



Universidad UTE



Carrera Tecnología Superior Desarrollo de Software

Proyecto final

Eventos sociales y su organización

Event Manage

“Organiza con propósito, impacta con pasión”

Informe Final

Calderón Diego

31 Julio 2025



Índice

Introducción	1
Objetivo General	1
Desarrollar un sistema de gestión de eventos, que permita a los usuarios administrar eventos tomando en cuenta el registro, la logística y la gestión de invitados y recursos	1
Objetivos específicos	1
Descripción de Funciones del Módulo de seguridad	2
<i>Desglose del código</i>	<i>2</i>
Descripción de Funciones del Módulo de ventas	4
<i>Desglose del código</i>	<i>4</i>
Descripción de Funciones del Módulo de Reportes y Resultados	7
<i>Desglose del código</i>	<i>7</i>
<i>Nuevas funciones y librerías empleadas en el programa.</i>	<i>11</i>
<i>Color en los módulos.....</i>	<i>13</i>
Manual de usuario módulo de reportes y resultados	14
Paso 6 Seleccionar Capacidad del Evento.....	16
Referencias bibliográficas	18
Anexos	19



Introducción

Event Manage es una solución integral diseñada para simplificar la planificación y gestión de eventos. El programa está estructurado en tres módulos principales: un módulo de seguridad que garantiza el acceso solo a usuarios autorizados, un módulo de ventas que ofrece una interfaz intuitiva para personalizar eventos, desde la selección del tipo de evento hasta la gestión de asistentes y la adquisición de recursos adicionales, y un módulo de reportes que consolida toda la información para generar e imprimir una factura detallada con los costos, el IVA y el total final. Con Event Manage, los usuarios pueden transformar un proceso complejo en una experiencia fluida y organizada.

Objetivo General

Desarrollar un sistema de gestión de eventos, que permita a los usuarios administrar eventos tomando en cuenta el registro, la logística y la gestión de invitados y recursos

Objetivos específicos

- Implementar un sistema de autenticación con una validación de 3 intentos, garantizando que solo quien cuente con las credenciales correctas pueda ingresar al programa.
- Crear una interfaz intuitiva que permita al usuario seleccionar un tipo de evento, gestionando la cantidad de asistentes y los recursos adicionales que requiera, generando una cotización precisa previo a la compra.

- Elaborar un proceso que recolecte la información del evento, calcule los costos y datos del usuario y genere una factura detallada, incluyendo la aplicación del IVA y el monto total.

Descripción de Funciones del Módulo de seguridad

Tabla 1

Desglose del código

<p>Validar_clave</p> <p>Comprueba si el usuario y la contraseña son correctos. Si no lo son, aumenta un contador de intentos y cierra la ventana después de 3 fallos.</p>	<pre>def Validar_clave(ingresar_clave, ingresar_usuario): global intentos usuario_ingresado = ingresar_usuario.get() clave_ingresada = ingresar_clave.get() if clave_ingresada == clave_correcta and usuario_ingresado == usuario_correcto: print("Login OK") respuesta_validacion.config(text="Acceso con éxito") ventana.after(400, abrir_ventas_y_cerrar_login) else: intentos += 1 if intentos >= 3: respuesta_validacion.config(text="Superó el límite de intentos") ventana.after(500, ventana.destroy()) else: respuesta_validacion.config(text=f"Usuario o contraseña incorrecta. Intentos restantes: {3 - intentos}")</pre> <p><i>Figura 1. Función para validar clave.</i></p>
<p>Abrir_ventana_cerrar_login</p> <p>Cierra la ventana actual de login y luego abre una nueva ventana, probablemente un módulo de ventas, para continuar con el programa.</p>	<pre>def abrir_ventas_y_cerrar_login(): ventana.destroy() from modulo_ventas import ventas ventas()</pre> <p><i>Figura 2. Función para cerrar la ventana y abrir el módulo de ventas.</i></p>

Salir_del_programa

Muestra un cuadro de diálogo de confirmación para preguntar al usuario si realmente quiere cerrar la aplicación, terminando la ejecución si la respuesta es "sí".

```
def Salir_del_programa():
    respuesta = messagebox.askquestion(
        title="¿Estás seguro/a de que qu
        message="Para finalizar presione
    )
    if respuesta == "yes":
        ventana.destroy()
```

Figura 3. Función para salir del programa.

