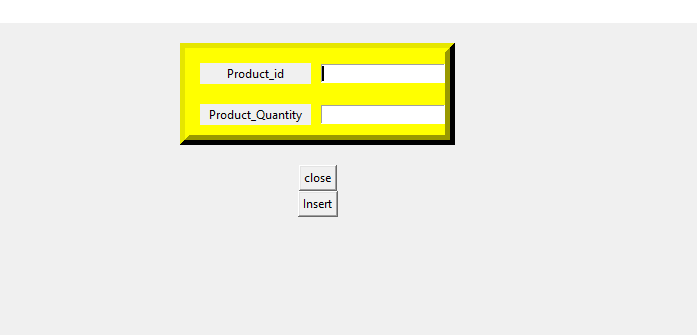
tb2.grid(row=2,column=1)

b1 = Button(top, text='close', command=top.destroy)

b1.pack()

b2 = Button(top, text='Insert', command=stock\_insert)

b2.pack()



**Code to display stock details:**

def stock\_2():

top=Toplevel()

top.geometry("720x320")

top.title('NEW Window')

top.config(background='gray')

sb = Scrollbar(top)

sb.pack(side = RIGHT, fill = Y)

mylist = Listbox(top,yscrollcommand = sb.set)

def stock\_display():

conn = pymysql.connect(host='localhost', user='root', password='', db='ims')

a = conn.cursor()

a.execute("select productid,productname,productQty

from product")

results = a.fetchall()

count = a.rowcount

if count > 0:

for row in results:

L=Label(top,text=row,font="Verdana20bold",width=30,relief= RAISED)

L.pack(pady=0,padx=0)

else:

messagebox.showinfo("record not found")

conn.close()

b1=Button(top,text='close',width=20,height=2,bg='skyblue',command=top.destroy)

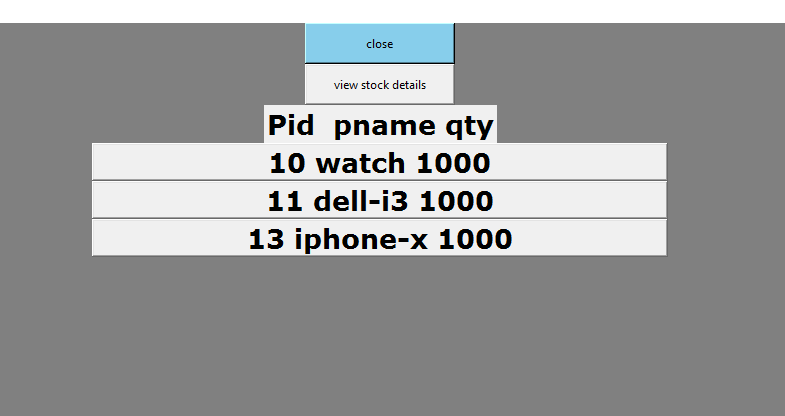
b1.pack()

b2=Button(top,text='show sale details',width=20,height=2,command=stock\_display)

b2.pack()

l=Label(top,text='Pid pname qty',font = "Verdana 20 bold")

l.pack()



**6. Creating product gui windows:** Now we make another gui window related to the products in inventory through which we can add new product in inventory and delete existing products and also view product details.

**Product insertion code through gui in database**:

def open\_1():

def insert():

pid=product\_id.get()

pn=product\_name.get()

pp=product\_price.get()

sp=sale\_price.get()

pq=product\_quantity.get()

try:

conn= pymysql.connect(host='localhost',user='root',password='',db='ims')

a=conn.cursor()

a.execute("insert into product values('"+pid+"','"+pn+"','"+pp+"','"+sp+"','"+pq+"')")

conn.commit()

messagebox.showinfo('submitted')

except:

conn.rollback()

messagebox.showinfo('Not submit')

conn.close()

top=Toplevel()

top.title('NEW Window')

