

Objetivo:

 Consolidar los conocimientos adquiridos en clase para la programación de una aplicación en Python.

Enunciado:

 Realizar una aplicación que permita gestionar los pedidos de una empresa almacenando la información dentro de archivos o base de datos (SQLite), para lo cual deben seguir los siguientes pasos:

La aplicación deberá: manejar clientes (se guarda su nombre, dirección, teléfono y e-mail), que pueden realizar pedidos de productos, de los cuales se anota la cantidad en stock. Un cliente puede tener una o varias cuentas para el pago de los pedidos. Cada cuenta está asociada a una tarjeta de crédito, y tiene una cierta cantidad disponible de dinero, que el cliente debe aumentar periódicamente para poder realizar nuevos pedidos.

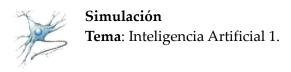
Un cliente puede empezar a realizar un pedido sólo si tiene alguna cuenta con dinero disponible. Además, sólo es posible realizar peticiones de productos en stock.

Existe una clase responsable del cobro, orden de distribución y confirmación de los pedidos. El cobro se realiza al finalizar el pedido. Si una cuenta no tiene suficiente dinero, el pedido se rechaza. Una vez que el pedido está listo para servirse, se ordena su distribución, y una vez entregado, pasa a estar confirmado.

* Se aprobará como puntos adicionales a la practicas si se realiza una implementación visual utilizando cualquier librería GUI (Tkinter – 1 punto) o mejor aún con Flask(3 puntos).

Finalmente, exportar un PDF del cuaderno de Jupyter Notebook visualizando el funcionamiento y validación del sistema.

Plazo: Se debe presentar el sistema funcionando hasta las **23:55 del 09/05/2021**, la misma que deberá ser subida al git personal y adicionalmente dentro de un cuaderno de Jupyter Notebook.





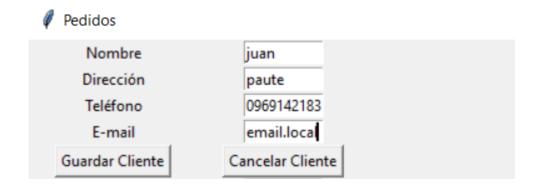
Nombre: Walter Bau

Página Principal:

Sistema de Pedidos								_
Nombre								
Dirección								
Teléfono								
E-mail								
Guardar Cliente	Cancelar Cliente							
Saldo								
Tarjeta Credito								
Nombre Cliente		Buscar Cliente	Dirección Cliente					
Guardar Cuenta	Cancelar Cuenta		Teléfono Cliente					
Nombre Producto			Email Cliente					
Precio Producto								
Stock Producto								
Guardar Producto	Cancelar Producto							
		- Pedidos -		 				
Nombre Cliente		Buscar Cliente	Nombre Producto	Buscar Producto				
Saldo			Stock					
Dirección Cliente			Precio					
Teléfono Cliente			Stock a Recargar	Recargar				
Email Cliente			•					
			Cantidad	Calcular				
			Total	Comprar				
Saldo a Recargar		Pagagan	10101	Compius				
Saldo a Recargar		Recargar						

Funcionamiento:

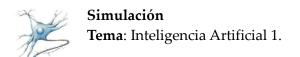
Registrar Cliente:



Al dar click en el botón guardar cliente, los datos ingresados se guardan en la base de datos, el botón cancelar borra los datos.

Cuenta:

Al dar click en buscar cliente se llenarán la dirección, teléfono y el correo automáticamente del





nombre que se escribió en el cuadro de texto, se debe primero buscar el cliente antes de crear la cuenta para que se genere la consulta en donde se podrá guardar la cuenta.

Al dar click en guard	ar cuenta se ingresa	n los datos en la b	ase de datos		
Saldo	100	Duran Cliente	Dirección Cliente		
Tarjeta Credito	1				
Nombre Cliente	juan		Teléfono Cliente Email Cliente		
Guardar Cuenta	Cancelar Cuenta				
Al dar click en busc	car clientes				
Saldo					
Tarjeta Credito					
Nombre Cliente	juan	Buscar Cliente	Dirección C	liente	paute
Guardar Cuenta	Cancelar Cuenta		Teléfono Cl	iente	0969142183

Productos:

Al dar click en el botón guardar producto, los datos ingresados se guardan en la base de datos, el botón cancelar borra los datos.

