UNIVERSIDAD UTE

Programación I

"Módulo de Compra y Venta de Boletos"

"Cinemax"

Desarrollador: **Fernandez Antoni Chasi Xavier**

Codificando experiencias, componiendo entretenimiento

Ciudad: Quito

Fecha: 24 de Julio del 2025

2.Tabla de Contenido

3. Un Viaje de Color: La Paleta de Cinemax

La elección de los colores en la aplicación Cinemax no es al azar; cada tono ha sido seleccionado para crear una experiencia visual inmersiva y funcional que transporte al usuario directamente a la magia del cine.

Negro Profundo (`#111111`): Evoca la oscuridad de la sala de cine, priorizando el contenido y reduciendo la fatiga visual, aportando elegancia.

Verde Azulado Brillante (`#00FFF7`): Representa la chispa y modernidad de la pantalla iluminada y las luces de neón, destacando elementos interactivos y la identidad de la marca.

Blanco Puro (`#FFFFFF`): Asegura la legibilidad de la información secundaria en contraste con el fondo oscuro.

Rojo Intenso (`#FF3B3F`): Para asientos ocupados, indica claridad y urgencia, con un toque dramático que recuerda al cine.

Verde Azulado Brillante (para asientos disponibles): Refuerza la identidad de la marca y hace la selección intuitiva y atractiva.

Dorado (`#FFD700`): En el logo "CINEMAX", añade un toque de lujo y prestigio.

Variantes de Azul y Naranja/Rojo: Usados en botones para indicar diferentes acciones (confianza, eficiencia, retroceso, advertencia) y el monto total, manteniendo la coherencia visual.

En resumen, la combinación de estos colores crea un ambiente **moderno, funcional y evocador**, guiando al usuario y resaltando la información clave para una experiencia de compra de boletos tan atractiva como la película misma.

4. Manual de Usuario: Cinemax - Compra y Venta de Boletos

4.1. Introducción al Sistema Cinemax

¡Bienvenido a Cinemax, tu plataforma para comprar boletos de cine de forma rápida y sencilla! Este manual te guiará a través de los pasos para seleccionar tu película favorita, elegir tus asientos y finalizar tu compra.

Objetivo General:

Desarrollar un sistema de gestión de compra y venta de boletos de cine intuitivo y eficiente, que permita a los usuarios seleccionar películas, funciones y asientos, así como procesar transacciones y visualizar el historial de compras, garantizando una experiencia de usuario fluida y confiable.

Objetivos Específicos:

Visualización de Cartelera: Implementar una interfaz gráfica que muestre de forma clara y atractiva la cartelera de películas disponibles, incluyendo detalles relevantes como título, género e imagen.

Gestión de Selección: Permitir a los usuarios seleccionar una película específica y elegir entre sus funciones (horarios) disponibles.

Administración de Asientos: Desarrollar un selector de asientos que muestre visualmente la disponibilidad de los mismos en tiempo real (ocupados vs. disponibles) y permita al usuario elegir y confirmar un asiento único por compra.

Procesamiento de Compras: Integrar la lógica necesaria para procesar la reserva del asiento seleccionado y registrar la compra de manera persistente.

Generación de Facturas: Crear una vista de factura que detalle los boletos adquiridos en la sesión actual, incluyendo información de la película, función, asiento y un resumen financiero con subtotal, IVA y total a pagar.

Persistencia de Datos: Asegurar que la información de los asientos reservados se guarde y se cargue correctamente entre sesiones de la aplicación, manteniendo la integridad del mapa de asientos.

4.2. Navegación Principal: Cartelera

Al iniciar la aplicación, verás la **Cartelera Principal** de Cinemax.

Películas en Cartelera: A la izquierda, se mostrará una lista de las películas disponibles con su imagen, título y género.

Panel Cinemax: A la derecha, encontrarás el logo de Cinemax y un botón para **"Ver Compras Actuales"**.

4.3. Ver Detalles de una Película

Para conocer más sobre una película:

Haz clic en el botón **"Ver más detalles"** junto a la película de tu interés en la cartelera.

