PYTHON PROJECT REPORT

(Project Term August-November 2021)

Title of the project: **Pizza Delivery System using Python**

Submitted by

B. Bharath   Registration Number: 12003815

Nalli Shiva   Registration Number: 12008888

Course Code: **INT- 213**

Under the Guidance of

Sagar Pande (23754)

School of Computer Science and Engineering

ACKNOWLEDGEMENT:

Executing such a project under the guidance of our mentor Sagar Pande, Assistant Professor, Lovely Professional University for giving us a chance to explore new ideas and expand our knowledge on a topic. With such an opportunity, we are thankful to our mentor and other respected persons who are involved in completing this project. We would like to thank everyone involved directly or indirectly in writing the research paper and now this report for our project.

Regards,

1. B.Bharath
2. Nalli Shiva

INTRODUCTION:

A pizzeria specialized in custom made pizzas is currently taking orders by phone. The current system where the customer calls the pizzeria takes time of employees to answer the phone and is more work consuming than necessary. They want to allow customers to customize and order their pizzas online. The pizzeria also aims to increase the sales, due to the easy to use order online website. The system will give the employees more time to "work" rather then to accept orders by phone, also the potential increase in customers are enough reason for the pizzeria to accept the change (website where customers can order their customized pizzas).

Goal:

Our goal is to deliver a database with a user interface (website) where customers can select various ingredients for their own pizza and place their order. The order will be sent to the "kitchen" where the pizza will be made. The focus is to create an "easy to use" website, which will allow a first time customer to complete their order with ease.

Functional Specifications:

In this Pizza Delivery System Users would be able to

- Order Pizza
- cancel Their Orders
- Track their Order

And vender would be able to access

- Cancelled Orders
- Served Orders
- Deliverd Orders
- Pending Orders

Technical specifications:

The system is made up of three layers. At the top there is the GUI (Graphical User Interface) layer, the middle layer is the storage and query manager, the bottom layer is the underlying database.

GUI layer:

The GUI layer allows users to access the system. All the functionalities of the system must be available through the GUI. There are two separate GUI's. One is for the customers to create orders and one is for employees for processing orders and administration purposes. With scripts, user input will be used to invoke queries from the storage and query manager layer to provide the user with various pages. The GUI should prevent input errors and in case of errors that could not be prevented, provide clear error messages. Nowadays people have usernames and passwords for a lot of websites and services. It is not practical to have users to remember information for a pizzeria. It is much easier for customers to type in their name and address than to have to remember the username and password. Therefore, customers do not have accounts to log on to. However customer information will be stored into the system to allow employees view previous orders by
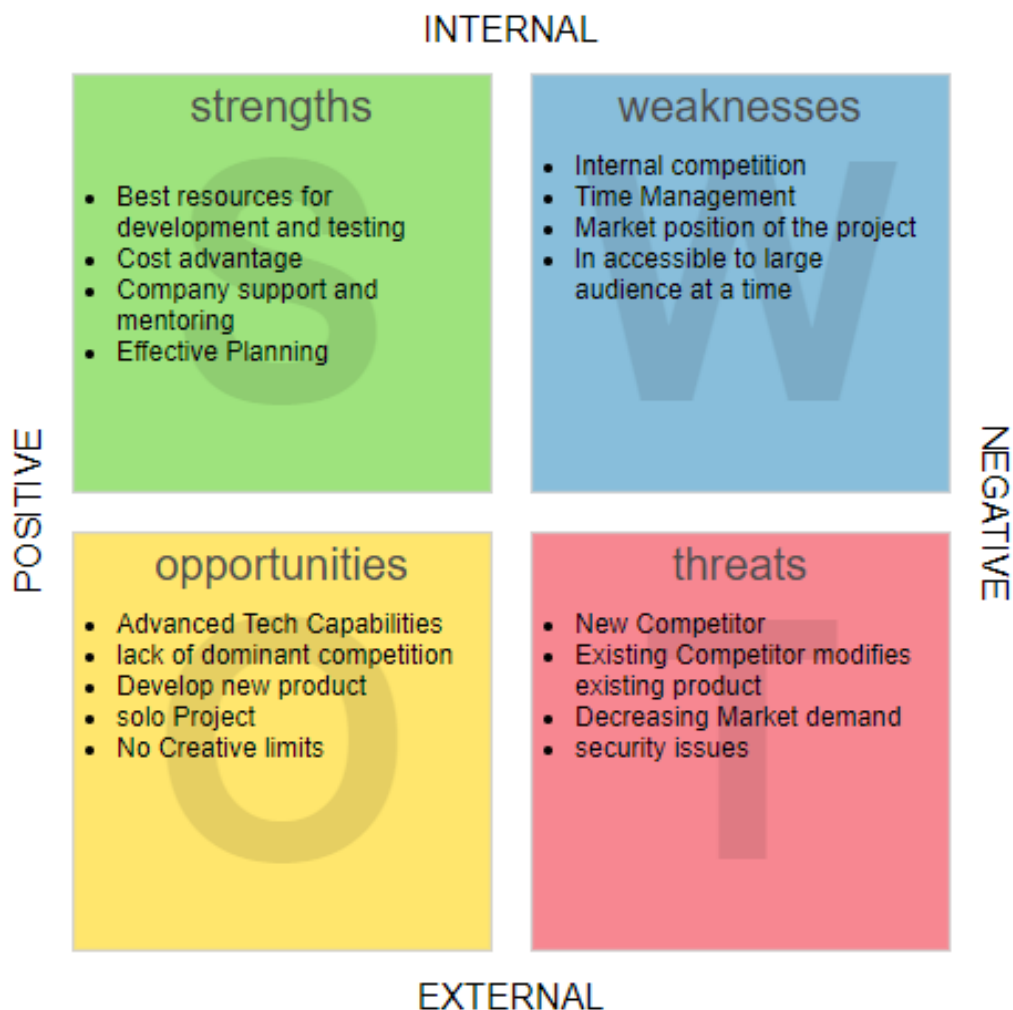
customers. The GUI for employees is on a separate URL. Users need a username and a password to gain access to the system.