Nombre Producto	iphone
Precio Producto	600
Stock Producto	4
Guardar Producto	Cancelar Producto

Realizar Pedido:

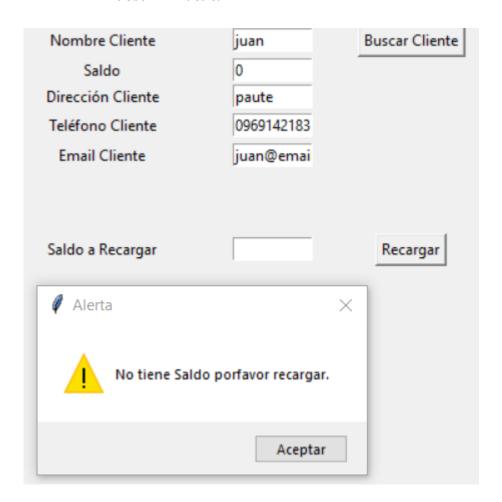
Buscar Cliente:



Tema: Inteligencia Artificial 1.



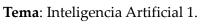
Prueba 1 - Practica



Al dar click en buscar cliente, y si no contiene suficiente saldo saldrá el siguiente recuadro de alarma y tendrá que recargar saldo para poder realizar el pedido.

Después de dar click al recargar se actualiza el saldo del cliente

Nombre Cliente	juan	Buscar Cliente
Saldo	10.0	
Dirección Cliente	paute	
Teléfono Cliente	0969142183	
Email Cliente	juan@emai	
Saldo a Recargar		Recargar





Buscar Producto:

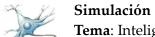
Nombre Producto	iphone	Buscar Producto			
Stock	2				
Precio	600.0				
Stock a Recargar		Recargar			

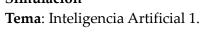
Al dar click en buscar producto se mostrará el stock y el precio, si no existe stock saltará una alarma de que no existe stock y se podrá actualizar el stock.

Nombre Pr	oducto	iphone		Buscar Producto		
Stock	c	0				
Preci	0	600.0				
Stock a Re	cargar			Recargar		
	Alerta			× 「		
Cantio				- [
Tota	No existe stock porfavor recargar.					
		А	ceptar			

Cuando se da click al calcular se realizara el total y después al de comprar luego se actualizara el stock del producto

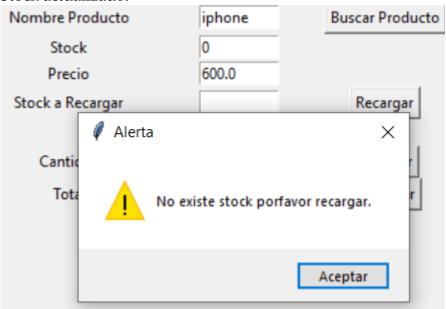
Nombre Producto	iphone	Buscar Producto
Stock	2	
Precio	600.0	
Stock a Recargar		Recargar
Cantidad	4	Calcular
Total	1200.0	Comprar







Stock actualizado.



Codigo Fuente:

Simulación Tema: Inteligencia Artificial 1.



```
In [9]: ▶ from tkinter import *
            from tkinter import messagebox as MessageBox
           import sqlite3
           connection = sqlite3.connect('Pedidos.db')
           ventana = Tk()
           ventana.title("Sistema de Pedidos")
           ventana.geometry('1000x700')
           lbl = Label(ventana, text="Nombre")
           lbl.grid(column=0, row=0)
           lbl1 = Label(ventana, text="Dirección")
           lbl1.grid(column=0, row=1)
           lbl2 = Label(ventana, text="Teléfono")
           lbl2.grid(column=0, row=2)
           lbl3 = Label(ventana, text="E-mail")
           lbl3.grid(column=0, row=3)
           lbl4 = Label(ventana, text="Saldo")
           lbl4.grid(column=0, row=5)
           lbl5 = Label(ventana, text="Tarjeta Credito")
           lbl5.grid(column=0, row=6)
           lbl6 = Label(ventana, text="Nombre Cliente")
            lbl6.grid(column=0, row=7)
            lbl6 = Label(ventana, text="Dirección Cliente")
            lbl6.grid(column=3, row=7)
            lbl7 = Label(ventana, text="Teléfono Cliente")
            lbl7.grid(column=3, row=8)
            lbl8 = Label(ventana, text="Email Cliente")
            lbl8.grid(column=3, row=9)
            lbl9 = Label(ventana, text="Nombre Producto")
            lbl9.grid(column=0, row=9)
            lbl10 = Label(ventana, text="Precio Producto")
            lbl10.grid(column=0, row=10)
            lbl11 = Label(ventana, text="Stock Producto")
            lbl11.grid(column=0, row=11)
            lbl12 = Label(ventana, text="----")
            lbl12.grid(column=0, row=13)
            lbl13 = Label(ventana, text="----")
            lbl13.grid(column=1, row=13)
            lbl14 = Label(ventana, text="Pedidos")
```

