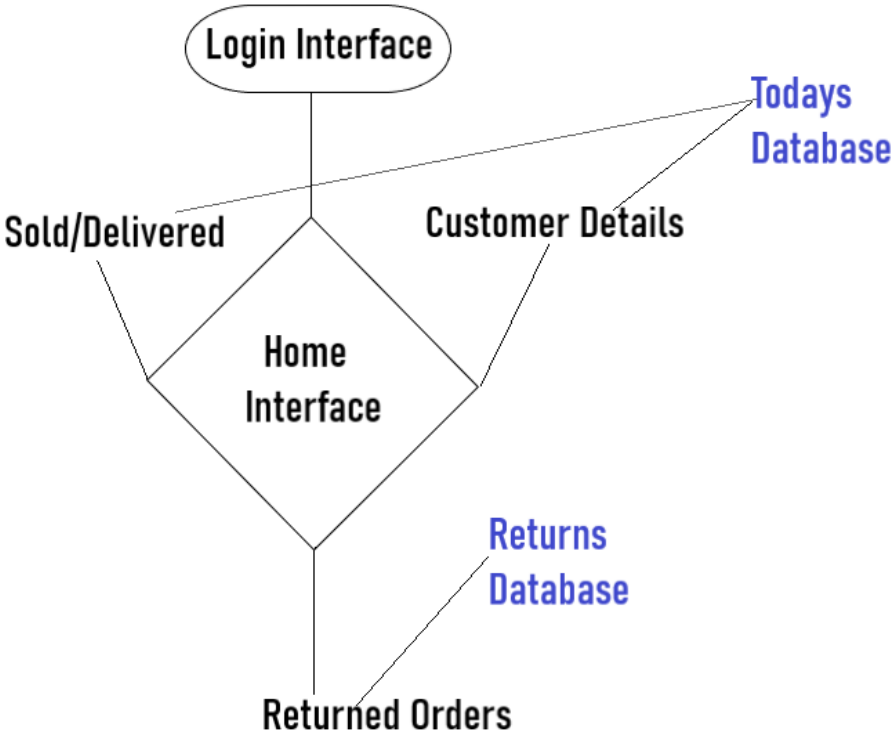
Index:

Introduction to the Project:

The Order Management System is a modern form of a ledger, which is ideally used to keep a record of the goods sold from a sop or a modern-day ecommerce website. It is used so widely across the globe because it can be backed up through cloud and the data goes nowhere else. Many of the local seller also have opted the same system especially the Medical Store owners found the system very useful so as they can keep a complete record of the client and this type of system is suitable to all the businesses.

After getting the approval of our teacher it took around 4 months to completely ear and develop with gradual taste of all the type of errors we can see while coding.

Structure of the System:



Note: All the code is written, tested and debugged by Rajsekhar Panda and Divyanshu Class XII, SSRVM Sen.Sec.School, Una, Himachal Pradesh, Academic Year 2022-23

System Requirements:

Operating System:

- **Windows 7 or any higher version**
- **Any Linux Distribution**

Software Requirement:

- **Python 3.11 (32 or 64 Bit)**
- **SQL (Complete Package)**

Code Editor Requirement:

- **Sublime Text form https://www.sublimetext.com/**
- **Microsoft Vs Code from https://code.visualstudio.com/**

Packages Installed:

- **Pillow (command for Windows : pip install pillow for Linux: pip3 install pillow)**
- **My SQL Connector**

Modules Used:

- **Tkinter**
- **Tkinter – ttk**
- **Tkinter – Messagebox**
- **Pillow – Pil**
- **Pil- Image and Image Tk**
- **My SQL Connector**

Background Image Source: https://undraw.co/search/undraw_empty_cart_co35

The Python Code:

The Login Interface:

```
from tkinter import*
from tkinter import ttk
from PIL import Image, ImageTk
from tkinter import messagebox
import mysql.connector
from Order_Management import Order_Management
from PIL import Image, ImageTk

def main():
    win = Tk()
    app = Login_Window(win)
    win.mainloop()

#making the widget of the loginn Window
class Login_Window:#widget for login window
    print("LW Stage 1 Working")

    def __init__(self,root):
        self.root=root
        self.root.title("Login")#displays title
        self.root.geometry("2400x1850+0+0")#specifies the width and height of the window

        #Adding the background in the window
        self.bg=ImageTk.PhotoImage(file= "jacques-dillies-jcav1COVvOc-unsplash.jpg")
        lbl_bg= Label(self.root,image=self.bg)
```

```
    lbl_bg.place(x=0,y=0,relwidth=1,relheight=1)#place is used to shove the conten ton screen of window

    #maing the login window frame:

    frame = Frame(self.root,bg="white")

  frame.place(x=610,y=170,width=340,height=500)

    # providing the user image icon

    img1 = Image.open(r"user.png")# r is to read

    img1 = img1.resize((100,100),Image.ANTIALIAS)#Anti-aliasing is the smoothing of jagged edges in digital images by averaging the colors of the pixels at a boundary

    self.photoimage =ImageTk.PhotoImage(img1)

    lblimg1 = Label(image=self.photoimage,bg="White",borderwidth=0)

    lblimg1.place(x =730,y=180,width=100,height=100)

    Usr_lin= Label(frame,text="USER LOGIN",font=("century gothic",30),fg="black",bg="white")

    Usr_lin.place(x= 50,y=120)

    # Lable for the user name  and password

    username =  lbl= Label(frame,text="Seller-Id:",font=("century gothic",17,"bold"),fg="black",bg="white")

    username.place(x=40,y=195 )

    self.txtuser=ttk.Entry(frame,font=("consolas",17,))

    self.txtuser.place(x=40,y = 225,width=270 )

    password =  lbl= Label(frame,text="Password:",font=("century gothic",17,"bold"),fg="black",bg="white")

    password.place(x=40,y=265 )

     self.txtpass=ttk.Entry(frame,font=("consolas",17,))

    self.txtpass.place(x=40,y = 300,width=270 )


    #---------LOGIN BUTTON----------#

    loginbtn = Button(frame,command=self.login,text="LOGIN",font=("century gothic",16,"bold"),bd=3,relief= RIDGE,fg="white",bg="#F70D1A")

    loginbtn.place(x=110,y=360,width=120,height=50 )

    print("LW Stage 2 Working")


#----------Function for Login Button----------#
```

```
    def login(self):
        if self.txtuser.get()=="" or self.txtpass.get()=="":
            messagebox.showerror("Error","All fields are required")
        else:
            if self.txtuser.get()== "order_management" and self.txtpass.get()=="0987654321" :
                self.new_window = Toplevel(self.root)
                self.app = Order_Management(self.new_window)
            else:
                messagebox.showerror("Error ","Use Correct Seller-Id and Password")


    """ the user id : order_management
        the password : 0987654321
    """
if __name__=="__main__":
    main()
```