Specific Requirements:

To avoid any kind of disturbances or errors at the time of working, we need to follow some specific requirements.
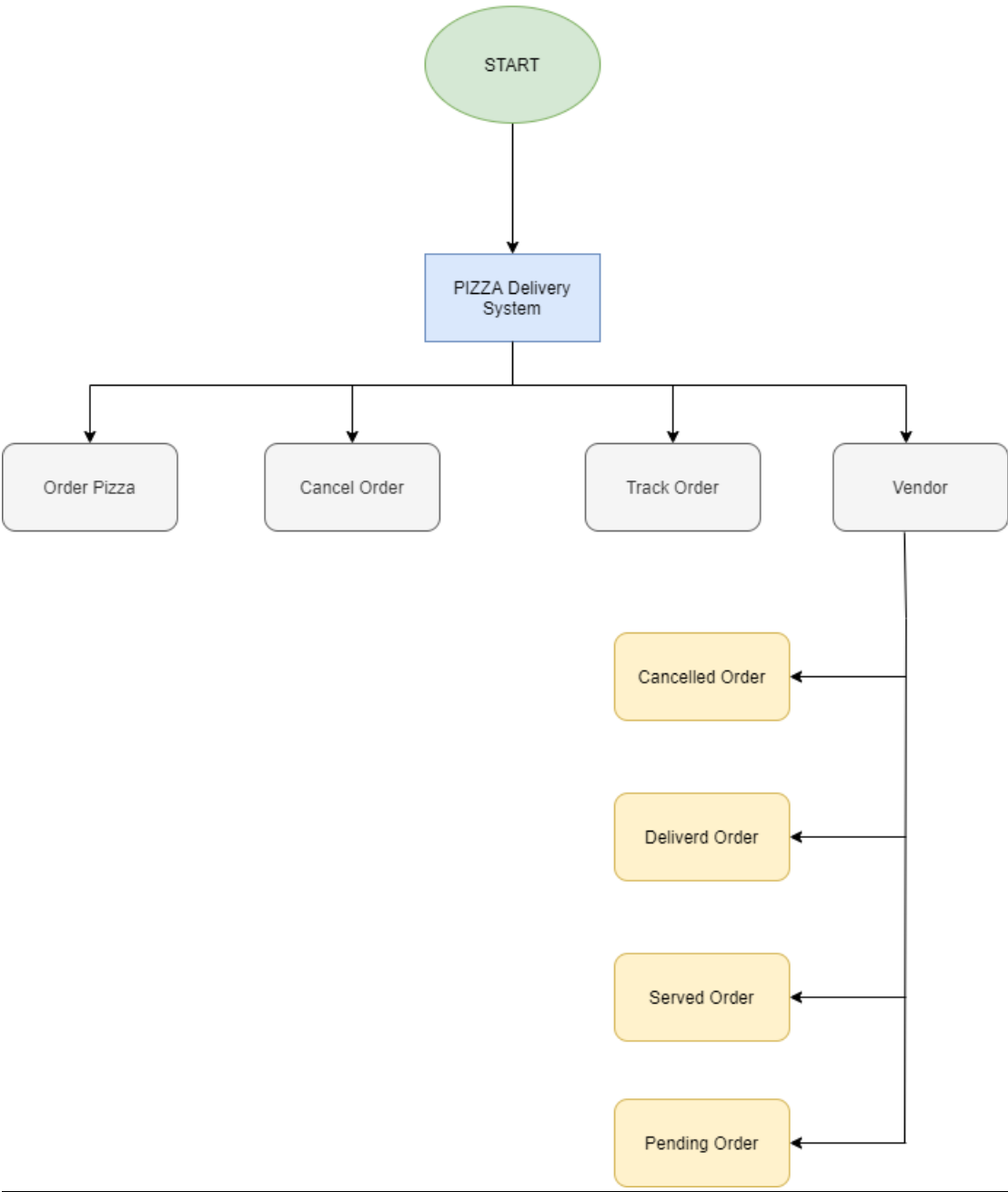
- Operating System(OS): Windows 7 with SPI; Recommended: Windows 10.
- CPU: Intel or AMD processor with 64-bit support; Recommended: 2.8 GHz or faster processor.
