



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 prácticas 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: Esteban Sibri

Página Persona:

PRUEBA

Persona Saldo Producto Pedidos

Nombre

Direccion

Telefono

Email

Guardar cliente

Cancelar cliente

Página Saldo:

PRUEBA

Persona Saldo Producto Pedidos

Saldo

Tarjeta de Credito

Nombre CLiente

Guardar Cuenta

Cancelar Cuenta

Página Producto:



The screenshot shows a window titled 'PRUEBA' with a green header bar. Below the header is a tabbed interface with four tabs: 'Persona', 'Saldo', 'Producto', and 'Pedidos'. The 'Producto' tab is currently selected. It contains three input fields: 'Nombre Producto', 'Precio Producto', and 'Stock Producto'. To the right of these fields are two buttons: 'Guardar Producto' and 'Cancelar Producto'.

Página Pedidos:

The screenshot shows the same 'PRUEBA' window, but now the 'Pedidos' tab is selected. It contains a form with several input fields: 'Nombre', 'Direccion', 'Telefono', 'Email', 'Nombre Producto', 'Precio Producto', 'Stock', 'Cantidad', 'Total', 'Saldo', and 'Recargar Saldo'. To the right of these fields are four buttons: 'Buscar Producto', 'Buscar Cliente', 'Recargar', and 'Comprar'.

Funcionamiento:

Registrar Cliente:



PRUEBA

Persona Saldo Producto Pedidos

Nombre esteban

Direccion totoracocha

Telefono 098671340

Email esibri@est.ups.edu.ed

Guardar cliente

Cancelar 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 guardar cuenta se ingresan los datos en la base de datos

PRUEBA

Persona Saldo Producto Pedidos

Saldo 600

Tarjeta de Credito 022569821

Nombre Cliente esteban

Guardar Cuenta

Cancelar Cuenta

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.



PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre Producto	Carne		
Precio Producto	4		Guardar Producto
Stock Producto	10		Cancelar Producto

Realizar Pedido:

Para realizar el pedido se debe primero ingresar el nombre del cliente y presionar el botón buscar cliente el cual cargara los datos del usuario, luego se debe escribir el nombre del producto y presionar el botón buscar producto el cual cargara los datos del producto al ser presionado

PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre	esteban		
Direccion	totoracocha		Buscar Producto
Telefono	0986712340		Buscar Cliente
Email	esibri@est.ups.edu.ec		Recargar
Nombre Producto	Carne		Calcular
Precio Producto	4.0		Comprar
Stock	10.0		
Cantidad	11		
Total			
Saldo			
Recargar Saldo			

Luego debemos ingresar la cantidad a comprar, si presionamos el botón calcular este nos mostrara el valor total de la compra en el campo total



The 'PRUEBA' application window contains a form with the following fields and buttons:

Persona	Saldo	Producto	Pedidos
Nombre		esteban	
Direccion		totoracocha	
Telefono		98671340.0	
Email		esibri@est.ups.edu.ec	
Nombre Producto		Carne	
Precio Producto		4.0	
Stock		10.0	
Cantidad		11	
Total		44	
Saldo			
Recargar Saldo			

Buttons on the right side of the form:

- Buscar Producto
- Buscar Cliente
- Recargar
- Calcular
- Comprar

Luego se presiona el botón comprar, si excede el valor de stock se mostrará en pantalla un mensaje de advertencia.

The 'PRUEBA' application window is shown with an 'Alerta' dialog box overlaid. The dialog box contains a yellow warning triangle icon and the text 'No hay en stock.' with an 'Aceptar' button.

Background form fields (partially visible):

Persona	Saldo	Producto	Pedidos
Nombre		esteban	
Direccion		t	
Telefono		9	
Email		e	
Nombre Producto		C	
Precio Producto		4	
Stock		1	
Cantidad		1	
Total		44	
Saldo			
Recargar Saldo			

Buttons on the right side of the form (partially visible):

- Buscar Producto
- Buscar Cliente
- Recargar
- Calcular
- Comprar

Si el producto se encuentra en stock y el usuario no tiene saldo en la tarjeta se mostrará el siguiente mensaje



PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre	esteban		
Direccion			
Telefono			
Email			
Nombre Producto			
Precio Producto			
Stock			
Cantidad	10		
Total	40		
Saldo			
Recargar Saldo			

Alerta

! No tiene saldo.