Ingreso

Ocultar los botones de bienvenida, muestra los campos de entrada para usuario y contraseña, y crea nuevos botones para "Ingresar" y "Salir".

```
def Ingreso():
    global boton_iniciar_sesion, salir_programa,
    boton_iniciar_sesion.place_forget()
    salir_programa.place_forget()
    respuesta.config(text="Inicie sesión para acc
    respuesta.place(relx=0.42, rely=0.28, anchor=
    tk.Label(ventana, text="Ingrese usuario:", fo
    ingreso_usuario = tk.Entry(ventana, width="36
    ingreso_usuario.place(relx=0.75, rely=0.42, a
    tk.Label(ventana, text="Ingrese contraseña:",
    ingreso_clave = tk.Entry(ventana, show="*", v
    ingreso_clave.place(relx=0.75, rely=0.50, and
    ingreso_clave_boton = tk.Button(ventana, text
    ingreso_clave_boton.place(relx=0.89, rely=0.7
    tk.Button(ventana, text="Salir", font=("monot
```

Figura 4. Función para ingresar al programa.

Login

Crea la ventana principal, configura su apariencia con imágenes y etiquetas, y muestra los botones iniciales de "Iniciar sesión" y "Salir" para comenzar la interacción del usuario.

```
def Login():
    global ventana, respuesta, respuesta_validacion,
    ventana = tk.Tk()
    ventana.title("EVENT MANAGE- Gestiona tus eventos")
    ventana.geometry("625x625")
    ventana.config(bg="#f1ebfe")
    marco_login_img = Image.open("bienvenida_marco.png")
    marco_login_img = ImageTk.PhotoImage(marco_login_img)
    marco_bienvenida = tk.Label(ventana, image=marco_login_img)
    marco_bienvenida.image = marco_login_img # guardar imagen
    marco_bienvenida.place(relx=0.01, rely=0.02, anchor="nw")
    respuesta = tk.Label(ventana, text=";Bienvenido/a")
    respuesta.place(relx=0.76, rely=0.38, anchor="se")
    respuesta_validacion = tk.Label(ventana, text="")
    respuesta_validacion.place(relx=0.56, rely=0.61, anchor="center")
    boton_iniciar_sesion = tk.Button(ventana, text="Iniciar sesión")
    boton_iniciar_sesion.place(relx=0.60, rely=0.65, anchor="center")
    salir_programa = tk.Button(ventana, text="Salir")
    salir_programa.place(relx=0.55, rely=0.75, anchor="center")
    ventana.mainloop()

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

Figura 5. Función para abrir la ventana principal.

Descripción de Funciones del Módulo de ventas

Tabla 2

Desglose del código

Crear_eventos

crea una ventana secundaria para gestionar eventos. Muestra los eventos disponibles en cuatro botones para seleccionar tipos de eventos (Galas, Ceremonias, Recepciones y Exposiciones), todos vinculados a la función Capacidad_evento.

```
eventos_disponibles=tk.Label(ventana_eventos, text="Lista de eventos disponibles")
eventos_disponibles.place(relx=0.5, rely=0.26, anchor="center")
boton_gala=tk.Button(ventana_eventos, text="Galas", font=(font_size))
boton_gala.place(relx=0.5, rely=0.36, anchor="center")
boton_ceremonia=tk.Button(ventana_eventos, text="Ceremonias", font=(font_size))
boton_ceremonia.place(relx=0.5, rely=0.46, anchor="center")
boton_recepcion=tk.Button(ventana_eventos, text="Recepciones", font=(font_size))
boton_recepcion.place(relx=0.5, rely=0.56, anchor="center")
boton_exposicion=tk.Button(ventana_eventos, text="Exposiciones", font=(font_size))
boton_exposicion.place(relx=0.5, rely=0.66, anchor="center")
def Cerrar_eventos():
    boton_salir_eventos=tk.Button(ventana_eventos, text="Salir", font=(font_size))
    boton_salir_eventos.place(relx=0.8, rely=0.86, anchor="center")
```

Figura 6. Función para abrir la selección de eventos.

Capacidad_evento

Ocultar los elementos de la interfaz previa y mostrar nuevos elementos para gestionar la cantidad de invitados en 3 opciones 50, 100 o 200 personas, cada una vinculada a la función seleccionar_capacidad con su valor correspondiente.

```
aforo_eventos=tk.Label(ventana_eventos, text="Tamaño de eventos disponible",
aforo_eventos.place(relx=0.49, rely=0.29, anchor="center")
boton_cincuenta=tk.Button(ventana_eventos, text="50 Personas", font=("monospace", 12))
boton_cincuenta.place(relx=0.5, rely=0.40, anchor="center")
boton_cien=tk.Button(ventana_eventos, text="100 Personas", font=("monospace", 12))
boton_cien.place(relx=0.5, rely=0.50, anchor="center")
boton_doscientos=tk.Button(ventana_eventos, text="200 Personas", font=("monospace", 12))
boton_doscientos.place(relx=0.5, rely=0.60, anchor="center")
boton_regresar_eventos=tk.Button(ventana_eventos, text="Regresar", font=("monospace", 12))
boton_regresar_eventos.place(relx=0.8, rely=0.76, anchor="center")
```

Figura 7. Función para escoger la capacidad del evento.

Recursos_disponibles

Actualizar la ventana de para permitir la selección de recursos y servicios para un evento en forma de un menú de opciones.