- GPU: nVidia GeForce GTX 1050 or equivalent;
     Recommended: nVidia GeForce GTX 1660 or Quadro T1000
- Disk Storage: 4 GB of free disk space.
- Monitor Resolution : 1280 x 800; Recommended: 1920x1080
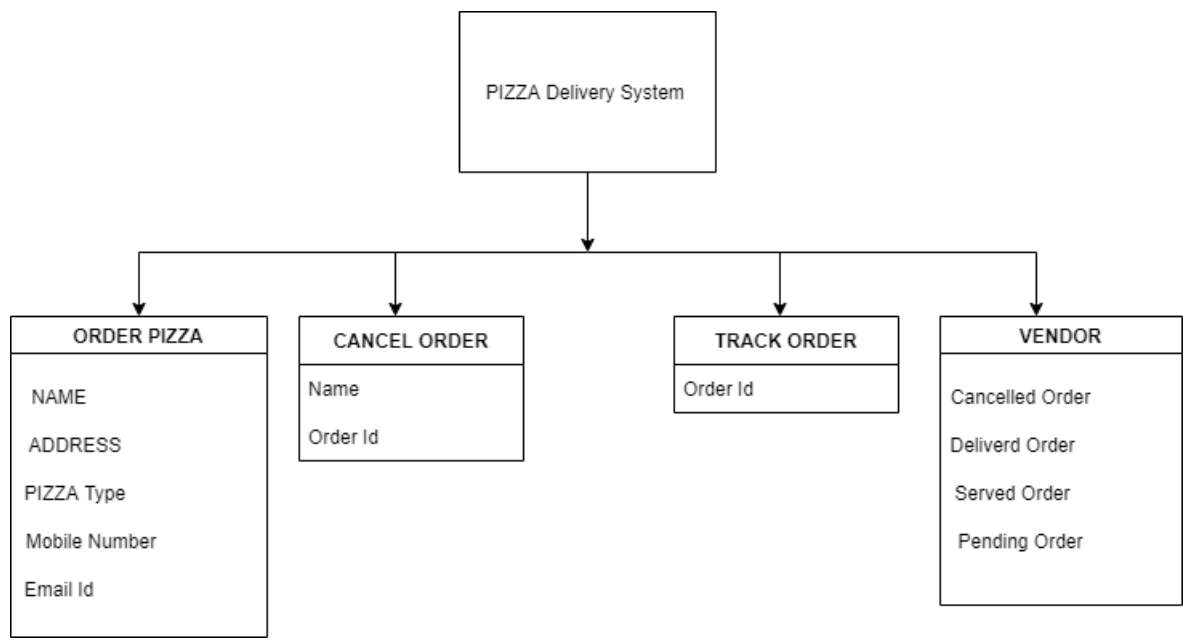- Internet: Internet connection required for software activation.