Aceptar

Buscar Producto

Buscar Cliente

Recargar

Calcular

Comprar

Si cumple con todos los requisitos se puede realizar la compra y se mostrara la disminucion del saldo en la base de datos.

DB Browser for SQLite - C:\Users\esteb\OneDrive\Documentos\IASEGUNDAmat\IA\mibase.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database

Database Structure Browse Data Edit Pragmas Execute SQL Edit Database Cell

Table: cuenta

saldo	tarjeta	nombre
Filter	Filter	Filter
1	100	22298... esteban

PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre	esteban		
Direccion	totoracocha		
Telefono	98671340.0		
Email	esibri@est.ups.edu.ec		
Nombre Producto	Carne		
Precio Producto	4.0		
Stock	10.0		
Cantidad	10		
Total	40		
Saldo			
Recargar Saldo			

Microsoft Search (Alt+Q)

Just start typing here to bring features to your fingertips and get help.

Tell me more

Go to: 1

SQL Log Plot DB Schema Remote

UTF-8



Simulación Tema: Inteligencia Artificial 1.

DB Browser for SQLite - C:\Users\esteb\OneDrive\Documentos\IASEGUNDAmat\IA\mibase.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database

Database Structure Browse Data Edit Pragmas Execute SQL Edit Database Cell

Table: cuenta

	saldo	tarjeta	nombre
1	50.0	22298...	esteban

Go to: 1

SQL Log Plot DB Schema Remote

UTF-8

PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre		esteban	
Direccion		totoracocha	
Telefono		98671340.0	
Email		esibri@est.ups.edu.ec	
Nombre Producto		pollo	
Precio Producto		5.0	
Stock		20.0	
Cantidad		10	
Total		50	
Saldo		50	
Recargar Saldo			

Buscar Producto

Buscar Cliente

Recargar

Calcular

Comprar

Para recargar saldo en la tarjeta del cliente se debe ingresar el nombre del cliente y rellenar el campo recargar saldo

DB Browser for SQLite - C:\Users\esteb\OneDrive\Documentos\IASEGUNDAmat\IA\mibase.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database

Database Structure Browse Data Edit Pragmas Execute SQL Edit Database Cell

Table: cuenta

	saldo	tarjeta	nombre
1	50.0	22298...	esteban

Go to: 1

SQL Log Plot DB Schema Remote

UTF-8

PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre		esteban	
Direccion		totoracocha	
Telefono		98671340.0	
Email		esibri@est.ups.edu.ec	
Nombre Producto		pollo	
Precio Producto		5.0	
Stock		20.0	
Cantidad		10	
Total		50	
Saldo		50	
Recargar Saldo		100	

Buscar Producto

Buscar Cliente

Recargar

Calcular

Comprar



Simulación Tema: Inteligencia Artificial 1.

DB Browser for SQLite - C:\Users\esteb\OneDrive\Documents\IASEGUNDAmat\IA\mibase.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database

Database Structure Browse Data Edit Pragma Execute SQL Edit Database Cell

Table: cuenta

	saldo	tarjeta	nombre
1	150.0	22298...	esteban

1 - 1 of 1 Go to: 1

UTF-8

PRUEBA

Persona	Saldo	Producto	Pedidos
Nombre		esteban	
Direccion		totoracocha	
Telefono		98671340.0	
Email		esibri@est.ups.edu.ec	
Nombre Producto		pollo	
Precio Producto		5.0	
Stock		20.0	
Cantidad		10	
Total		50	
Saldo		50	
Recargar Saldo		100	