```
recursos = ["Staff", "Equipo técnico", "Mobiliario", "Entretenimiento", "Catering"]
recursos_eventos = tk.Label(ventana_eventos, text="Recursos y servicios disponibles",
recursos_eventos.place(relx=0.49, rely=0.29, anchor="center")

for i, recurso in enumerate(recursos):
    var = tk.BooleanVar()
    chk = tk.Checkbutton(ventana_eventos, text=recurso, font=("monospace", 12),
                        variable=var, bg="#f1ebfe", anchor="w", padx=10)
    chk.place(relx=0.22, rely=0.35 + i * 0.08, anchor="w")
    variables_recursos.append(var)
```

Figura 8. Función para seleccionar los recursos.

Presupuesto

Calcular y mostrar el presupuesto estimado para un evento. Luego, abrir una ventana emergente con el presupuesto estimado en dólares y la cantidad de personas para este valor y especificar que el precio presupuestado no incluye IVA.

```
def Presupuesto():
    total = calcular_presupuesto(variables_recursos, recursos, capacidad_seleccionada)
    messagebox.showinfo("Presupuesto estimado", f"Su presupuesto estimado es: ${int(total)}")
```

Figura 9. Función para mostrar presupuesto.

Comprar

Crea una ventana secundaria para realizar el pago de un evento. Muestra celdas para ingresar información del usuario y su método de pago. Incluye una función `validar_datos` que supervisa que los datos para el nombre sean solo letras y tarjeta, teléfono y clave sean solo números con una longitud exacta.

```
tk.Label(ventana_pago, text="Por favor ingrese sus datos", font=("monospace", 12)).pack()
tk.Label(ventana_pago, text="Nombre y Apellido:", font=("monospace", 12)).pack()
entry_nombre = tk.Entry(ventana_pago)
entry_nombre.pack()
tk.Label(ventana_pago, text="Número de Tarjeta (16 dígitos):", font=("monospace", 12)).pack()
entry_tarjeta = tk.Entry(ventana_pago)
entry_tarjeta.pack()
tk.Label(ventana_pago, text="Clave de seguridad (3 dígitos):", font=("monospace", 12)).pack()
entry_clave = tk.Entry(ventana_pago, show="*")
entry_clave.pack()
tk.Label(ventana_pago, text="Teléfono (10 dígitos):", font=("monospace", 12)).pack()
entry_telefono = tk.Entry(ventana_pago)
entry_telefono.pack()
tk.Label(ventana_pago, text="Dirección:", font=("monospace", 12)).pack()
entry_direccion = tk.Entry(ventana_pago)
entry_direccion.pack()

def validar_datos():
    nombre = entry_nombre.get().strip()
    tarjeta = entry_tarjeta.get().strip()
    clave = entry_clave.get().strip()
    telefono = entry_telefono.get().strip()
    direccion = entry_direccion.get().strip()

    if not (nombre.replace(" ", "").isalpha() and
            tarjeta.isdigit() and len(tarjeta) == 16 and
            clave.isdigit() and len(clave) == 3 and
            telefono.isdigit() and len(telefono) == 10):
        messagebox.showerror("Hay un error en sus datos, por favor revise.")
        return
    ventana_pago.destroy()
    total_presupuesto = calcular_presupuesto(variables_recursos)
    mostrar_factura(nombre, telefono, direccion, total_presupuesto)
```

Figura 10. Función para proceder con el ingreso de datos para la compra y llamar al módulo de reportes

Obtener_precios

Esta función devuelve un diccionario con una lista fija de categorías y sus precios asociados a los componentes Staff, Equipo técnico, mobiliario entretenimiento y Catering.

```
def obtener_precios():
    return {
        "Staff": 1000,
        "Equipo técnico": 800,
        "Mobiliario": 5300,
        "Entretenimiento": 500,
        "Catering": 3200
    }
```

Figura 11. Función para guardar los costos de los recursos en una librería.

Volver_login

Esta función cierra la ventana actual y permite al usuario retornar

```
from modulo_seguridad import Login
```

Figura 12. Importar login.

a el módulo de seguridad,
específicamente a la ventana de
Login en el módulo de seguridad.

```
def Volver_Login():
    ventana_ventas.destroy()
    Login()
```

Figura 13. Función para regresar al login.

Descripción de Funciones del Módulo de Reportes y Resultados

Tabla 3

Desglose del código

Librerías utilizadas:

para la interfaz gráfica “tkinter”
, para manejo de archivos “os”,
administrar la fecha “datetime”,
para crear PDFs “reportlab”,
para capturar el diseño de la
factura como imagen “mss”.