The Home Interface:

```
from tkinter import*
from PIL import Image, ImageTk
from Todays import Todays_Ord
from Returns import Returns
from Customer_Info import Customer_Info
def main():
    win = Tk()
    app = Order_Management(win)
    win.mainloop()

class Order_Management:#widget for the window
    print("OM Stage 1 Working")
```

```
    def __init__(self,root):
        self.root=root
        self.root.title("Order Management System")# giving title
        self.root.geometry("2400x1850+0+0")#giving the window height and width


        #----Heading of the System-------#
        label = Label(root, text="Order Management Sytem",font=("century gothic",43 ), fg  = "gold", bg = "black", width = 45)
        label.place(x = 0,y= 0 )
        self.bg=ImageTk.PhotoImage(file= "undraw_empty_cart_co35_n.png")#backgroundimage
        lbl_bg= Label(self.root,image=self.bg)
        lbl_bg.place(x=190,y=80,relwidth=1,relheight=1,height = -10)#
        frame = Frame(self.root,bg="White")
        frame.place(x=0,y=80,width=350,height=800)


        #-----------Making the label frame----------------#
        label_frame_left = LabelFrame(self.root,bd = 10, relief  =  RIDGE, font = ("century gothic",18,"bold"))
        label_frame_left.place(x = 0,y = 79,width = 425 ,height = 750)
        lbl0 = Label(label_frame_left, text = "नमस्ते!",font=("Arial",35))
        lbl0.place(x = 110, y = 35)
        lbl1 = Label(label_frame_left, text = "Welcome to Order Management System",font=("century gothic",15),)
        lbl1.place(x = 1, y = 105)


        #Todays Sales
        label1 = Button(label_frame_left, text = " Sold/ Delivered",command = self.Todays_order,font=("century gothic",22),bd=3,relief= RIDGE,fg="white",bg="#F70D1A")
        label1.place(x = 40, y = 150)


        #returned Orders
        label4 = Button(label_frame_left, text = "Returns",command = self.Returns,font=("century gothic",22),bd=3,relief= RIDGE,fg="white",bg="#F70D1A")
```

```
        label4.place(x = 40, y = 300)


        #Customer Detalils :
        label69 = Button(label_frame_left, text = "Customer Details", command =
self.Customer_Details,font=("century gothic",22),bd=3,relief= RIDGE,fg="white",bg="#F70D1A")
        label69.place(x = 40, y = 450)
    print("OM Stage 2 Working")


    #----------Defining Functions for the Buttons --------#
    def Todays_order(self):
        self.new_window  = Toplevel(self.root)
        self.app = Todays_Ord(self.new_window)


    def Returns(self):
        self.new_window = Toplevel(self.root)
        self.app = Returns(self.new_window)


    def Customer_Details(self):
         self.new_window = Toplevel(self.root)
         self.app = Customer_Info(self.new_window)
print("OM Stage 3 Working")


if __name__=="__main__":
    main()
```