Mframe=Frame(top,bg="yellow",width=1000,height=800,relief='raise',bd=10)

Mframe.pack(padx=50,pady=20)

lb1=Label(Mframe,text="Product\_id",width=15)

lb2=Label(Mframe,text="Product\_name",width=15)

lb3=Label(Mframe,text="Product\_price",width=15)

lb4=Label(Mframe,text="Sale\_price",width=15)

lb5=Label(Mframe,text="Product\_quantity",width=15)

lb1.grid(row=3,column=0,padx=10,pady=10)

lb2.grid(row=4,column=0,padx=10,pady=10)

lb3.grid(row=5,column=0,padx=10,pady=10)

lb4.grid(row=6,column=0,padx=10,pady=10)

lb5.grid(row=7,column=0,padx=10,pady=10)

product\_id=StringVar()

product\_name=StringVar()

product\_price=StringVar()

sale\_price=StringVar()

product\_quantity=StringVar()

tb1=Entry(Mframe,textvariable=product\_id)

tb2=Entry(Mframe,textvariable=product\_name)

tb3=Entry(Mframe,textvariable=product\_price)

tb4=Entry(Mframe,textvariable=sale\_price)

tb5=Entry(Mframe,textvariable=product\_quantity)

tb1.grid(row=3,column=1)

tb2.grid(row=4,column=1)

tb3.grid(row=5,column=1)

tb4.grid(row=6,column=1)

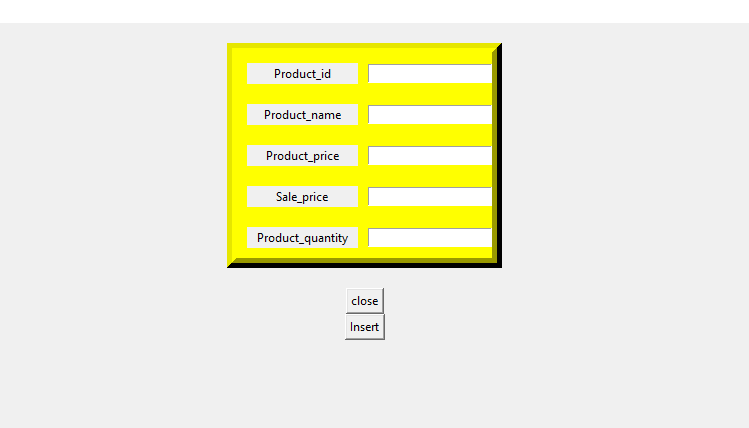
tb5.grid(row=7,column=1)

b1=Button(top,text='close',command=top.destroy)

b1.pack()

b2=Button(top,text='Insert',command=insert)

b2.pack()



b2=Button(top,text='DELETE PRODUCT DETAILS',width=20,height=2,command=delete)

b2.pack()

b1=Button(top,text='close',width=20,height=2,bg='skyblue',command=top.destroy)

b1.pack()

**Product deletion code through gui from database**:

def open3():

def delete():

pid=product\_id.get()

try:

conn = pymysql.connect(host='localhost', user='root', password='', db='ims')

a = conn.cursor()

a.execute("delete from product where productid='"+pid+"'")

conn.commit()

messagebox.showinfo('deleted')

except:

conn.rollback()

messagebox.showinfo('Not deleted')

conn.close()

top=Toplevel()

top.geometry("720x320")

top.title('NEW Window')

top.config(background='gray')