SWOT Analysis:

INTERNAL

strengths
- Best resources for development and testing
- Cost advantage
- Company support and mentoring
- Effective Planning

weaknesses
- Internal competition
- Time Management
- Market position of the project
- In accessible to large audience at a time

POSITIVE

NEGATIVE

opportunities
- Advanced Tech Capabilities
- lack of dominant competition
- Develop new product
- solo Project
- No Creative limits

threats
- New Competitor
- Existing Competitor modifies existing product
- Decreasing Market demand
- security issues

EXTERNAL

Architecture:

UML DIAGRAM

OF

PIZZA DELIVERY SYSTEM

START

PIZZA Delivery System

Order Pizza

Cancel Order

Track Order

Vendor

Cancelled Order

Deliverd Order

Served Order

Pending Order

Class Diagram

PIZZA Delivery System

ORDER PIZZA

NAME

ADDRESS

PIZZA Type

Mobile Number

Email Id

CANCEL ORDER

Name

Order Id

TRACK ORDER

Order Id

VENDOR

Cancelled Order

Deliverd Order

Served Order

Pending Order

USE CASE DIAGRAM

USER

INPUT

Output

VENDOR

Input

output

PIZZA
DELIVERY
SYSTEM

INSERT DATA

Retrieve data

SERVER

Source code:

```python
from tkinter import *
import sqlite3


db=sqlite3.connect("pizza.sqlite3")
def varr():
    cursor =db.cursor()
    cursor.execute("SELECT * FROM pizza")
    for order_Id,status,Name,mobile,Address,Email,Type in cursor:
        vf=order_Id
    return vf+1;
a=Tk()
a.title("CUSTOMER")
a.geometry('1380x1400')
C = Canvas(a, bg="blue", height=1000, width=1000)
filename = PhotoImage(file = "p2.png")
background_label = Label(a, image=filename)
background_label.place(x=0, y=0, relwidth=1, relheight=1)


C.pack()
def pizza():
    p=Tk()
    p.title("Order Pizza")
    p.geometry('1570x1400')
    l=Label(p,text="Name\n\nAddress\n\nPizza Type\n\nMobile no\n\nEmail id",font=20)
    l.pack()
    l.place(x=200,y=300)
    e=Entry(p)
    e.pack()
    e.place(x=300,y=300)
    e1=Entry(p)
    e1.pack()
    e1.place(x=300,y=350)
    e2=Entry(p)
    e2.pack()
    e2.place(x=300,y=445)
    e3=Entry(p)
    e3.pack()
```

```python
    e3.place(x=300,y=485)
    var = IntVar()
    var.set(1)##
    R1 = Radiobutton(p, text="Small(95 rs)", variable=var, value=0,font=20)
    R1.pack()
    R1.place(x=300,y=385)
    R2 = Radiobutton(p, text="Medium(195 rs)", variable=var, value=1,font=20)
    R2.pack()
    R2.place(x=450,y=385)
    R3 = Radiobutton(p, text="Large(295 rs)", variable=var, value=2,font=20)
    R3.pack()
    R3.place(x=600,y=385)
    def yu():
        if  e.get() and e1.get() and e2.get() and e3.get():
            a=varr()