Simulación Tema: Inteligencia Artificial 1.



```
lbl14.grid(column=2, row=13)
lbl15 = Label(ventana, text="----")
lbl15.grid(column=3, row=13)
lbl16 = Label(ventana, text="----")
lbl16.grid(column=4, row=13)
lbl17 = Label(ventana, text="Nombre Cliente")
lbl17.grid(column=0, row=14)
lbl18 = Label(ventana, text="Nombre Producto")
lbl18.grid(column=3, row=14)
lbl19 = Label(ventana, text="Saldo")
lbl19.grid(column=0, row=15)
lbl20 = Label(ventana, text="Stock")
lbl20.grid(column=3, row=15)
lbl21 = Label(ventana, text="Precio")
lbl21.grid(column=3, row=16)
lbl22 = Label(ventana, text="Dirección Cliente")
lbl22.grid(column=0, row=16)
lbl23 = Label(ventana, text="Teléfono Cliente")
lbl23.grid(column=0, row=17)
lbl24 = Label(ventana, text="Email Cliente")
lbl24.grid(column=0, row=18)
lbl25 = Label(ventana, text="Cantidad")
lbl25.grid(column=3, row=19)
lbl26 = Label(ventana, text="Total")
lbl26.grid(column=3, row=20)
lbl27 = Label(ventana, text="Saldo a Recargar")
lbl27.grid(column=0, row=21)
lbl28 = Label(ventana, text="Stock a Recargar")
lbl28.grid(column=3, row=17)
txt28 = Entry(ventana, width=10)
txt28.grid(column=4, row=17)
txt = Entry(ventana, width=10)
txt.grid(column=1, row=0)
txt1 = Entry(ventana,width=10)
txt1.grid(column=1, row=1)
txt2 = Entry(ventana, width=10)
txt2.grid(column=1, row=2)
txt3 = Entry(ventana,width=10)
txt3.grid(column=1. row=3)
```

15

Simulación

Tema: Inteligencia Artificial 1.



```
txt4 = Entry(ventana, width=10)
txt4.grid(column=1, row=5)
txt5 = Entry(ventana, width=10)
txt5.grid(column=1, row=6)
txt6 = Entry(ventana,width=10)
txt6.grid(column=1, row=7)
txt7 = Entry(ventana, width=10)
txt7.grid(column=4, row=7)
txt8 = Entry(ventana, width=10)
txt8.grid(column=4, row=8)
txt9 = Entry(ventana, width=10)
txt9.grid(column=4, row=9)
txt10 = Entry(ventana, width=10)
txt11 = Entry(ventana, width=10)
txt11.grid(column=1, row=9)
txt12 = Entry(ventana, width=10)
txt12.grid(column=1, row=10)
txt13 = Entry(ventana, width=10)
txt13.grid(column=1.row=11)
txt14 = Entry(ventana,width=10)
txt14.grid(column=1, row=14)
txt15 = Entry(ventana, width=10)
txt15.grid(column=4, row=14)
txt16 = Entry(ventana, width=10)
txt16.grid(column=1, row=15)
txt17 = Entry(ventana, width=10)
txt17.grid(column=4, row=15)
txt18 = Entry(ventana, width=10)
txt18.grid(column=4, row=16)
txt19 = Entry(ventana, width=10)
txt19.grid(column=1, row=16)
txt20 = Entry(ventana, width=10)
txt20.grid(column=1, row=17)
txt21 = Entry(ventana, width=10)
txt21.grid(column=1, row=18)
txt22 = Entry(ventana, width=10)
txt22.grid(column=4, row=19)
txt23 = Entry(ventana, width=10)
txt23.grid(column=4, row=20)
txt24 = Entry(ventana, width=10)
txt25 = Entry(ventana,width=10)
txt26 = Entry(ventana, width=10)
txt26.grid(column=1, row=21)
```