The Sold/Delivered Interface:

```
from tkinter import*
from tkinter import ttk
from PIL import Image, ImageTk
from tkinter import messagebox
```

```
import mysql.connector
import random
def main():
   win = Tk()
   app = Todays_Ord(win)
   win.mainloop()
class Todays_Ord:
   def __init__(self,root):
      self.root=root
      self.root.title("Today's Orders")
      self.root.geometry("1920x1080+0+0")
      print("TO Stage 1 Working")

      #----------Declaration of Variable ---------- #
      self.var_ref = StringVar()
      self.var_name    =  StringVar()
      self.var_num=  StringVar()
      self.var_mail=  StringVar()
      self.var_pin=  StringVar()
      self.var_addr=  StringVar()
      self.var_product=  StringVar()
      self.var_price  =  StringVar()

      #===========Page Hheading==============#
      label = Label(root, text="Sold/ Delivered",font=("century gothic",43 ),fg  = "black", bg = "white",
width = 45)
      label.place(x = 0,y= 0 )
      ##Costomer info ##
      labelframeleft = LabelFrame(self.root,bd = 5, relief  =  RIDGE, text = " Customer Details : ",font =
("century gothic",18,"bold"))
      labelframeleft.place(x = 2,y = 76,width = 420 ,height = 800)
      # Label and entry
```

```
#reference Id:
lbl_Cust_Ref = Label(labelframeleft, text = "Date(yyyy.mm.dd)",font =("century gothic",18),padx = 2,pady = 6)
lbl_Cust_Ref.grid(row = 0, column = 0, sticky = W)
entry_ref = ttk.Entry(labelframeleft,textvariable = self.var_ref,width = 29,font = ("Consolas",18))
entry_ref.grid(row = 1, column = 0)

# name
cname = Label(labelframeleft, text = "Full Name:",font =("century gothic",18),padx = 2,pady = 6)
cname.grid(row = 2, column = 0, sticky = W)
txtcname = ttk.Entry(labelframeleft,textvariable = self.var_name,width = 29,font = ("Consolas",18))
txtcname.grid(row = 3, column = 0)

# number
cnum = Label(labelframeleft, text = "Mobile Number:",font =("century gothic",18),padx = 2,pady = 6)
cnum.grid(row = 4, column = 0, sticky = W)
txtcnum = ttk.Entry(labelframeleft,textvariable = self.var_num,width = 29,font = ("Consolas",18))
txtcnum.grid(row = 5, column = 0)

#mail
cmail = Label(labelframeleft, text = " Mail Id:",font =("century gothic",18),padx = 2,pady = 6)
cmail.grid(row = 6, column = 0, sticky = W)
txtcmail = ttk.Entry(labelframeleft,textvariable = self.var_mail,width = 29,font = ("Consolas",18))
txtcmail.grid(row = 7, column = 0)

#Pincode
cpin = Label(labelframeleft, text = "Pincode :",font =("century gothic",18),padx = 2,pady = 6)
cpin.grid(row = 8, column = 0, sticky = W)
txtcpin = ttk.Entry(labelframeleft,textvariable = self.var_pin,width = 29,font = ("Consolas",18))
```

```
        txtcpin.grid(row = 9, column = 0)


        #Full Add
        caddr = Label(labelframeleft, text = "Address:",font =("century gothic",18),padx = 2,pady = 6)
        caddr.grid(row = 10, column = 0, sticky = W)
        txtcaddr = ttk.Entry(labelframeleft,textvariable = self.var_addr,width = 29,font =
("Consolas",18))
        txtcaddr.grid(row = 11, column = 0)


        #Order
        cproduct = Label(labelframeleft, text = "Product :",font =("century gothic",18),padx = 2,pady = 6)
        cproduct.grid(row = 12, column = 0, sticky = W)
        txtcproduct = ttk.Entry(labelframeleft,textvariable  = self.var_product,width = 29,font =
("Consolas",18))
        txtcproduct.grid(row = 13, column =0)


        #Price
        cprice = Label(labelframeleft, text = "Price:",font =("century gothic",18),padx = 2,pady = 6)
        cprice.grid(row = 14, column = 0, sticky = W)
        txtcprice = ttk.Entry(labelframeleft,textvariable = self.var_price,width = 29,font =
("Consolas",18))
        txtcprice.grid(row = 15, column = 0)
        print("TO Stage 2 Working")


        # Add BUTTON
        btnADD = Button(labelframeleft,command = self.add_data,  text = "Add",font =
("Arial",11,"bold"),bg = "black",fg = "gold", width = 10)
        btnADD.place(x = 50, y = 650)
        btnADD = Button(labelframeleft,command = self.reset,  text = "Reset",font =
("Arial",11,"bold"),bg = "black",fg = "gold", width = 10)
        btnADD.place(x = 180, y = 650)


        # Making the tableframe
```

```
    Table_frm = LabelFrame(self.root, bd = 10, relief=  RIDGE, text=  "View Details :",font = ("century gothic",16,"bold"),padx = 2)

    Table_frm.place(x = 455,y =90,height = 700, width  = 1070)


  #----------------Show Table----------#

    details_table = Frame(Table_frm,bd = 3,relief = RIDGE)

    details_table.place(x = 0, y = 10, height = 650, width = 1042)

    scroll_x  = ttk.Scrollbar(details_table, orient = HORIZONTAL)

    scroll_y  = ttk.Scrollbar(details_table, orient = VERTICAL)

    self.Cust_Details_Table = ttk.Treeview(details_table,column = ( "ref","name","num","mail","pin","addr","product","price"),xscrollcommand = scroll_x.set,yscrollcommand = scroll_y.set)

    scroll_x.pack(side = BOTTOM,fill = X)

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

    scroll_x.config( command = self.Cust_Details_Table.xview)

    scroll_y.config( command = self.Cust_Details_Table.yview)

    self.Cust_Details_Table.heading("ref", text = "Date")

    self.Cust_Details_Table.heading("name", text = "Name")

    self.Cust_Details_Table.heading("num", text ="Mobile Number")

    self.Cust_Details_Table.heading("mail", text ="Mail Id")

    self.Cust_Details_Table.heading("pin", text ="Pincode")

    self.Cust_Details_Table.heading("addr", text ="Address")

    self.Cust_Details_Table.heading("product", text ="Product")

    self.Cust_Details_Table.heading("price", text ="Price")

    self.Cust_Details_Table["show"] = "headings"

    self.Cust_Details_Table.column("ref",      width = 200        )

    self.Cust_Details_Table.column("name",     width = 200        )

    self.Cust_Details_Table.column("num",      width = 200        )

    self.Cust_Details_Table.column("mail",     width = 200        )

    self.Cust_Details_Table.column("pin",      width = 200        )

    self.Cust_Details_Table.column("addr",     width = 200        )

    self.Cust_Details_Table.column("product",  width = 200        )
```

```
        self.Cust_Details_Table.column("price",      width = 200          )
        self.Cust_Details_Table.pack(fill = BOTH , expand = 1 )
        self.fetch_data()
    print("TO Stage 3 Working")
#------------------------Function to Add data-----------------------------------------------------------#
    def add_data(self):
        if self.var_num.get()=="":
            messagebox.showerror("Error", "Mobile Number Is Required")
        else:
            try:
                conn =
mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
                my_cursor = conn.cursor()
                my_cursor.execute("insert into todays values(%s,%s,%s,%s,%s,%s,%s,%s)",(
                                        self.var_ref.get(),
                                        self.var_name.get(),
                                        self.var_num.get(),
                                        self.var_mail.get(),
                                        self.var_pin.get(),
                                        self.var_addr.get(),
                                        self.var_product.get(),
                                        self.var_price.get()
                                                ))
                conn.commit()
                self.fetch_data()
                conn.close()
                messagebox.showinfo("Success", "Data has been Successfully inserted",parent = self.root)
            except Exception as es:
                messagebox.showwarning("Warning ", f"Something went wrong : {str(es)}")
```

```
    def fetch_data(self):
        conn = mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
        my_cursor = conn.cursor()
        my_cursor.execute("select *  from  todays ")
        rows  = my_cursor.fetchall()
        if len(rows)!= 0:
            self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
            for i in rows:
                self.Cust_Details_Table.insert("", END, values = i)
            conn.commit()
        conn.close()
    def reset(self):
        self.var_ref = StringVar()
        self.var_name.set(" ")
        self.var_num.set(" ")
        self.var_mail.set(" ")
        self.var_addr.set(" ")
        self.var_product.set(" ")
        self.var_price.set(" ")
    print("TO Stage 4 Working")
if __name__=="__main__":
    main()
```