            db.execute("INSERT INTO pizza
VALUES({},'Pending','{}','{}','{}','{}',{})".format(a,e.get(),e2.get(),e1.get(),e3.get(),var.get()))
            db.commit()
            las=Label(p,text="Ordered sucessfully \n Order Id="+str(a))
            las.pack()
    b7=Button(p,text="submit",font=20,command=yu)
    b7.pack()
    b7.place(x=450,y=500)
b=Button(a,text="Order Pizza",font=20,bg="sky blue",command=pizza)
b.pack()
b.place(x=400,y=550)
def cancel():
    c=Tk()
    c.title("Cancel Order")
    c.geometry('1470x1300')
    l1=Label(c,text="Name\n\nOrder Id",font=20)
    l1.pack()
    l1.place(x=200,y=300)
    e4=Entry(c)
    e4.pack()
    e4.place(x=300,y=300)
    e5=Entry(c)
    e5.pack()
    e5.place(x=300,y=340)
    def can():
```

```python
        cursor =db.cursor()
        cursor.execute("SELECT * FROM pizza")
        for order_Id,status,Name,mobile,Address,Email,Type in cursor:


            if order_Id==int(e5.get()):
                if Name==e4.get():
                    update_sql="UPDATE pizza SET status='Cancelled' WHERE order_Id={}".format(e5.get())
                    update_cursor=db.cursor()
                    update_cursor.execute(update_sql)
                    update_cursor.connection.commit()
                    update_cursor.close()
                    lk=Label(c,text="order cancelled Id="+e5.get())
                    lk.pack()
                    break

    b7=Button(c,text="submit",font=20,command=can)
    b7.pack()
    b7.place(x=300,y=400)
b1=Button(a,text="Cancel Order",font=40,bg="sky blue",command=cancel)
b1.pack()
b1.place(x=570,y=550)
def track():
    t=Tk()
    t.title("Track Order")
    t.geometry('1470x1300')
    l2=Label(t,text="Order Id",font=20)
    l2.pack()
    l2.place(x=200,y=300)
    e6=Entry(t)
    e6.pack()
    e6.place(x=300,y=300)
    def ta():
        cursor =db.cursor()
        cursor.execute("SELECT * FROM pizza")
        for order_Id,status,Name,mobile,Address,Email,Type in cursor:
            if order_Id==int(e6.get()):
                lk=Label(t,text="order status="+status)
                lk.pack()
                break
```

```python
      b8=Button(t,text="submit",font=40,command=ta)
      b8.pack()
      b8.place(x=300,y=350)
b2=Button(a,text="Track Order",font=40,bg="sky blue",command=track)
b2.pack()
b2.place(x=750,y=550)
def vendor():
    v=Tk()
    v.title("Vendor")
    v.geometry('1470x1300')
    def dl():
        d=Tk()
        d.title("Delivered")
        d.geometry('1470x1300')
        l21=Label(d,text="Order Id",font=40)