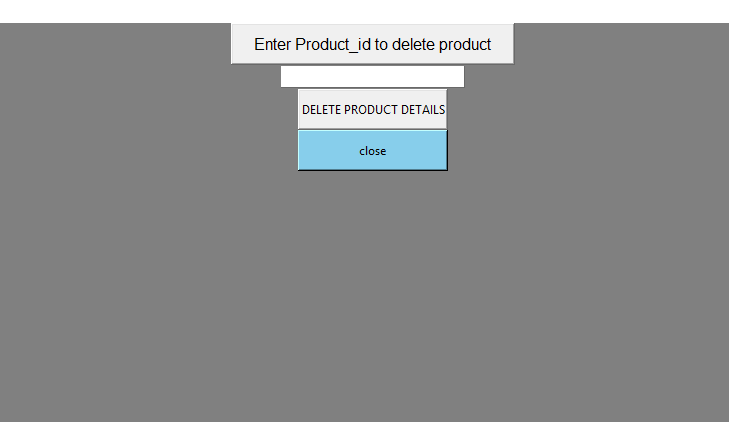
lb1=Label(top,text="Enter Product\_id to delete product",width=100)

lb1.pack()

product\_id=StringVar()

tb1=Entry(top,textvariable=product\_id)

tb1.pack()



**Code to display product details through gui:**

def open2():

top=Toplevel()

top.geometry("720x320")

top.title('NEW Window')

top.config(background='gray')

sb = Scrollbar(top)

sb.pack(side = RIGHT, fill = Y)

mylist = Listbox(top,yscrollcommand = sb.set)

def display():

conn = pymysql.connect(host='localhost', user='root', password='', db='ims')

a = conn.cursor()

a.execute("select \* from product")

results = a.fetchall()

count = a.rowcount

#print(results)

print(count)

if count > 0:

for row in results:

print()

L = Label(top,text=row,padx=5,pady=10,font=50,width=30,relief= RAISED)

L.pack(pady=0,padx=0)

else:

messagebox.showinfo("record not found")

conn.close()

b1=Button(top,text='close',width=20,height=2,bg='skyblue',command=top.destroy)

b1.pack()

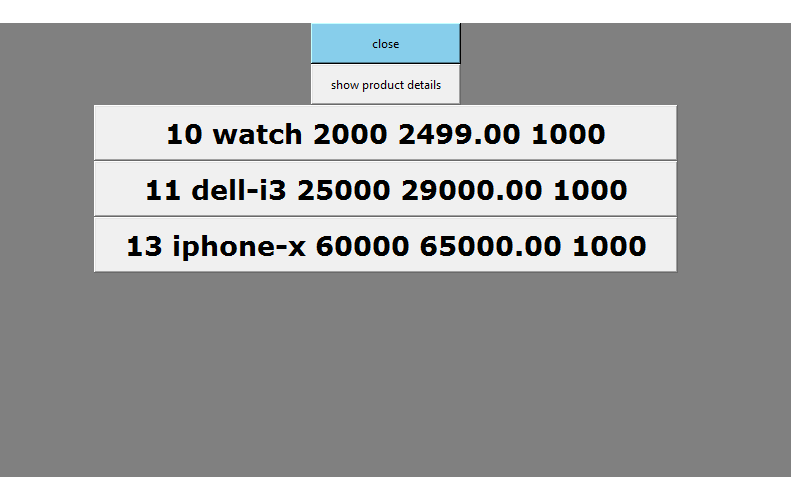
b2=Button(top,text='show product details',

width=20,height=2,command=display)

b2.pack()

l=Label(top,text='pid pn pp sp qty',font=50)

l.pack()



**7. Creating Sales gui windows**

**Code to insert sale details in database through gui:**

def open\_s1():

def insertsale():

sid=sale\_id.get()

pid=product\_id.get()

sp=sale\_price.get()

date=sale\_date.get()

sq=sale\_quantity.get()

try:

conn= pymysql.connect(host='localhost',user='root',password='',db='ims')

a=conn.cursor()

a.execute("insert into sale values('"+sid+"','"+pid+"','"+sp+"','"+date+"','"+sq+"')")

conn.commit()

messagebox.showinfo('submitted')

except:

conn.rollback()

messagebox.showinfo('Not submit')

conn.close()

top=Toplevel()

top.title('NEW Window')

Mframe=Frame(top,bg="yellow",width=1000,height=800,relief='raise',bd=10)