The Returns Interface:

```
from tkinter import*
from tkinter import ttk
from PIL import Image, ImageTk
from tkinter import messagebox
import mysql.connector
def main():
    win = Tk()
    app = Returns(win)
    win.mainloop()

class Returns:
    print(" R stage 1 Working")
    def __init__(self,root):
        self.root=root
        self.root.title("Returns")# giving title
        self.root.geometry("1920x1080+0+0")

        # Declaring the variables

        self.var_ref = StringVar()
        self.var_name    =  StringVar()
        self.var_num=  StringVar()
        self.var_mail=  StringVar()
        self.var_pin=  StringVar()
        self.var_addr=  StringVar()
        self.var_product=  StringVar()
        self.var_price  =  StringVar()
        self.var_rsn=  StringVar()
```

```
label = Label(root, text="Returned Orders",font=("century gothic",40 ), fg  = "Black")

label.place(x = 550,y= 0 )


##Costomer info ##

labelframeleft = LabelFrame(self.root,bd = 5, relief  =  RIDGE, text = " Customer Details : ",font = ("century gothic",18,"bold"))

labelframeleft.place(x = 2,y = 5,width = 420 ,height = 800)


# Label and entry

#reference Id:

lbl_Cust_Ref = Label(labelframeleft, text = "Date(yyyy.mm.dd)",font =("century gothic",18),padx = 2,pady = 6)

lbl_Cust_Ref.grid(row = 0, column = 0, sticky = W)

entry_ref = ttk.Entry(labelframeleft,textvariable = self.var_ref,width = 29,font = ("Consolas",18))

entry_ref.grid(row = 1, column = 0)

# name

cname = Label(labelframeleft, text = "Full Name:",font =("century gothic",18),padx = 2,pady = 6)

cname.grid(row = 2, column = 0, sticky = W)

txtcname = ttk.Entry(labelframeleft,textvariable = self.var_name,width = 29,font = ("Consolas",18))

txtcname.grid(row = 3, column = 0)


# number

cnum = Label(labelframeleft, text = "Mobile Number:",font =("century gothic",18),padx = 2,pady = 6)

cnum.grid(row = 4, column = 0, sticky = W)

txtcnum = ttk.Entry(labelframeleft,textvariable = self.var_num,width = 29,font = ("Consolas",18))

txtcnum.grid(row = 5, column = 0)


#mail

cmail = Label(labelframeleft, text = " Mail Id:",font =("century gothic",18),padx = 2,pady = 6)

cmail.grid(row = 6, column = 0, sticky = W)
```

```
txtcmail = ttk.Entry(labelframeleft,textvariable = self.var_mail,width = 29,font = ("Consolas",18))
txtcmail.grid(row = 7, column = 0)

#Pincode
cpin = Label(labelframeleft, text = "Pincode :",font =("century gothic",18),padx = 2,pady = 6)
cpin.grid(row = 8, column = 0, sticky = W)
txtcpin = ttk.Entry(labelframeleft,textvariable = self.var_pin,width = 29,font = ("Consolas",18))
txtcpin.grid(row = 9, column = 0)

#Full Add
caddr = Label(labelframeleft, text = "Address:",font =("century gothic",18),padx = 2,pady = 6)
caddr.grid(row = 10, column = 0, sticky = W)
txtcaddr = ttk.Entry(labelframeleft,textvariable = self.var_addr,width = 29,font =
("Consolas",18))
txtcaddr.grid(row = 11, column = 0)

#Order
cproduct = Label(labelframeleft, text = "Product :",font =("century gothic",18),padx = 2,pady = 6)
cproduct.grid(row = 12, column = 0, sticky = W)
txtcproduct = ttk.Entry(labelframeleft,textvariable  = self.var_product,width = 29,font =
("Consolas",18))
txtcproduct.grid(row = 13, column =0)

#Price
cprice = Label(labelframeleft, text = "Price:",font =("century gothic",18),padx = 2,pady = 6)
cprice.grid(row = 14, column = 0, sticky = W)
txtcprice = ttk.Entry(labelframeleft,textvariable = self.var_price,width = 29,font =
("Consolas",18))
txtcprice.grid(row = 15, column = 0)

#reason
```

```
crsn  = Label(labelframeleft, text = "Reason:",font =("century gothic",18),padx = 2,pady = 6)

crsn.grid(row = 16, column = 0, sticky = W)

txtcprice = ttk.Entry(labelframeleft,textvariable = self.var_rsn,width = 29,font = ("Consolas",18))

txtcprice.grid(row = 17, column = 0)

print("R stage 2 Working")


#-------Buttons----------#

btnADD = Button(labelframeleft,command = self.add_data,  text = "Add",font =
("Arial",11,"bold"),bg = "black",fg = "gold", width = 10)

btnADD.place(x = 50, y = 725)

btnADD = Button(labelframeleft,command = self.reset,  text = "Reset",font =
("Arial",11,"bold"),bg = "black",fg = "gold", width = 10)

btnADD.place(x = 180, y = 725)


# Making the tableframe

Table_frm = LabelFrame(self.root, bd = 2, relief=  RIDGE, text=  "View Details and Search
System:",font = ("century gothic",16,"bold"),padx = 2)

Table_frm.place(x = 455,y =90,height = 700, width  = 1050)


#=======Show table=======#

details_table = Frame(Table_frm,bd = 2,relief = RIDGE)

details_table.place(x = 0, y = 60, height = 570, width = 1040)

scroll_x  = ttk.Scrollbar(details_table, orient = HORIZONTAL)

scroll_y  = ttk.Scrollbar(details_table, orient = VERTICAL)

self.Cust_Details_Table = ttk.Treeview(details_table,column = (
"ref","name","num","mail","pin","addr","product","price","rsn"),xscrollcommand =
scroll_x.set,yscrollcommand = scroll_y.set)

scroll_x.pack(side = BOTTOM,fill = X)

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


scroll_x.config( command = self.Cust_Details_Table.xview)

scroll_y.config( command = self.Cust_Details_Table.yview)

self.Cust_Details_Table.heading("ref", text = "Order Id")
```

```
        self.Cust_Details_Table.heading("name", text = "Name")
        self.Cust_Details_Table.heading("num", text ="Mobile Number")
        self.Cust_Details_Table.heading("mail", text ="Mail Id")
        self.Cust_Details_Table.heading("pin", text ="Pincode")
        self.Cust_Details_Table.heading("addr", text ="Address")
        self.Cust_Details_Table.heading("product", text ="Product")
        self.Cust_Details_Table.heading("price", text ="Price")
        self.Cust_Details_Table.heading("rsn", text = "Reason")
        self.Cust_Details_Table["show"] = "headings"
        self.Cust_Details_Table.column("ref",       width = 200          )
        self.Cust_Details_Table.column("name",      width = 200          )
        self.Cust_Details_Table.column("num",       width = 200          )
        self.Cust_Details_Table.column("mail",      width = 200          )
        self.Cust_Details_Table.column("pin",       width = 200          )
        self.Cust_Details_Table.column("addr",      width = 200          )
        self.Cust_Details_Table.column("product",   width = 200          )
        self.Cust_Details_Table.column("price",     width = 200          )
        self.Cust_Details_Table.column("rsn",       width = 200          )
        self.Cust_Details_Table.pack(fill = BOTH , expand = 1 )
        self.fetch_data()
        print("R stage 3 Working")
#----------------Buttons-------------------------#
    def add_data(self):
        if self.var_num.get()=="":
            messagebox.showerror("Error", "Mobile Number Is Required")
        else:
            try:
                conn =
mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
                my_cursor = conn.cursor()
                my_cursor.execute("insert into returns values(%s,%s,%s,%s,%s,%s,%s,%s,%s)",(
```

```
                                self.var_ref.get(),
                                self.var_name.get(),
                                self.var_num.get(),
                                self.var_mail.get(),
                                self.var_pin.get(),
                                self.var_addr.get(),
                                self.var_product.get(),
                                self.var_price.get(),
                                self.var_rsn.get()
                                        ))
            conn.commit()
            self.fetch_data()
            conn.close()
            messagebox.showinfo("Success", "Data has been Successfully inserted",parent = self.root)
        except Exception as es:
            messagebox.showwarning("Warning ", f"Something went wrong : {str(es)}")


    def fetch_data(self):
        conn = mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
        my_cursor = conn.cursor()
        my_cursor.execute("select *  from  returns ")
        rows  = my_cursor.fetchall()
        if len(rows)!= 0:
            self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
            for i in rows:
                self.Cust_Details_Table.insert("", END, values = i)
            conn.commit()
        conn.close()
    def reset(self):
        self.var_ref = StringVar()
```

```
        self.var_name.set(" ")
        self.var_num.set(" ")
        self.var_mail.set(" ")
        self.var_addr.set(" ")
        self.var_product.set(" ")
        self.var_price.set(" ")
print ("R stage 4 Working")


if __name__=="__main__":
    main()
```