Buscar Producto
Buscar Cliente
Recargar
Calcular
Comprar

Apply

SQL Log Plot DB Schema Remote

Codigo Fuente:

```
class Aplicacion:
    def cliente(self):
        nombre=self.entry.get()
        direccion=self.entry1.get()
        telefono=self.entry2.get()
        email=self.entry3.get()
        cursorObj.execute('INSERT INTO cliente VALUES (?, ?, ?, ?)', (nombre,direccion,telefono,email))
        con.commit()

    def cuenta(self):
        saldo=self.entry4.get()
        tarjeta=self.entry5.get()
        nombreC=self.entry6.get()
        cursorObj.execute('INSERT INTO cuenta VALUES (?, ?, ?)', (saldo,tarjeta,nombreC))
        con.commit()

    def producto(self):
        nombre=self.entry14.get()
        precio=self.entry15.get()
        stock=self.entry16.get()
        cursorObj.execute('INSERT INTO producto VALUES (?, ?, ?)', (nombre,precio,stock))
        con.commit()

    def Bcliente(self):
        nombre=self.entry17.get()
        direccion=self.entry18.get()
        telefono=self.entry19.get()
        email=self.entry20.get()
        cursor=cursorObj.execute("select nombre,direccion,telefono,email from cliente where nombre=?", (nombre, ))
        print(nombre+str(cursor))
        fila=cursor.fetchone()
        self.entrv18.insert(0, fila[1])
```



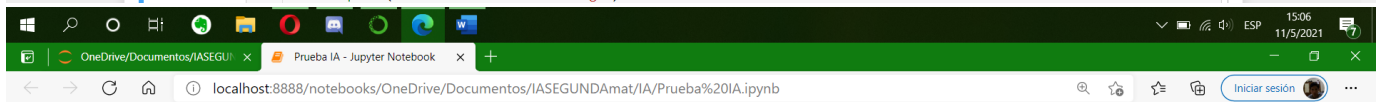
Simulación Tema: Inteligencia Artificial 1.



```
jupyter Prueba IA Last Checkpoint: hace 16 horas (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

64         else:
65             print("No existe.")
66
67     def Bproducto(self):
68         nombre=self.entry14.get()
69         precio=self.entry15.get()
70         stock=self.entry16.get()
71         cursor=cursorObj.execute("select nombre,precio,stock from producto where nombre=?", (nombre, ))
72         filal=cursor.fetchone()
73         self.entry22.insert(0, filal[1])
74         self.entry23.insert(0, filal[2])
75
76
77     def calcular(self):
78         nombre=self.entry17.get()
79         cant=self.entry25.get()
80         total=int(float(self.entry22.get()))*int(float(self.entry24.get())) #cantidad
81         self.entry25.insert(0, total)
82
83
84     def recargo(self):
85         nombre=self.entry17.get()
86         recarga=self.entry27.get()
87         cursor=cursorObj.execute("select saldo from cuenta where nombre=?", (nombre, ))
88         filal=cursor.fetchone()
89         saldoF=int(float(filal[0]))+int(float(recarga))
90
91         update = """Update cuenta set saldo = ? where nombre = ?"""
92         data = (saldoF, nombre)
93         cursorObj.execute(update, data)
94         con.commit()
95         MessageBox.showwarning("Alerta", "Se ha realizado una recarga")
96
97     print("se ha realizado una recarga")
```



```
jupyter Prueba IA Last Checkpoint: hace 16 horas (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

99
100     def comprar(self):
101         nombre=self.entry17.get()
102         cant=self.entry25.get()
103         total=int(float(self.entry22.get()))*int(float(self.entry24.get())) #cantidad
104
105         cursor=cursorObj.execute("select saldo from cuenta where nombre=?", (nombre, ))
106         filal=cursor.fetchone()
107         if filal[0]>total:
108             if self.entry16.get()>cant:
109                 print(filal)
110                 self.entry26.insert(0, filal[0]-total)
111                 act=self.entry26.get()
112                 update = """Update cuenta set saldo = ? where nombre = ?"""
113                 data = (act, nombre)
114                 cursor.execute(update, data)
115                 con.commit()
116             else:
117                 MessageBox.showwarning("Alerta", "No hay en stock por cuarentena.")
118                 print("NO hay en stock por cuarentena.")
119         else:
120             MessageBox.showwarning("Alerta", "No tiene saldo che pobre.")
121             print("No tiene saldo che pobre.")
122
123
124
125     def __init__(self):
126         self.ventanal=tk.Tk()
127         self.ventanal.title("PRUEBA ")
128         self.cuaderno1 = ttk.Notebook(self.ventanal)
129
130         self.paginal = ttk.Frame(self.cuaderno1)
131         self.cuaderno1.add(self.paginal, text="Persona")
132         self.label1=ttk.Label(self.paginal, text="Nombre")
```