        l21.pack()
        l21.place(x=200,y=300)
        e16=Entry(d)
        e16.pack()
        e16.place(x=300,y=300)
        def taa():
            cursor =db.cursor()
            cursor.execute("SELECT * FROM pizza")
            for order_Id,status,Name,mobile,Address,Email,Type in cursor:
                if order_Id==int(e16.get()):
                    update_sql="UPDATE pizza SET status='Served' WHERE order_Id={}".format(e16.get())
                    update_cursor=db.cursor()
                    update_cursor.execute(update_sql)
                    update_cursor.connection.commit()
                    update_cursor.close()
                    lk1=Label(d,text="order delivered Id="+e16.get())
                    lk1.pack()
                    break
        b81=Button(d,text="submit",font=20,command=taa)
        b81.pack()
        b81.place(x=300,y=350)
    b21=Button(v,text="Delivered Order",font=20,bg="sky blue",command=dl)
    b21.pack()
    b21.place(x=700,y=200)
    def ca():
```
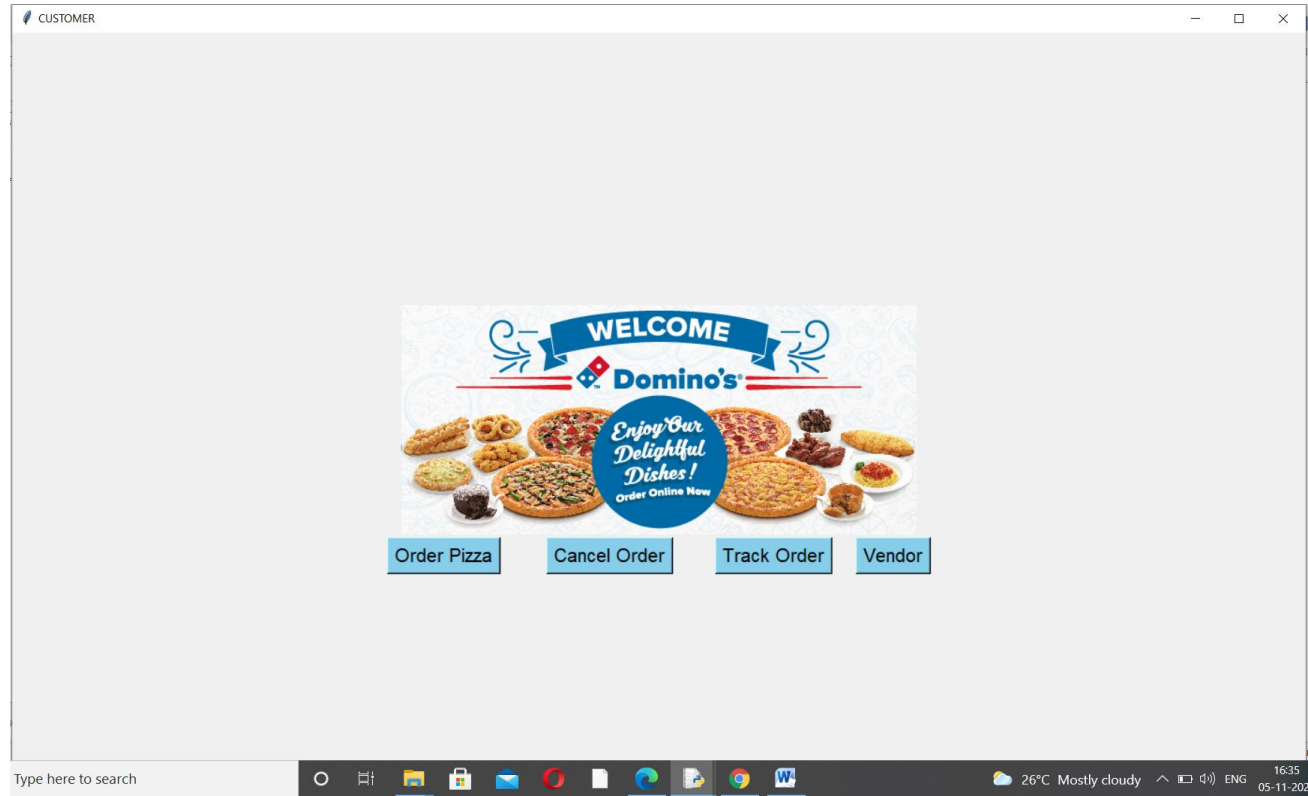
```python
    ca=Tk()
    ca.title("Cancelled Order")
    ca.geometry('1470x1300')
    cursor =db.cursor()
    cursor.execute("SELECT * FROM pizza")
    for order_Id,status,Name,mobile,Address,Email,Type in cursor:
        if status=="Cancelled" :
            lc1=Label(ca,text="Order_Id="+str(order_Id),font=50)
            lc1.pack()
  b22=Button(v,text="Cancelled Order",font=20,bg="sky blue",command=ca)
  b22.pack()
  b22.place(x=400,y=200)
  def se():
    se=Tk()
    se.title("Served Order")
    se.geometry('1470x1300')
    cursor =db.cursor()
    cursor.execute("SELECT * FROM pizza")
    for order_Id,status,Name,mobile,Address,Email,Type in cursor:
        if status=="Served" :
            lc2=Label(se,text="Order_Id="+str(order_Id),font=20)
            lc2.pack()
  b23=Button(v,text="Served Order",font=50,bg="sky blue",command=se)
  b23.pack()
  b23.place(x=400,y=400)
  def pe():
    pe=Tk()
    pe.title("Pending Order")
    pe.geometry('1470x1300')
    cursor =db.cursor()
    cursor.execute("SELECT * FROM pizza")
    for order_Id,status,Name,mobile,Address,Email,Type in cursor:
        if status=="Pending" :
            lc3=Label(pe,text="Order_Id="+str(order_Id),font=20)
            lc3.pack()
  b24=Button(v,text="Pending Order",font=50,bg="sky blue",command=pe)
  b24.pack()
  b24.place(x=700,y=400)
b12=Button(a,text="Vendor",font=20,bg="sky blue",command=vendor)
b12.pack()
```

b12.place(x=900,y=550)

OUT PUT:

Cancel Order

Name

Order Id

submit

Track Order

Order Id

submit