The Customer Details Interface:

```
from tkinter import*
from tkinter import ttk
from PIL import Image, ImageTk
from tkinter import messagebox
import mysql.connector
import random

def main():
    win = Tk()
    app = Customer_Info(win)
    win.mainloop()

class Customer_Info:
    def __init__(self,root):
        self.root=root
        self.root.title("Customer Details")
        self.root.geometry("1920x1080+0+0")
```

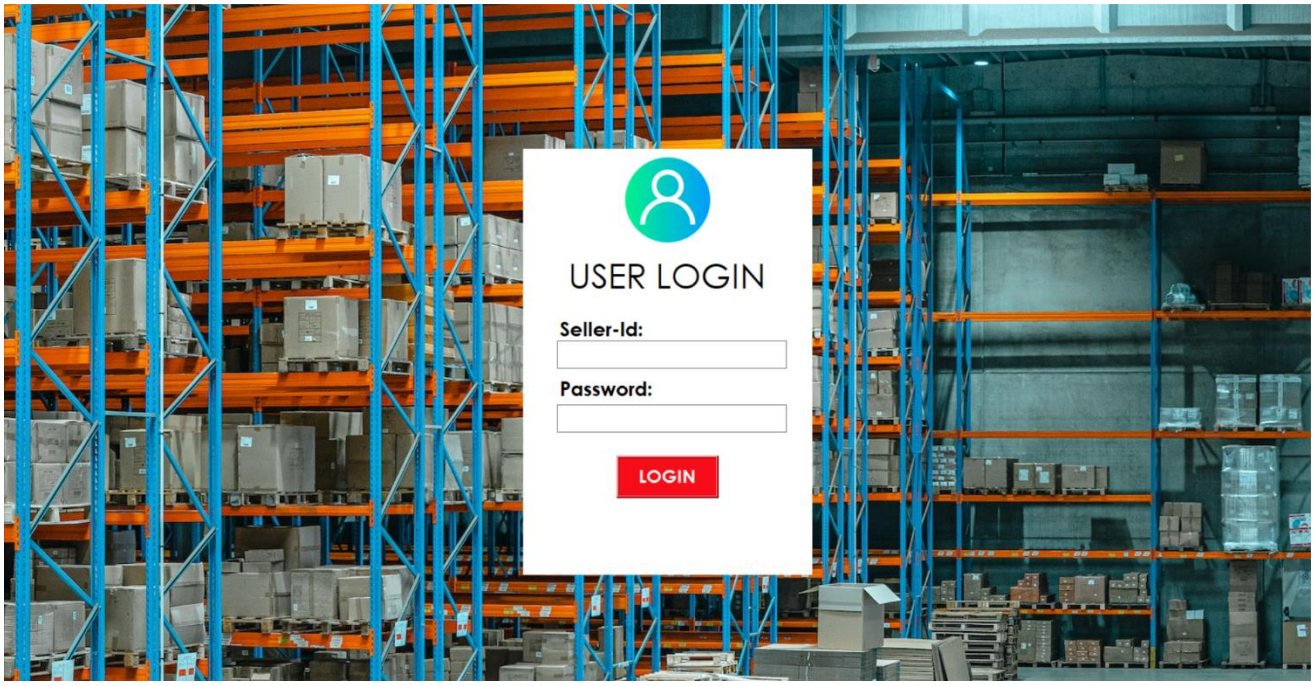
```
print("CI Stage 1 Working")

label = Label(root, text="Customer Details",font=("century gothic",43 ),fg  = "black", bg = "white", width = 45)

label.place(x = 0,y= 0 )


# Making the tableframe

Table_frm = LabelFrame(self.root, bd = 10, relief=  RIDGE, text=  " View Details : ",font = ("century gothic",16,"bold"),padx = 2)

Table_frm.place(x = 2,y =90,height = 700, width  = 1530)

print(" CIStage 2 Working")


#---------show table----------------#

details_table = Frame(Table_frm,bd = 3,relief = RIDGE)

details_table.place(x = 0, y = 50, height = 610, width = 1510)

scroll_x  = ttk.Scrollbar(details_table, orient = HORIZONTAL)

scroll_y  = ttk.Scrollbar(details_table, orient = VERTICAL)

self.Cust_Details_Table = ttk.Treeview(details_table,column = ( "ref","name","num","mail","pin","addr","product","price"),xscrollcommand = scroll_x.set,yscrollcommand = scroll_y.set)

scroll_x.pack(side = BOTTOM,fill = X)

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

scroll_x.config( command = self.Cust_Details_Table.xview)

scroll_y.config( command = self.Cust_Details_Table.yview)


self.Cust_Details_Table.heading("ref", text = "Order Id")

self.Cust_Details_Table.heading("name", text = "Name")

self.Cust_Details_Table.heading("num", text ="Mobile Number")

self.Cust_Details_Table.heading("mail", text ="Mail Id")

self.Cust_Details_Table.heading("pin", text ="Pincode")

self.Cust_Details_Table.heading("addr", text ="Address")

self.Cust_Details_Table.heading("product", text ="Product")

self.Cust_Details_Table.heading("price", text ="Price")
```

```
        self.Cust_Details_Table["show"] = "headings"
        self.Cust_Details_Table.column("ref",       width = 200          )
        self.Cust_Details_Table.column("name",      width = 200          )
        self.Cust_Details_Table.column("num",       width = 200          )
        self.Cust_Details_Table.column("mail",      width = 200          )
        self.Cust_Details_Table.column("pin",       width = 200          )
        self.Cust_Details_Table.column("addr",      width = 200          )
        self.Cust_Details_Table.column("product",   width = 200          )
        self.Cust_Details_Table.column("price",     width = 200          )
        self.Cust_Details_Table.pack(fill = BOTH , expand = 1 )
        self.fetch_data()
    print("CI Stage 3 Working")
#----------------------------Defining Functions-------------------------------------------#
    def add_data(self):
        if self.var_num.get()=="":
            messagebox.showerror("Error", "Mobile Number Is Required")
        else:
            try:
                conn =
mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
                my_cursor = conn.cursor()
                my_cursor.execute("insert into todays values(%s,%s,%s,%s,%s,%s,%s,%s)",(
                                        self.var_ref.get(),
                                        self.var_name.get(),
                                        self.var_num.get(),
                                        self.var_mail.get(),
                                        self.var_pin.get(),
                                        self.var_addr.get(),
                                        self.var_product.get(),
                                        self.var_price.get()
                                                ))
```

```
            conn.commit()
            self.fetch_data()
            conn.close()
            messagebox.showinfo("Success", "Data has been Successfully inserted",)
        except Exception as es:
            messagebox.showwarning("Warning ", f"Something went wrong : {str(es)}")



    def fetch_data(self):
        conn = mysql.connector.connect(host="localhost",user="root",password="rspanda",database
="management_01")
        my_cursor = conn.cursor()
        my_cursor.execute("select *  from  todays ")
        rows  = my_cursor.fetchall()
        if len(rows)!= 0:
            self.Cust_Details_Table.delete(*self.Cust_Details_Table.get_children())
            for i in rows:
                self.Cust_Details_Table.insert("", END, values = i)
            conn.commit()
        conn.close()


    print(" CI Stage 4 Working")


if __name__=="__main__":
    main()
```