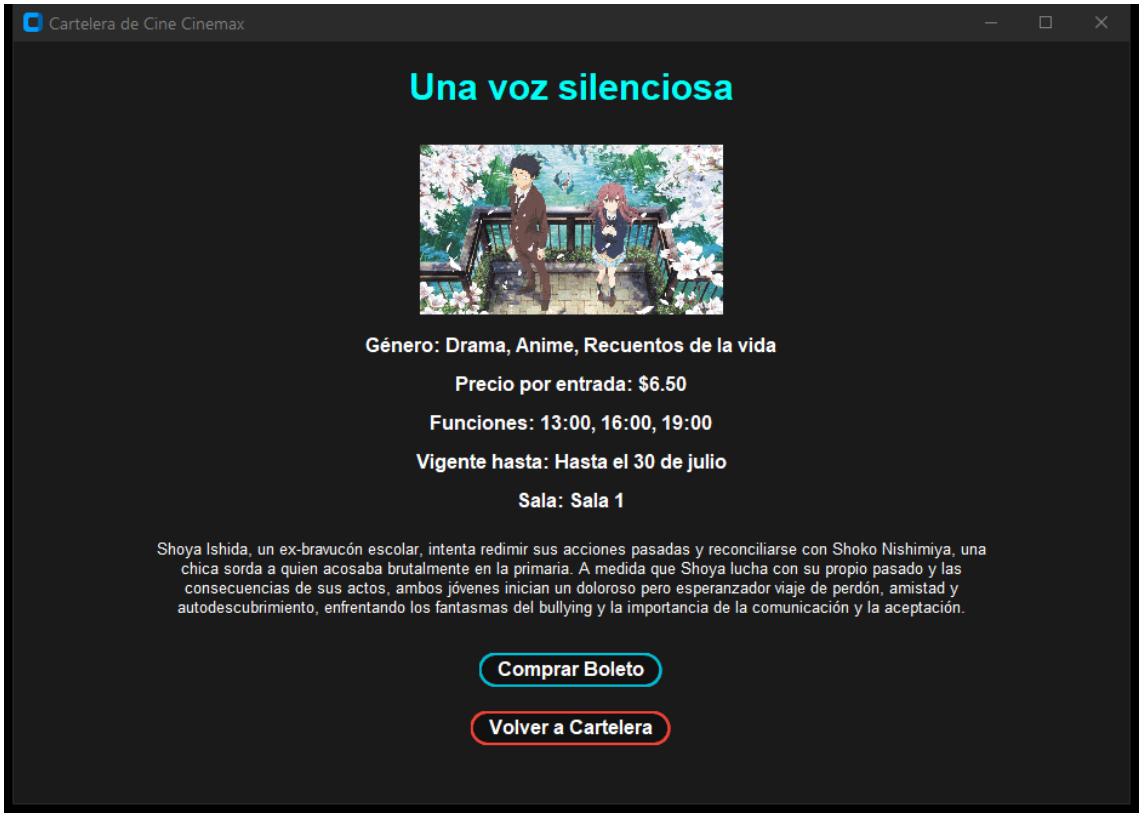
Accederás a la **Pantalla de Detalles**, donde verás:

Una imagen más grande de la película.

Título, género, precio por entrada, horarios de funciones, vigencia y sala.

Una descripción completa de la trama.

Para volver a la cartelera, presiona **"Volver a Cartelera"**.



4.4. Proceso de Compra de Boletos

4.4.1. Seleccionar Función

Desde la **Pantalla de Detalles** de una película, haz clic en **"Comprar Boleto"**.

Se te presentará la **Pantalla de Compra**, donde deberás seleccionar el horario de la función deseada en el menú desplegable.

Haz clic en **"Seleccionar Asiento"** para continuar. Si no seleccionas una función, el sistema te lo indicará.

4.4.2. Elegir Asiento

En la **Pantalla de Selección de Asiento**, verás un mapa de la sala de cine con asientos representados por botones.