Mframe.pack(padx=50,pady=20)

lb1=Label(Mframe,text="Sale\_id",width=15)

lb2=Label(Mframe,text="Product\_id",width=15)

lb3=Label(Mframe,text="Sale\_price",width=15)

lb4=Label(Mframe,text="Sale\_Date",width=15)

lb5=Label(Mframe,text="Sale\_quantity",width=15)

lb1.grid(row=3,column=0,padx=10,pady=10)

lb2.grid(row=4,column=0,padx=10,pady=10)

lb3.grid(row=5,column=0,padx=10,pady=10)

lb4.grid(row=6,column=0,padx=10,pady=10)

lb5.grid(row=7,column=0,padx=10,pady=10)

sale\_id=StringVar()

product\_id=StringVar()

sale\_price=StringVar()

sale\_date=StringVar()

sale\_quantity=StringVar()

tb1=Entry(Mframe,textvariable=sale\_id)

tb2=Entry(Mframe,textvariable=product\_id)

tb3=Entry(Mframe,textvariable=sale\_price)

tb4=Entry(Mframe,textvariable=sale\_date)

tb5=Entry(Mframe,textvariable=sale\_quantity)

tb1.grid(row=3,column=1)

tb2.grid(row=4,column=1)

tb3.grid(row=5,column=1)

tb4.grid(row=6,column=1)

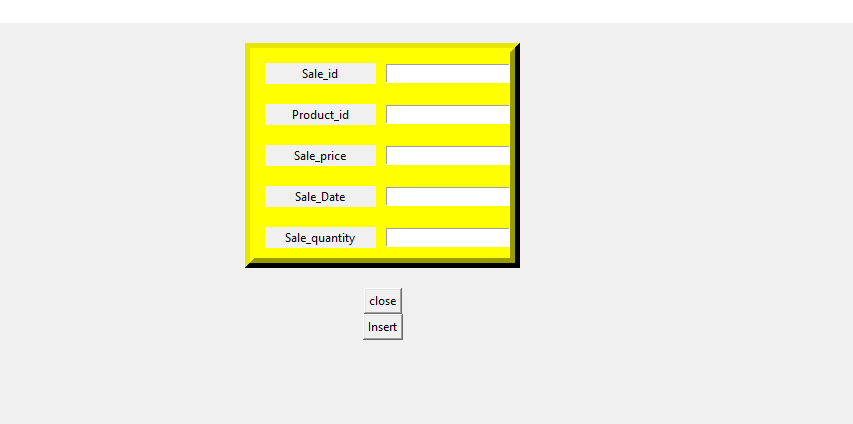
tb5.grid(row=7,column=1)

b1=Button(top,text='close',command=top.destroy)

b1.pack()

b2=Button(top,text='Insert',command=insertsale)

b2.pack()



**Code to display sale details through gui:**

def open\_s3():

top=Toplevel()

top.geometry("720x320")

top.title('NEW Window')

top.config(background='gray')

sb = Scrollbar(top)

sb.pack(side = RIGHT, fill = Y)

mylist = Listbox(top,yscrollcommand = sb.set)

def display():

conn = pymysql.connect(host='localhost', user='root', password='', db='ims')

a = conn.cursor()

a.execute("select \* from sale")

results = a.fetchall()

count = a.rowcount

print(count)

if count > 0:

for row in results:

print()

L = Label(top,text=row,padx=5,pady=10,font=50,width=30,relief= RAISED)

L.pack(pady=0,padx=0)

else:

messagebox.showinfo("record not found")

conn.close()

b1=Button(top,text='close',width=20,height=2,bg='skyblue',command=top.destroy)