```
import tkinter as tk
from tkinter import messagebox
import os
from datetime import datetime
from reportlab.pdfgen import canvas
from reportlab.lib.pagesizes import A4
from reportlab.lib.units import mm
from reportlab.lib.utils import ImageReader
import mss
```

Figura 14. Bibliotecas importadas.

Función capturar pantalla:

Usa la librería mss para tomar
una captura de pantalla y guarda
la imagen en la variable
“archivo” que por defecto crea
una imagen llamada
"captura.png".

```
def capturar_pantalla(archivo="captura.png"):
    with mss.mss() as sct:
        sct.shot(output=archivo)
```

Figura 15. Función para capturar pantalla en imagen.

Función obtener_precios

Devuelve un diccionario con los precios base de cada recurso o servicio mismos que sirve para calcular el presupuesto.

```
def obtener_precios():
    return {
        "Staff": 1000,
        "Equipo técnico": 800,
        "Mobiliario": 5300,
        "Entretenimiento": 500,
        "Catering": 3200
    }
```

Figura 16. Diccionario de precios

Función obtener_factor

Asigna el precio correspondiente según la capacidad del evento. Los presupuestos varían según el tamaño del evento siendo para 200 personas el precio completo, para 100 la mitad y para 50 una cuarta parte.

```
def obtener_factor(capacidad_seleccionada):
    if capacidad_seleccionada == 200:
        return 1.0
    elif capacidad_seleccionada == 100:
        return 0.5
    elif capacidad_seleccionada == 50:
        return 0.25
    else:
        return 1.0
```

Figura 17. Función para ajustar precios al número de personas de los eventos.

Función calcular_presupuesto

Recibe la información de variables_recursos seleccionados en el módulo de ventas al cual este módulo es llamado que indican si un recurso está seleccionado o no para la compra. Suma el precio de cada recurso seleccionado,

```
def calcular_presupuesto(variables_recursos, recursos, capacidad_seleccionada):
    precios = obtener_precios()
    factor = obtener_factor(capacidad_seleccionada)
    total = 0
    for i, var in enumerate(variables_recursos):
        if var.get():
            recurso = recursos[i]
            total += precios[recurso]
    return total * factor
```

Figura 18. Función para calcular presupuesto.

aplica el factor correspondiente y retorna el presupuesto total ajustado.

Función generar_factura_pdf

Permite generar un archivo PDF con la factura. Usa la librería reportlab para colocar texto y números en las posiciones mostradas en la ventana. Ordena los datos del encabezado, datos cliente, lista de servicios seleccionados con sus precios, subtotal, IVA y total y final mente Guarda el archivo PDF con el nombre indicado.

```
def datos_factura(texto, tamaño=14, negrita=False, y_espacio=18):
    nonlocal y
    if negrita:
        c.setFont("Helvetica-Bold", tamaño)
    else:
        c.setFont("Helvetica", tamaño)
    c.drawString(x_margen, y, texto)
    y -= y_espacio
factor = obtener_factor(capacidad_seleccionada)
precios = obtener_precios()
datos_factura("Event Manage", tamaño=12)
datos_factura("Optimiza con pro", tamaño=12)
datos_factura(f"Nombre: {nombre}", tamaño=12)
datos_factura(f"Teléfono: {telefono}", tamaño=12)
datos_factura(f"Dirección: {direccion}", tamaño=12)
datos_factura(f"Fecha: {fecha}", tamaño=12)
datos_factura("", y_espacio=12)
datos_factura("Detalle de servicios", tamaño=14, negrita=True, y_espacio=12)
for i, var in enumerate(variables_recursos):
    if var.get():
        recurso = recursos[i]
        costo = precios[recurso] * factor
        c.setFont("Helvetica", 12)
        c.drawString(x_margen, y, recurso)
        c.drawRightString(ancho - x_margen, y, f"${costo:.2f}")
        y -= 16
y -= 10
iva = total_presupuesto * 0.15
total_con_iva = total_presupuesto + iva
c.setFont("Helvetica", 12)
c.drawRightString(ancho - x_margen, y, f"Subtotal: ${total_presupuesto:.2f}")
y -= 16
c.drawRightString(ancho - x_margen, y, f"IVA (15%): ${iva:.2f}")
y -= 16
c.setFont("Helvetica-Bold", 14)
c.drawRightString(ancho - x_margen, y, f"Total: ${total_con_iva:.2f}")
c.save()
```

Figura 19. Función para generar información en el formato que se muestra la factura.

Función factura_de_captura

Usando la librería mss toma una captura de pantalla. Y exporta la imagen dentro de un PDF, elimina la imagen temporal para no dejar basura.