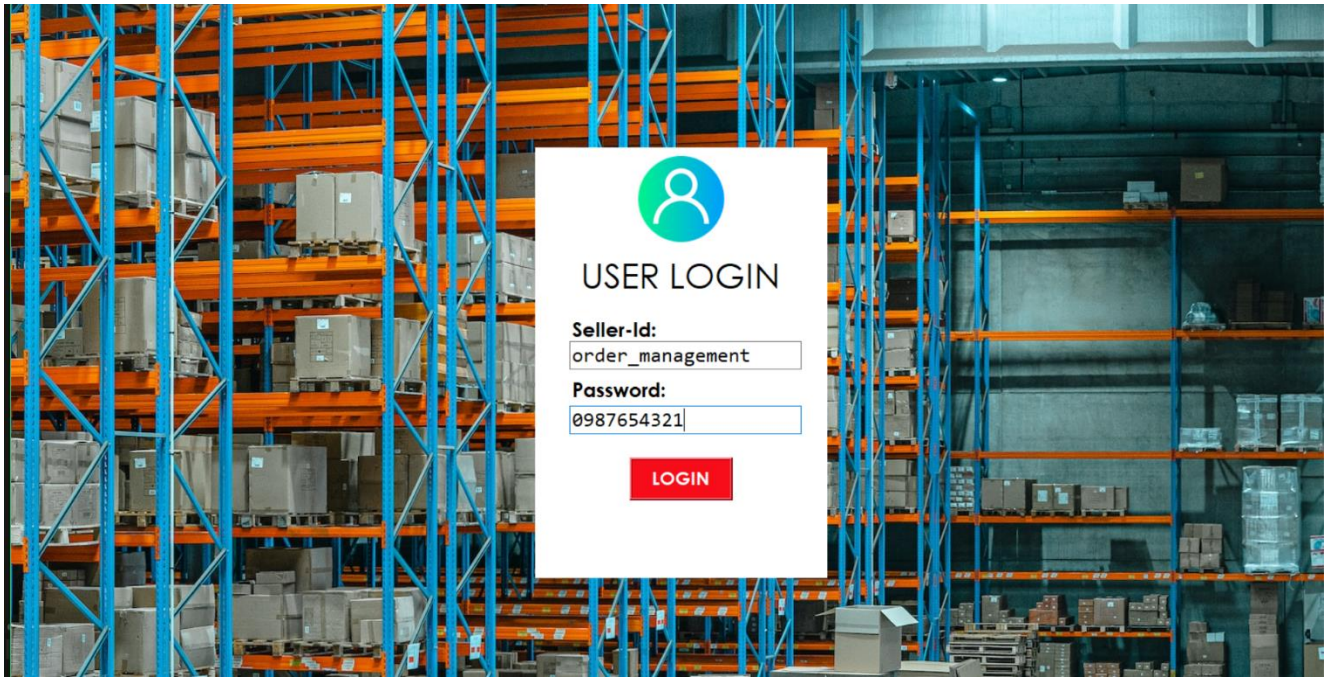
OUTPUT FOR THE PROJECT:

The Login Interface:

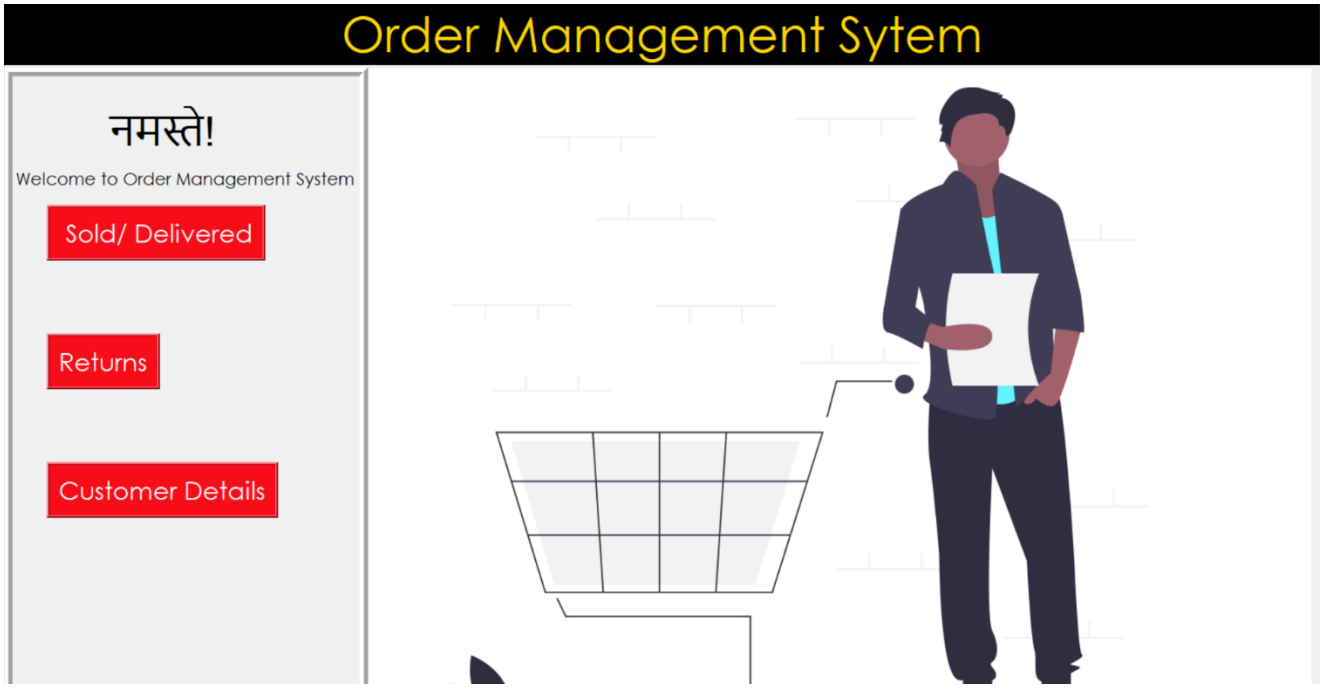


Seller Id: order_management

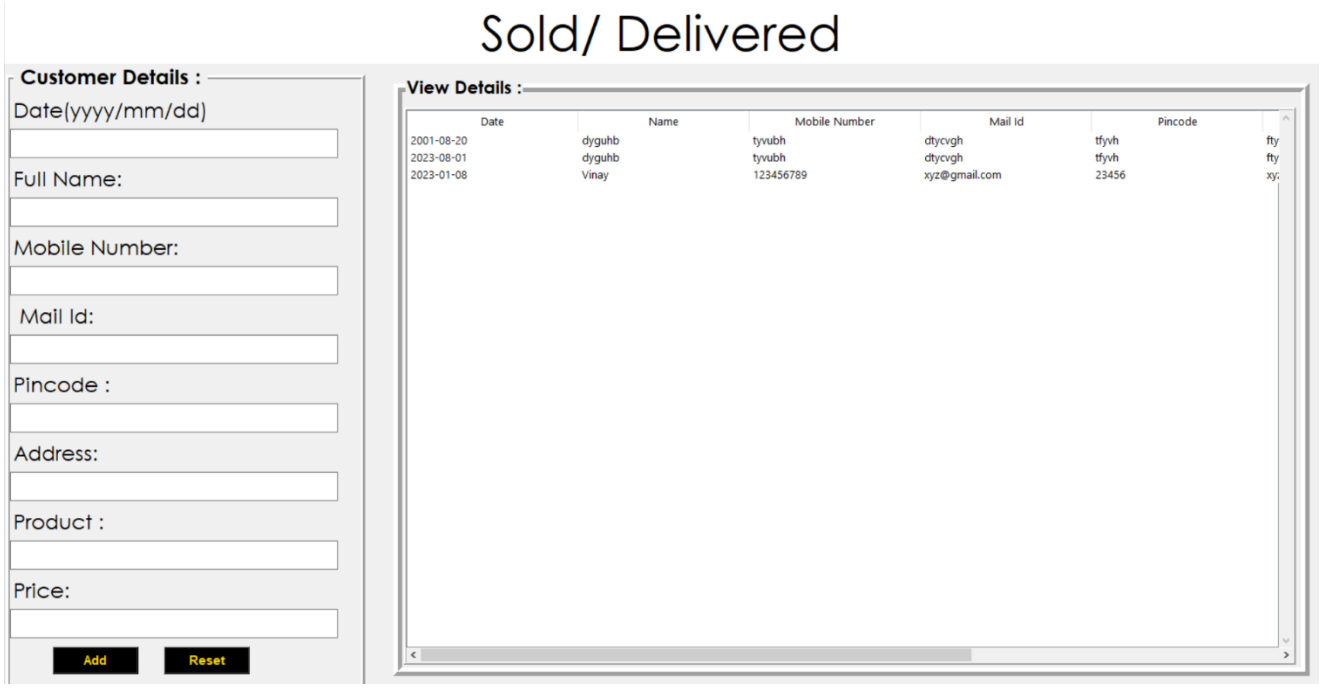
Password: 0987654321



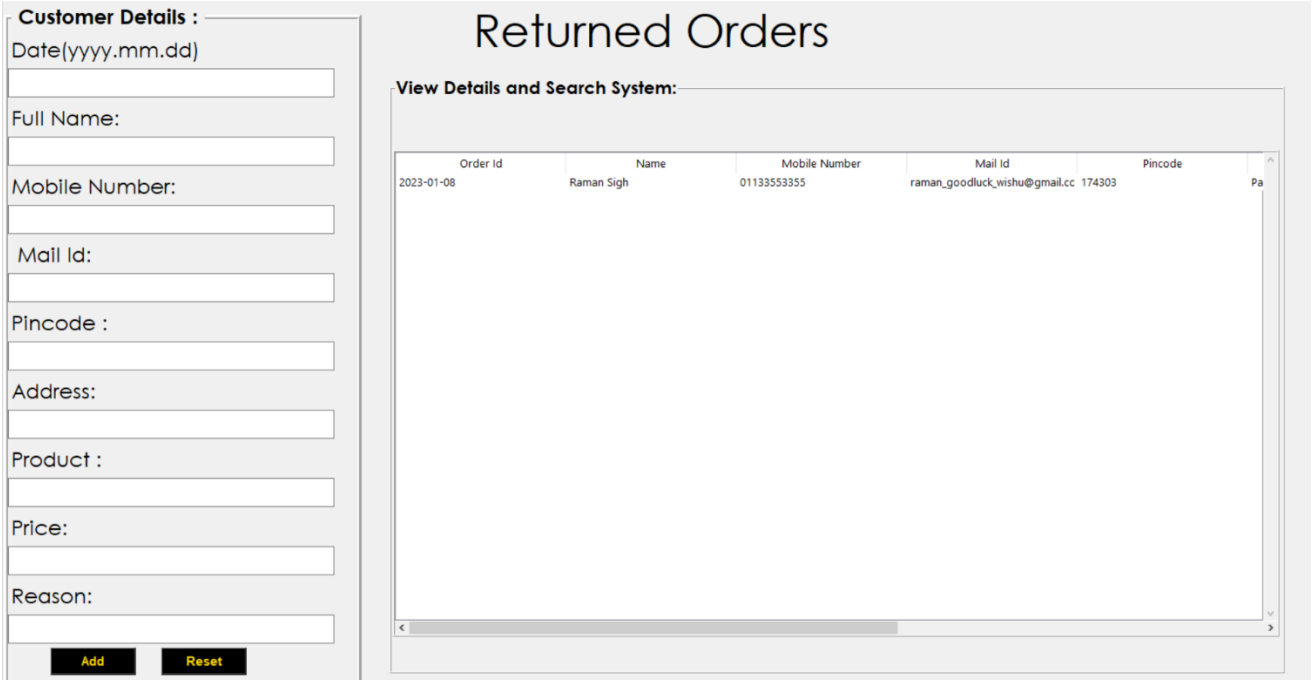
The Home Screen:



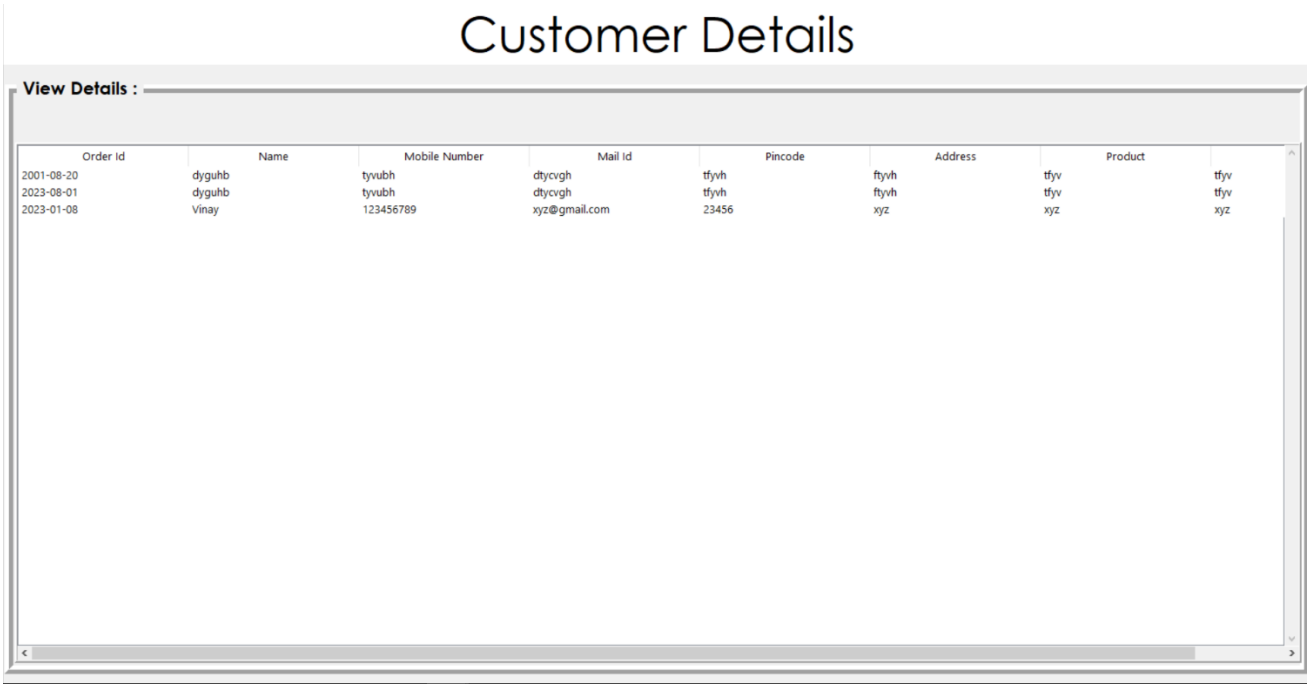
Sales/Delivered:



Returns:

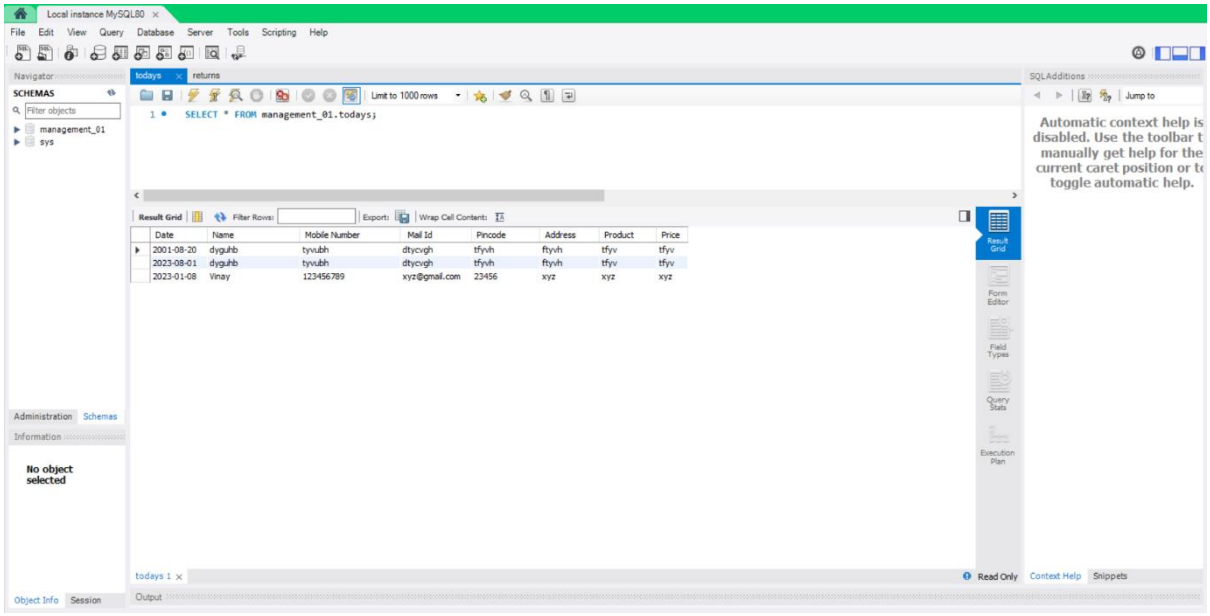


Customer Details:

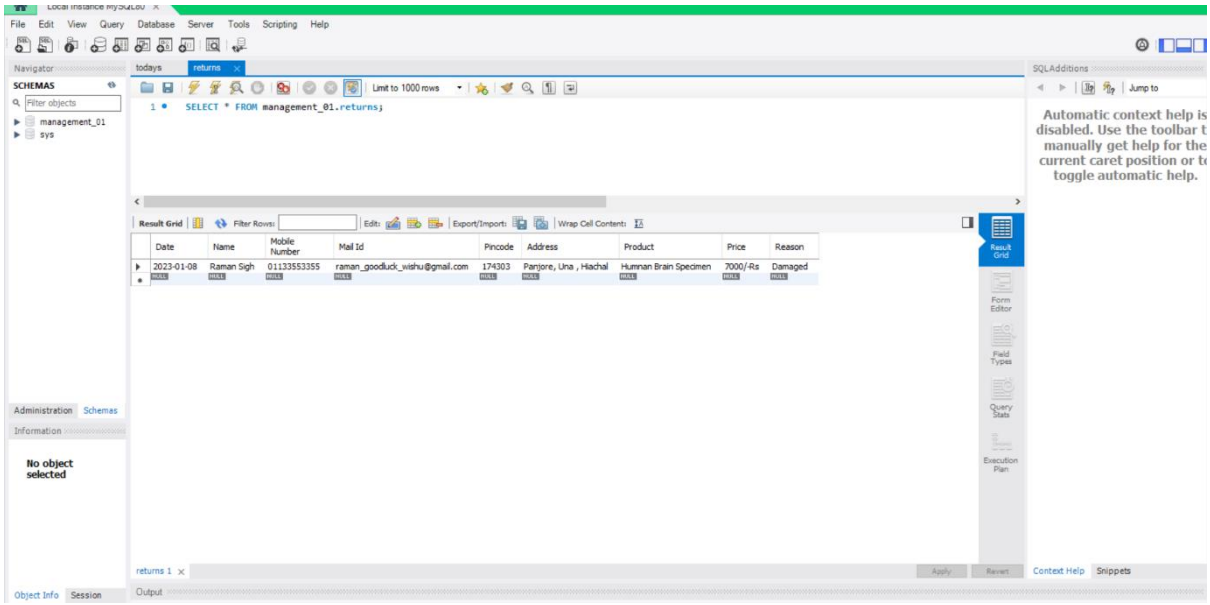


The SQL Database:

Sold/Delivered



Returns



Bibliography:

- Python.org
- Computer Science Class XII NCERT
- Develop responsive and powerful GUI applications with Tkinter by Alan D.Moore
- Think Python How To Think Like A Computer Scientist by Allen B.Downey
- Computer Programming For Beginers by Dylan Mach
- SQL For Dummies by Allen G.Tylor