Tema: Inteligencia Artificial 1.



```
def sql_insertar_Cliente(con, entities):
    cursorObj = con.cursor()
    cursorObj.execute('INSERT INTO cliente(nombre, dirección, telefono, email) VALUES( ?, ?, ?, ?)', entities)
    con.commit()
def guardarCliente():
    con = sqlite3.connect('Pedidos.db')
    entities = (txt.get(), txt1.get(), txt2.get(), txt3.get())
    sql_insertar_Cliente(con, entities)
    txt.delete(0, END)
    txt1.delete(0, END)
txt2.delete(0, END)
    txt3.delete(0, END)
btn = Button(ventana, text="Guardar Cliente", command=guardarCliente)
btn.grid(column=0, row=4)
def clickedCancelar():
    txt.delete(0, END)
    txt1.delete(0, END)
txt2.delete(0, END)
    txt3.delete(0, END)
btn1 = Button(ventana, text="Cancelar Cliente", command=clickedCancelar)
btn1.grid(column=1, row=4)
def sql_datos_Cliente():
    con = sqlite3.connect('Pedidos.db')
    cursorObj = con.cursor()
    primera='select * from cliente where nombre = "'
nombre= ""
    nombre=txt6.get()
    consult = primera+nombre+'"'
    cursorObj.execute(consult)
    rows = cursorObj.fetchall()
    for row in rows:
        txt7.insert(0, row[2])
txt8.insert(0, row[3])
txt9.insert(0, row[4])
        txt10.insert(0, row[0])
btn4 = Button(ventana, text="Buscar Cliente", command=sql_datos_Cliente)
```

Tema: Inteligencia Artificial 1.



```
def sql_insertar_Cuenta(con, entities):
   cursorObj = con.cursor()
    cursorObj.execute('INSERT INTO cuenta(saldo, num_tarjeta, idcliente) VALUES( ?, ?, ?)', entities)
   con.commit()
def guardarCuenta():
   con = sqlite3.connect('Pedidos.db')
   entities = (txt4.get(), txt5.get(), txt10.get())
   sql_insertar_Cuenta(con, entities)
    txt4.delete(0, END)
   txt5.delete(0, END)
    txt7.delete(0, END)
   txt8.delete(0, END)
txt9.delete(0, END)
    txt10.delete(0, END)
btn2 = Button(ventana, text="Guardar Cuenta", command=guardarCuenta)
btn2.grid(column=0, row=8)
def clickedCancelarCuenta():
    txt4.delete(0, END)
    txt5.delete(0, END)
    txt6.delete(0, END)
    txt7.delete(0, END)
    txt8.delete(0, END)
    txt9.delete(0, END)
    txt10.delete(0, END)
btn3 = Button(ventana, text="Cancelar Cuenta", command=clickedCancelarCuenta)
btn3.grid(column=1, row=8)
def sql_insertar_Producto(con, entities):
    cursorObj = con.cursor()
    cursorObj.execute('INSERT INTO producto(nombre, precio, stock) VALUES( ?, ?, ?)', entities)
    con.commit()
def guardarProducto():
   con = sqlite3.connect('Pedidos.db')
   entities = (txt11.get(), txt12.get(), txt13.get())
    sql_insertar_Producto(con, entities)
    txt11.delete(0, END)
    txt12.delete(0, END)
    txt13.delete(0, END)
btn4 = Button(ventana, text="Guardar Producto", command=guardarProducto)
```