```
def factura_de_captura(nombre_pdf="factura_visual.pdf", nombre_captura="captura_factura.png"):
    # Captura la pantalla
    capturar_pantalla(nombre_captura)
    c = canvas.Canvas(nombre_pdf, pagesize=A4)
    imagen = ImageReader(nombre_captura)
    c.drawImage(imagen, 50, 200, width=500, preserveAspectRatio=True)
    c.save()
    # Elimina la imagen temporal
    if os.path.exists(nombre_captura):
        os.remove(nombre_captura)
```

Figura 20. Función para capturar pantalla de la ventana de facturación en imagen y transformarlo a pdf.

Función generar_nombre_pdf

Asigna un nombre único para el PDF con fecha y hora. Crea una ventana emergente con la ruta completa de guardado.

```
def nombre_factura():
    ahora = datetime.now().strftime("%Y%m%d_%H%M%S")
    home = os.path.expanduser("~")
    nombre_pdf = f"factura_{ahora}.pdf"
    ruta_completa = os.path.join(home, nombre_pdf)
    return ruta_completa
```

Figura 21. Función para nombrar el pdf.

Función mostrar_factura

Usando la función “Toplevel” crea una ventana con la estructura de la factura. Muestra los datos del cliente, fecha actual, y detalle de servicios seleccionados con sus precios así como el subtotal, IVA, total. El botón Imprimir vinculado a las otras funciones genera y guarda el PDF.

```
def mostrar_factura(nombre, telefono, direccion, total_presupuesto):
    factor = obtener_factor(capacidad_seleccionada)
    factura = tk.Toplevel(ventana_eventos)
    factura.title("Factura")
    factura.geometry("500x600")
    factura.config(bg="white")
    tk.Label(factura, text="Event Manage", font=("Arial", 20), bg="white").pack()
    tk.Label(factura, text="Optimiza con propósito, impacta con acciones", font=("Arial", 12), bg="white").pack()
    tk.Label(factura, text=f"Nombre: {nombre}", bg="white", font=("Arial", 12)).pack()
    tk.Label(factura, text=f"Teléfono: {telefono}", bg="white", font=("Arial", 12)).pack()
    tk.Label(factura, text=f"Dirección: {direccion}", bg="white", font=("Arial", 12)).pack()

    # Fecha actual dinámicamente al abrir la ventana
    from datetime import datetime
    fecha_actual = datetime.now().strftime("%Y-%m-%d")
    tk.Label(factura, text=f"Fecha: {fecha_actual}", bg="white", font=("Arial", 12)).pack()
    tk.Label(factura, text="Detalle de servicios", font=("Arial", 12), bg="white").pack()
    frame_detalle = tk.Frame(factura, bg="white")
    frame_detalle.pack()
    precios = obtener_precios()
    row = 0
    for i, var in enumerate(variables_recursos):
        if var.get():
            recurso = recursos[i]
            costo = precios[recurso] * factor
            tk.Label(frame_detalle, text=recurso, bg="white", font=("Arial", 12)).pack()
            tk.Label(frame_detalle, text=f"${costo:.2f}", bg="white", font=("Arial", 12)).pack()
            row += 1

    iva = total_presupuesto * 0.15
    total_con_iva = total_presupuesto + iva
    tk.Label(factura, text=f"Subtotal: ${total_presupuesto:.2f}", bg="white", font=("Arial", 12)).pack()
    tk.Label(factura, text=f"IVA (15%): ${iva:.2f}", bg="white", font=("Arial", 12)).pack()
    tk.Label(factura, text=f"Total: ${total_con_iva:.2f}", font=("Arial", 12), bg="white").pack()