Asientos Ocupados: Los asientos en color rojo (o similar) y deshabilitados ya están reservados.

Asientos Disponibles: Los asientos con el texto en color blanco (o similar) son libres.

Seleccionar un Asiento: Haz clic en el asiento que desees. Este cambiará de color para indicar que ha sido seleccionado. Puedes cambiar tu elección haciendo clic en otro asiento disponible.

Una vez seleccionado el asiento, el botón **"Confirmar Reserva"** se activará y mostrará el nombre del asiento elegido.

Haz clic en **"Confirmar Reserva"** para proceder con la compra.

Si el asiento ya fue tomado por otra persona (poco probable en una sesión simple, pero posible), recibirás una notificación.

Si la reserva es exitosa, se te informará y serás redirigido a la factura.

Para volver a la selección de función, haz clic en **"Volver a Compra"**.

4.5. Ver Compras Actuales (Previsualización de la Factura)

Después de confirmar una reserva o al hacer clic en **"Ver Compras Actuales"** desde la cartelera, accederás a la **Pantalla de Factura de Boletos**.

Aquí se listarán todos los boletos que hayas comprado en la sesión actual.

La factura incluirá: **Película, Función, Asiento, y Precio (USD)**.

Al final, verás el **Subtotal**, el **IVA (15%)** calculado, y el **Total a Pagar**.

Para regresar a la cartelera principal, usa el botón **"Volver a Cartelera"**.

5.Manual Técnico: Cinemax - Módulo de Compra y Venta

5.1. Resumen del Módulo

El sistema "Cinemax" es una aplicación de escritorio desarrollada en Python utilizando las librerías `tkinter` y `customtkinter` para la interfaz gráfica. Este módulo gestiona la visualización de la cartelera de películas, la selección de funciones, la elección de asientos y la

generación de una factura de compra. Las reservas de asientos son persistentes gracias al almacenamiento en un archivo JSON.

5.2. Estructura del Código

El código está organizado en una única clase principal, `CineApp`, que encapsula toda la lógica de la interfaz de usuario y la gestión de datos.

`CineApp` Clase Principal:

`__init__(self, ventana_inicial):` Constructor que inicializa la ventana principal de Tkinter, establece el título, el tamaño y centra la ventana. Carga las reservas existentes desde `reservas.json`.

```
class CineApp:
    def __init__(self, ventana_inicial):
        self.ventana_raiz_app = ventana_inicial
        self.ventana_raiz_app.title("Cartelera de Cine Cinemax")
        self.ventana_raiz_app.configure(bg=COLOR_FONDO_PRINCIPAL)
        self.ventana_raiz_app.geometry("920x720")
        centrar_ventana(self.ventana_raiz_app, 920, 720)

        self.pelicula_actual_seleccionada = None
        self.funcion_horario_seleccionada = None
        self.asiento_elegido_actualmente = None

        self.mapa_reservas_global = cargar_reservas_desde_json()
        self.lista_boletos_sesion = [] # Boletos comprados en la sesión actual

        self.mostrar_cartelera_principal()
```

`limpiar_contenido_ventana(self):` Método utilitario para destruir todos los widgets de la ventana, permitiendo cambiar de vista limpiamente.

`mostrar_cartelera_principal(self):` Crea y muestra la interfaz de la cartelera, incluyendo la lista de películas con imágenes y el panel lateral de Cinemax. Utiliza un `Canvas` con `Scrollbar` para permitir el desplazamiento si hay muchas películas.

`mostrar_detalles_pelicula(self, peli_elegida):` Presenta una vista detallada de la película seleccionada, con una imagen más grande y toda la información relevante.

`mostrar_interfaz_compra(self):` Muestra la interfaz para que el usuario seleccione la función (horario) de la película antes de elegir un asiento.

`navegar_a_seleccion_asiento(self):` Valida que una función haya sido seleccionada y llama al método `mostrar_selector_asientos`.

`mostrar_selector_asientos(self):` Dibuja el mapa de asientos (grid 5x5). Los asientos ocupados se muestran deshabilitados y en un color distinto. Maneja la lógica de selección de un único asiento.