Tema: Inteligencia Artificial 1.



```
def clickedCancelarProducto():
    txt11.delete(0, END)
    txt12.delete(0, END)
    txt13.delete(0, END)
btn5 = Button(ventana, text="Cancelar Producto", command=clickedCancelarProducto)
btn5.grid(column=1, row=12)
def sql_datos_Cliente1():
    con = sqlite3.connect('Pedidos.db')
    primera='select * from cliente c, cuenta cu where nombre = "'
nombre= ""
    nombre=txt14.get()
    consult = primera+nombre+'" AND c.id = cu.idcliente'
    cursorObj.execute(consult)
    rows = cursorObj.fetchall()
    sald=0
    for row in rows:
        txt19.insert(0, row[2])
         txt20.insert(0, row[3])
         txt21.insert(0, row[4])
        txt16.insert(0, row[6])
txt25.insert(0, row[5])
        sald = row[6]
    if sald == 0:
        MessageBox.showwarning("Alerta", "No tiene Saldo porfavor recargar.")
btn6 = Button(ventana, text="Buscar Cliente", command=sql_datos_Cliente1)
btn6.grid(column=2, row=14)
def sql_datos_Cliente1():
    con = sqlite3.connect('Pedidos.db')
    cursorObj = con.cursor()
    primera='select * from producto where nombre = "'
nombre= ""
    nombre=txt15.get()
   consult = primera+nombre+'"'
   cursorObj.execute(consult)
    rows = cursorObj.fetchall()
    stock = 0
    for row in rows:
        txt18.insert(0, row[2])
        txt17.insert(0, row[3])
        txt24.insert(0, row[0])
    stock =row[3]
if stock == 0:
        MessageBox.showwarning("Alerta", "No existe stock porfavor recargar.")
btn6 = Button(ventana, text="Buscar Producto", command=sql_datos_Cliente1)
btn6.grid(column=5, row=14)
def sumar_Cantidad():
    suma = 0
    aux1 = 0
    aux2 = 0
    aux1 = txt22.get()
    aux2 = txt18.get()
suma = float(aux1) * float(aux2)
    txt23.insert(0, float(suma))
btn7 = Button(ventana, text="Calcular", command=sumar_Cantidad)
btn7.grid(column=5, row=19)
```



Tema: Inteligencia Artificial 1.



```
def sql_update_Producto():
      con = sqlite3.connect('Pedidos.db')
      cursorObj = con.cursor()
      num1 = 0
num2 = 0
     num3 = 0
num1 = int(txt17.get())
num2 = int(txt22.get())
num3 = int(num1)-int(num2)
      num11 = 0
      num21 = 0
      num31 = 0
     num11 = 0
num11 = float(txt16.get())
num21 = float(txt23.get())
num31 = float(num11)-float(num21)
tx11 = "UPDATE cuenta SET saldo = " + str(num31)
tx21 = " where idC = " + str(txt25.get())
tx31 = tx11+tx21
     tx1 = "UPDATE producto SET stock = " + str(num3)
tx2 = "where idProducto = " + str(txt24.get())
      tx3 = tx1 + tx2
      cursorObj.execute(tx3)
      con.commit()
      txt18.delete(0, END)
      txt17.delete(0, END)
      txt24.delete(0, END)
      txt15.delete(0, END)
txt22.delete(0, END)
      txt23.delete(0, END)
      cursorObj1 = con.cursor()
      cursorObj1.execute(tx31)
      con.commit()
      txt19.delete(0, END)
      txt20.delete(0, END)
      txt21.delete(0, END)
      txt25.delete(0, END)
      txt16.delete(0, END)
      txt14.delete(0, END)
btn8 = Button(ventana, text="Comprar", command=sql_update_Producto)
btn8.grid(column=5, row=20)
def sql_update_Cliente():
      con = sqlite3.connect('Pedidos.db')
      cursorObj1 = con.cursor()
      num11 = 0
      num21 = 0
      num31 = 0
     num31 = 0
num11 = float(txt16.get())
num21 = float(txt26.get())
num31 = float(num11)+float(num21)
tx11 = "UPDATE cuenta SET saldo = " + str(num31)
tx21 = " where idC = " + str(txt25.get())
tx31 = tx11 +tx21
      cursorObj1.execute(tx31)
      con.commit()
      txt16.delete(0, END)
      txt16.insert(0, num31)
txt26.delete(0, END)
```

15

Simulación

Tema: Inteligencia Artificial 1.



```
btn9 = Button(ventana, text="Recargar", command=sql_update_Cliente)
btn9.grid(column=2, row=21)

def sql_update_Stock():
    con = sqlite3.connect('Pedidos.db')
    cursorObj = con.cursor()
    num1 = 0
    num1 = int(txt28.get())
    tx1 = "UPDATE producto SET stock = " + str(num1)
    tx2 = " where idProducto = " + str(txt28.get())
    tx3 = tx1 +tx2
    cursorObj.execute(tx3)
    con.commit()
    txt17.delete(0, END)
    txt17.insert(0, num1)

btn10 = Button(ventana, text="Recargar", command=sql_update_Stock)
btn10.grid(column=5, row=17)
ventana.mainloop()
```