    def guardar():
        nombre_pdf = nombre_factura()
        factura_pdf(nombre, telefono, direccion, total_presupuesto, nombre_pdf)
        messagebox.showinfo("PDF generado", f"Factura guardada en {nombre_pdf}")
    imprimir = tk.Button(factura, text="Imprimir", font=("Arial", 12), bg="white")
    imprimir.place(relx=0.8, rely=0.9, anchor="center")
```

Figura 22. Función para mostrar factura y botón de imprimir en una ventana.

Tabla 3

Nuevas funciones y librerías empleadas en el programa.

Librerías	Descripción
reportlab.pdfgen.canvas	Es un tipo de pincel para crear gráficos y texto directamente en un documento PDF.
reportlab.lib.pagesizes	Da las medidas exactas de formatos de papel, como A4.
reportlab.lib.units	Permite usar diferentes unidades de medida para que todo quede en el lugar correcto.
reportlab.lib.utils	Biblioteca para manejar imágenes.
mss	Permite capturar imágenes de pantalla.
Funciones	
global	Permite a una función acceder y modificar variables definidas fuera de su entorno local, las variables globales se pueden llamar sin importar en que parte del código se posicionen.
.messagebox.askquestion()	Muestra un cuadro de diálogo con una pregunta y dos botones de respuesta.
.destroy()	Cierra la ventana que se llama. Si se llama en la ventana principal, cierra toda la aplicación.
.place_forget()	Hace que un widget desaparezca de la ventana sin eliminarlo del código. El widget sigue existiendo en memoria, pero ya no se muestra en la interfaz.
.winfo_children()	Método de Tkinter que devuelve una lista con todos los widgets hijos directos de un widget padre dado, útil para ocultar

	elementos de una ventana y retornar sin cerrarla.
.isalpha()	Método de cadenas en Python que devuelve True si todos los caracteres de la cadena son letras.
enumerate	Función Python que permite recorrer una secuencia (lista, tupla, etc.) y obtener simultáneamente el índice y el valor de cada elemento.
BooleanVar	Clase de Tkinter que crea una variable especial que guarda valores booleanos y que puede vincularse a widgets para controlar estados.
Checkbutton	Widget de Tkinter que representa una casilla de verificación para marcar o desmarcar una opción.
Showinfo	Función de Tkinter muestra un cuadro de diálogo informativo con un título y mensaje, para notificar al usuario sin interrumpir el programa.
.strftime()	Es parte del módulo datetime. Formatear una fecha y hora, dándoles el estilo que se necesite.
.mss()	Pertenecen a la librería mss. Permite tomar una fotografía de lo que se ve en la pantalla.
.shot()	Pertenecen a la librería mss. Captura la imagen de la pantalla en ese preciso momento.
.setFont()	Pertenecen a la librería ReportLab. El selector de fuentes y su configuración.
.drawString()	Pertenecen a la librería ReportLab. Escribe un texto en un lugar

	específico del PDF.
.drawRightString()	Pertenecen a la librería ReportLab. Alinea el texto hacia la derecha en un punto determinado.