`seleccionar_asiento(self, asiento_a_elegir):` Actualiza visualmente el asiento seleccionado y habilita el botón de confirmación de reserva.

`ejecutar_confirmacion_reserva(self):` Realiza la reserva del asiento seleccionado, actualiza el `mapa_reservas_global` y guarda los cambios en `reservas.json`. Añade el boleto a la `lista_boletos_sesion`.

`mostrar_factura_final(self):` Genera y muestra una tabla con los boletos comprados en la sesión actual, calculando subtotal, IVA y total.

5.3. Componentes Clave y Librerías

`tkinter`: Librería estándar de Python para interfaces gráficas de usuario.

`customtkinter` (`ctk`): Extensión de `tkinter` que proporciona widgets modernos y personalizables con temas oscuros/claros, mejorando la estética y experiencia de usuario.

`PIL` (Pillow): Usada para el procesamiento de imágenes (`Image`, `ImageTk`, `ImageFont`). Es crucial para cargar, redimensionar y mostrar las imágenes de las películas y el logo de manera eficiente y manteniendo la proporción.

```
try:
    # Carga de imagen de cartelera manteniendo proporción
    imagen_original_cartelera = Image.open(datos_peli["imagen"])
    ancho_original, alto_original = imagen_original_cartelera.size
    ancho_deseado, alto_deseado = 100, 150 # Tamaño deseado para la cartelera

    # Calcular el nuevo tamaño manteniendo la proporción
    ratio_ancho = ancho_deseado / ancho_original
    ratio_alto = alto_deseado / alto_original
    ratio = min(ratio_ancho, ratio_alto) # Usar el ratio más pequeño para que quepa

    nuevo_ancho = int(ancho_original * ratio)
    nuevo_alto = int(alto_original * ratio)

    # Redimensionar con LANCZOS para mejor calidad
    imagen_redimensionada_cartelera = imagen_original_cartelera.resize((nuevo_ancho, nuevo_alto), Image.Resampling.LANCZOS)
    imagen_cartelera_tk = ImageTk.PhotoImage(imagen_redimensionada_cartelera)

    label_img_widget = ctk.CTkLabel(marco_pelicula_individual, image=imagen_cartelera_tk, text="")
    label_img_widget.image = imagen_cartelera_tk # Mantener referencia
    label_img_widget.grid(row=0, column=0, rowspan=3, padx=10, pady=10) # Padding interno
except FileNotFoundError:
    ctk.CTkLabel(marco_pelicula_individual, text="Sin imagen", width=100, height=150, fg_color="gray", text_color="white", corner_radius=10,
                 font=FUENTE_PARRAFO_DESCRIPCION).grid(row=0, column=0, rowspan=3, padx=10, pady=10)
```

`json`: Módulo para la serialización y deserialización de datos en formato JSON. Se utiliza para persistir las reservas de asientos en un archivo `reservas.json`.

`os`: Módulo para interactuar con el sistema operativo, usado para verificar la existencia del archivo `reservas.json`.

5.4. Gestión de Datos (Persistencia)

`peliculas` (Lista de Diccionarios): Datos estáticos de las películas (título, descripción, funciones, sala, género, precio, imagen). Estos datos se cargan directamente en la aplicación.

`reservas.json`: Archivo utilizado para almacenar las reservas de asientos realizadas por los usuarios.

`cargar_reservas_desde_json()`: Lee el archivo `reservas.json` y construye un diccionario (`mapa_reservas_global`) donde la clave es una tupla `(pelicula, funcion)` y el valor es un `set` de asientos ocupados para esa función. Maneja errores de archivo dañado.

```python
def cargar_reservas_desde_json():
    """Carga las reservas desde un archivo JSON."""
    if os.path.exists(ruta_archivo_reservas):
        try:
            with open(ruta_archivo_reservas, 'r') as f_reservas:
                datos_reservas = json.load(f_reservas)
                mapa_reservas = {}
                for reserva_item in datos_reservas:
                    clave_funcion_asiento = (reserva_item["pelicula"], reserva_item["funcion"])
                    if clave_funcion_asiento not in mapa_reservas:
                        mapa_reservas[clave_funcion_asiento] = set()
                    mapa_reservas[clave_funcion_asiento].add(reserva_item["asiento"].upper())
                return mapa_reservas
        except json.JSONDecodeError:
            messagebox.showerror("Error de Carga", "El archivo de reservas está dañado o vacío. Se creará uno nuevo.")
            return {} # Retorna un diccionario vacío para empezar de nuevo
    return {} # Si el archivo no existe, retorna un diccionario vacío
```