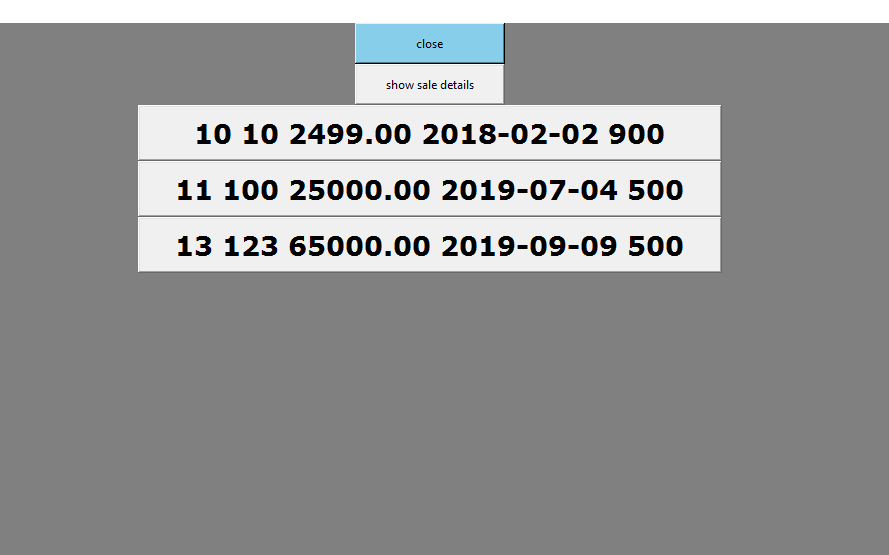
b1.pack()

b2=Button(top,text='show sale details',width=20,height=2,command=display)

b2.pack()

l=Label(top,text='pid pn pp sp qty',font=50)

l.pack()



**Code to update sale details through gui:**

def open\_s2():

def update():

sid=sale\_id.get()

pid=product\_id.get()

sp=sale\_price.get()

date=sale\_date.get()

sqty=sale\_quantity.get()

try:

conn = pymysql.connect(host='localhost', user='root', password='', db='ims')

a = conn.cursor()

a.execute("update sale set productid='"+pid+"',price='"+sp+"',date='"+date+"',saleqty='"+sqty+"' where saleid='"+sid+"'")

conn.commit()

messagebox.showinfo('updated')

except:

conn.rollback()

messagebox.showinfo('Not updated')

conn.close()

top=Toplevel()

top.title('NEW Window')

Mframe=Frame(top,bg="yellow",width=1000,height=800,relief='raise',bd=10)

Mframe.pack(padx=50,pady=20)

lb1=Label(Mframe,text="Sale\_id",width=15)

lb2=Label(Mframe,text="Product\_id",width=15)

lb3=Label(Mframe,text="Sale\_price",width=15)

lb4=Label(Mframe,text="Sale\_Date",width=15)

lb5=Label(Mframe,text="Sale\_quantity",width=15)

lb1.grid(row=3,column=0,padx=10,pady=10)

lb2.grid(row=4,column=0,padx=10,pady=10)

lb3.grid(row=5,column=0,padx=10,pady=10)

lb4.grid(row=6,column=0,padx=10,pady=10)

lb5.grid(row=7,column=0,padx=10,pady=10)

sale\_id=StringVar()

product\_id=StringVar()

sale\_price=StringVar()

sale\_date=StringVar()

sale\_quantity=StringVar()

tb1=Entry(Mframe,textvariable=sale\_id)

tb2=Entry(Mframe,textvariable=product\_id)

tb3=Entry(Mframe,textvariable=sale\_price)

tb4=Entry(Mframe,textvariable=sale\_date)

tb5=Entry(Mframe,textvariable=sale\_quantity)

tb1.grid(row=3,column=1)

tb2.grid(row=4,column=1)

tb3.grid(row=5,column=1)

tb4.grid(row=6,column=1)

tb5.grid(row=7,column=1)