.save()	Pertenecen a la librería ReportLab. Guarda todos los cambios.
.drawImage()	Pertenecen a la librería ReportLab. Pegar una imagen en un lugar específico de tu documento.
.exists()	Es parte del módulo os. Revisa si un archivo o carpeta existe.
.remove()	Es parte del módulo os. Borra un archivo de forma definitiva
.expanduser()	Es parte del módulo os. Ayuda a encontrar rápidamente el directorio principal.
.join()	Parte del módulo os. Forma una dirección de archivo completa.
.showinfo()	Es parte del módulo tkinter. Muestra un mensaje simple en la pantalla para darle información al usuario.

Tabla 4

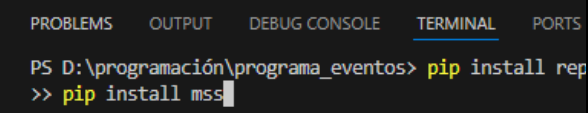

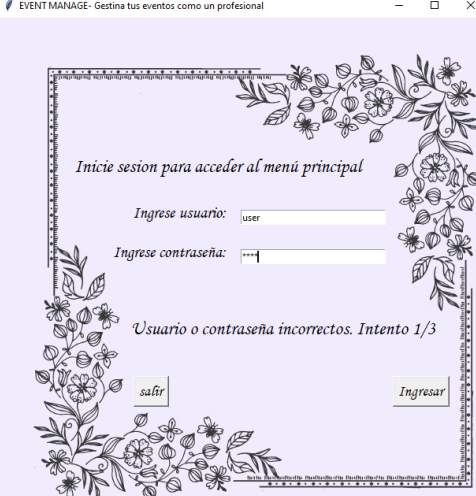
Color en los módulos

Color	Por qué y donde se usó
#F1EBFE Lavanda	<p>color suave que transmite tranquilidad, sofisticación y un toque de lujo, ideal para eventos de alta gama como bodas, galas o ceremonias formales.</p> <p>Se utilizó en casi todas las ventanas como color principal del fondo, a excepción de las ventanas de mensaje y la factura. También se empleó en algunos botones para que contraste con el otro color complementario del programa.</p>
#A5D3F5 Azul cielo pastel	<p>El azul es universalmente asociado con la fiabilidad, la seguridad y la profesionalidad.</p> <p>Se implementó en el módulo de ventas, para contrastar con los colores de las imágenes y el tono principal del fondo. También se utilizó en algunos botones, para mantener un ambiente sobrio y agradable para el usuario.</p>

Manual de usuario módulo de reportes y resultados

Tabla 5

Manual de usuario

Pasos	Funcionamiento
Paso 1: Instalar Requisitos Instalar desde la terminal Python las librerías tkinter, pillow, reportlab y mss usando el siguiente comando: pip install reportlab mss pillow tkinter	 <p>Figura 23. Ejemplo terminal de Python y comando de instalación.</p>
Paso 2 ingresar al login En esta ventana de clic en el boton iniciar sesión para acceder a la ventana de login. O en Salir para cerrar el programa.	 <p>Figura 24. Ventana de bienvenida.</p>
Paso 3 Ingresar al Login En la ventana de login debes ingresar el (Usuario predeterminado = Usuario1) y para la contraseña debe ingresar (la contraseña predeterminada = 1234). Para continuar de clic en ingresar. En caso de ingresos fallidos da un mensaje de error.	 <p>Figura 25. Ventana de login.</p>

Paso 4 crea tu evento

Crear eventos haciendo clic en el botón correspondiente.

Salir del programa usando el botón Salir.

Regresar al login con el botón Regresar, si deseas cambiar de usuario.

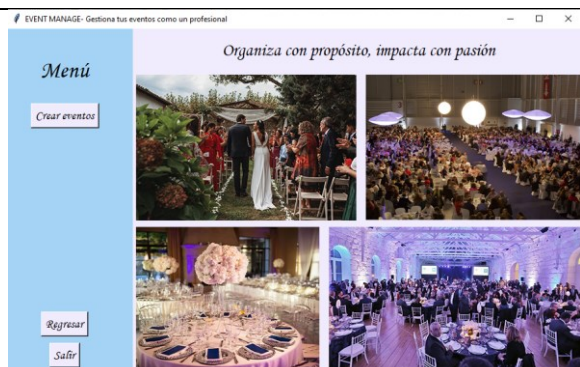


Figura 26. Ventana principal de eventos.

Paso 5 selecciona tu evento

Al crear un evento, se abrirá una nueva ventana con una lista de tipos de eventos disponibles (Galas, Ceremonias, Recepciones, Exposiciones).

Selecciona uno haciendo clic en el botón correspondiente para continuar.



Figura 27. Ventana selección de eventos.

Paso 6 Seleccionar Capacidad del Evento

Elige el tamaño del evento entre las opciones de 50, 100 o 200 personas.

Al seleccionar, accederás a la siguiente pantalla para elegir recursos.



Figura 28. Ventana selección de asistentes.

Paso 7 Selección de Recursos y Servicios

Verás una lista de recursos disponibles (Staff, Equipo técnico, Mobiliario, Entretenimiento, Catering).

Marca las casillas para seleccionar los que necesites.

Puedes calcular el presupuesto estimado haciendo clic en Calcular presupuesto.

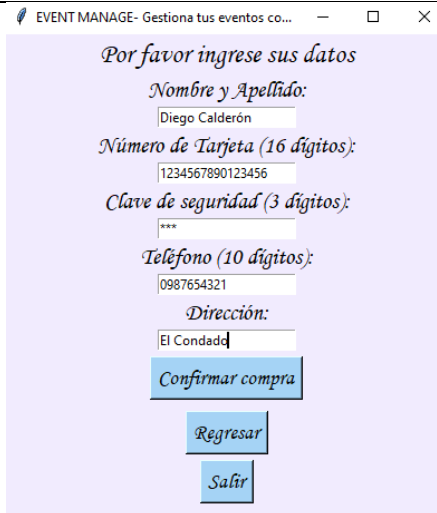
Si decides proceder a la compra, presiona el botón Comprar.



Figura 29. Ventana selección de eventos.

Paso 8: Iniciar el módulo

Este módulo de reportes y resultados es llamado desde el módulo de ventas tras ingresar los datos dando clic en la opción de confirmar compra.



EVENT MANAGE- Gestiona tus eventos co...

Por favor ingrese sus datos

Nombre y Apellido:
Diego Calderón

Número de Tarjeta (16 dígitos):
1234567890123456

Clave de seguridad (3 dígitos):

Teléfono (10 dígitos):
0987654321

Dirección:
El Condado

Confirmar compra

Regresar

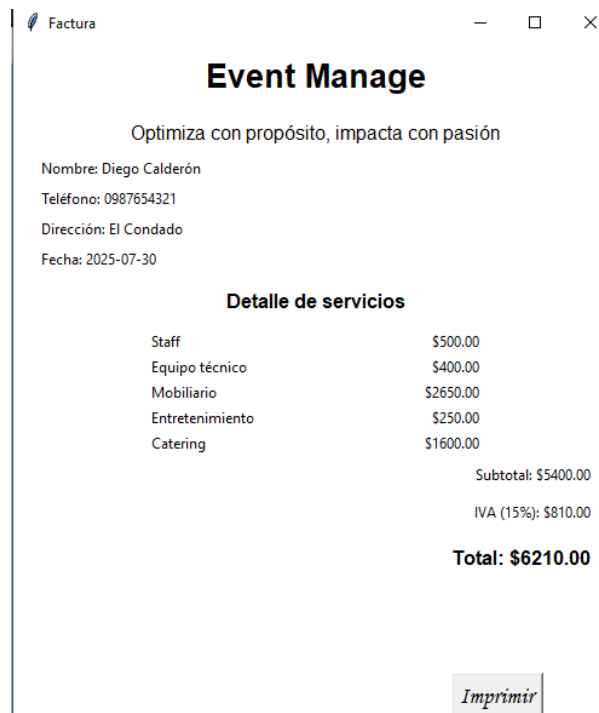
Salir

Figura 30. Llamar al módulo de ventas.

Paso 9: Generar la Factura en pdf.

Los datos y el detalle del presupuesto se mostrarán en la ventana de la factura.

Para guardar una copia digital, haz clic en el botón "Imprimir". Esto generará un archivo PDF con la factura completa. En caso de no querer una copia, dar clic en la "x" para cerrar.



Factura

Event Manage

Optimiza con propósito, impacta con pasión

Nombre: Diego Calderón
Teléfono: 0987654321
Dirección: El Condado
Fecha: 2025-07-30

Detalle de servicios

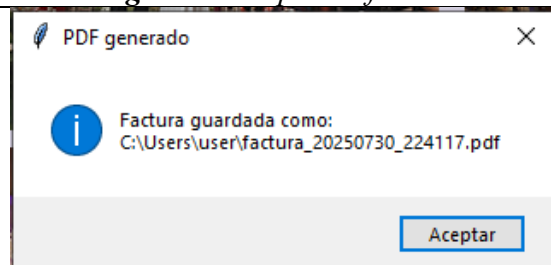
Staff	\$500.00
Equipo técnico	\$400.00
Mobiliario	\$2650.00
Entretenimiento	\$250.00
Catering	\$1600.00
Subtotal:	\$5400.00
IVA (15%):	\$810.00
Total:	\$6210.00

Imprimir

Figura 31. Imprimir factura.

Paso 10: ver ruta del pdf.

Al imprimir se mostrará la ruta donde se guarda el pdf.



PDF generado

Factura guardada como:
C:\Users\user\factura_20250730_224117.pdf

Aceptar

Figura 32. Ubicación del archivo.

Paso 5: ir a ruta del pdf.

Escribir la ruta del archivo en una ventana del explorador y dar enter.

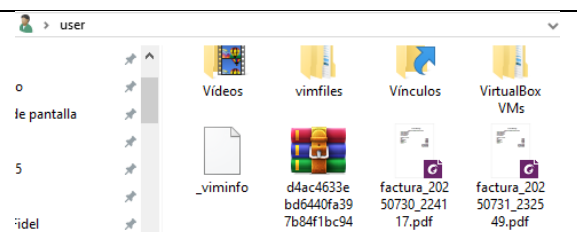


Figura 33. Ir a la ruta del archivo.

Paso 6: ir a ruta del pdf.

Revisar datos de la facturación.

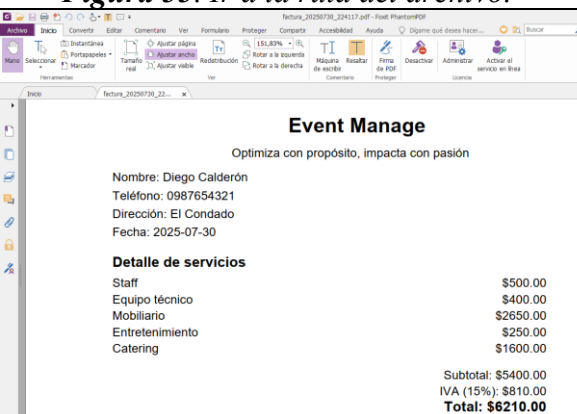


Figura 34. Abrir Factura.

Referencias bibliográficas

Calderon, D. (2025). *Event Manage: Módulo de Reportes y Resultados*. Quito, Ecuador: UTE. Trabajo académico no publicado.

Silva, Daniela Sánchez. 2024. «Uso del color para mejorar el impacto de tu evento». DanielaSanchezSilva. Recuperado (<https://www.danielasanchezsilva.com/post/uso-del-color-para-mejorar-el-impacto-de-tu-evento>).

Sweigart, A. (2019). *Automate the Boring Stuff with Python: Practical Programming for Total Beginners* (2.^a ed.). No Starch Press.

Anexos

Anexo A

Boceto inicial del programa