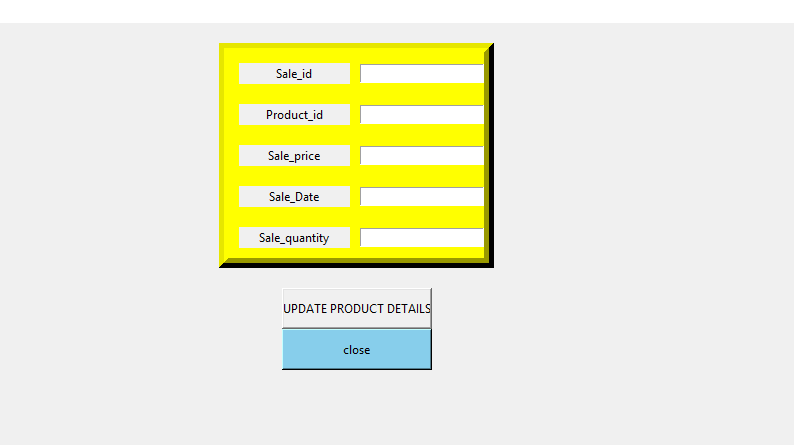
b2=Button(top,text='UPDATE PRODUCT DETAILS',

width=20,height=2,command=update)

b2.pack()

b1=Button(top,text='close',width=20,height=2,bg='skyblue',command=top.destroy)

b1.pack()



**TOOLS AND SOFTWARE USED**

**Tkinter Module:** Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

**To create a tkinter:**

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

**import tkinter as tk(use by importing)**

**Pymysql database :** Pymysql  is a pure-Python MySQL client library, based on PEP 249. Most public APIs are compatible with mysqlclient and MySQLdb. PyMySQL works with MySQL 5.5+ and MariaDB 5.5+.

MySQL is a leading open source database management system. It is a multiuser, multithreaded database management system. MySQL is especially popular on the web.

**Pymysql setup** : pip install PyMySQL

We use the pip tool to install PyMySQL.

**RESULT AND APPLICATION**

**Inventory management software** is a [software system](https://en.wikipedia.org/wiki/Software_system) for tracking [inventory](https://en.wikipedia.org/wiki/Inventory) levels, orders, [sales](https://en.wikipedia.org/wiki/Sales) and deliveries. It can also be used in the [manufacturing](https://en.wikipedia.org/wiki/Manufacturing) industry to create a [work order](https://en.wikipedia.org/wiki/Work_order), [bill of materials](https://en.wikipedia.org/wiki/Bill_of_materials) and other production-related documents. Companies use inventory management software to avoid product [overstock](https://en.wikipedia.org/wiki/Overstock) and outages. It is a tool for organizing inventory [data](https://en.wikipedia.org/wiki/Data) that before was generally stored in hard-copy form or in [spreadsheets](https://en.wikipedia.org/wiki/Spreadsheet).

 For more complex manufacturing jobs, manufacturers can create multilevel work orders and bills of materials, which have a timeline of processes that need to happen in the proper order to build a final product. Other work orders that can be created using inventory management software include reverse work orders and auto work orders. Manufacturers also use inventory management software for tracking assets, receiving new inventory and additional tasks businesses in other industries use it for.

**FUTURE ASPECT**

There is no doubt that artificial intelligence will continue to be one of the biggest game changers in the future of inventory management. For example, artificial intelligence can help a company incorporate chatbots on their eCommerce site. These bots can answer customer queries about what’s in stock, the price of items, shipping policies and so much more. This eliminates the need for someone to be staffing a chat channel. It can free up space for an employee to do work elsewhere.

Artificial intelligence can also guide business decisions through the results of running detailed analytics. These can include how to maximise warehouse space, production cost reduction and demand forecasting support.

Currently, machine learning is the most common artificial intelligence tool, which yields very reliable data analysis. It is a great tool to reduce human error and it is constantly evolving to become better.

**REFERENCES**

[1]. <https://docs.python.org/3/>

[2]. <https://docs.python.org/3/library/tkinter.html>

[3]. <https://pymysql.readthedocs.io/en/latest/>

[4]. <https://dev.mysql.com/doc/>

[5]. <https://tkdocs.com/>