`guardar_reservas_a_json(mapa_reservas_a_guardar)`: Serializa el `mapa_reservas_global` a una lista de diccionarios y lo escribe en `reservas.json` con formato legible.

```python
def guardar_reservas_a_json(mapa_reservas_a_guardar):
    """Guarda las reservas en un archivo JSON."""
    lista_para_json = []
    for (nombre_pelicula, horario_funcion), conjunto_asientos in mapa_reservas_a_guardar.items():
        for asiento_reservado in conjunto_asientos:
            lista_para_json.append({"pelicula": nombre_pelicula, "funcion": horario_funcion, "asiento": asiento_reservado})
    with open(ruta_archivo_reservas, 'w') as f_reservas:
        json.dump(lista_para_json, f_reservas, indent=4) # Guarda con indentación para legibilidad
```

5.5. Estilización y UX

Constantes: Se definen constantes para colores y fuentes para una gestión centralizada y fácil modificación del estilo.

```
# --- Constantes de Estilo y Colores ---
COLOR_FONDO_PRINCIPAL = "#111111"
COLOR_TEXTO_DESTACADO = "#00FFF7"  # Verde azulado brillante
COLOR_TEXTO_NORMAL = "#FFFFFF"  # Blanco puro
COLOR_ASIENTO_OCUPADO = "#FF3B3F"  # Rojo intenso para asientos ocupados
COLOR_ASIENTO_DISPONIBLE = COLOR_TEXTO_DESTACADO # Utiliza el color primario para asientos disponibles

FUENTE_BASE_APP = ("Montserrat", 14)
FUENTE_ENCABEZADO_GRANDE = ("Montserrat", 38, "bold") # Para "Cartelera", "FACTURA DE BOLETOS"
FUENTE_SUBTITULO_GRANDE = ("Montserrat", 30, "bold") # Para títulos de película en detalle, "Seleccionar Asiento"
FUENTE_LOGO_CINEMAX = ("Montserrat", 48, "bold") # Para el texto "CINEMAX" muy grande
FUENTE_TITULO_PELICULA_C = ("Montserrat", 20, "bold") # Títulos de películas en la cartelera
FUENTE_INFO_PELICULA = ("Montserrat", 16, "bold") # Género, funciones, sala, precio en detalles
FUENTE_GENERO_CARTELERA = ("Montserrat", 14, "italic") # Género en la cartelera
FUENTE_PARRAFO_DESCRIPCION = ("Montserrat", 13) # Texto de descripción largo
FUENTE_BOTON_PRINCIPAL = ("Montserrat", 16, "bold") # Botones principales
FUENTE_ENCABEZADO_TABLA = ("Montserrat", 14, "bold")
FUENTE_CONTENIDO_TABLA = ("Montserrat", 13)
FUENTE_RESUMEN_FACTURA = ("Montserrat", 20, "bold")
FUENTE_TOTAL_FACTURA = ("Montserrat", 26, "bold")
```

`centrar_ventana()`: Función utilitaria para posicionar la ventana de la aplicación en el centro de la pantalla.

`crear_boton_estilizado()`: Función de ayuda para generar botones `customtkinter` con un estilo uniforme (colores, bordes, fuente, hover).

```
def crear_boton_estilizado(master_widget, texto_boton, comando_ejecutar, color_borde_btn="#0097A7", color_hover_btn="#0097A7"):
    """Crea un botón con estilo personalizado de CustomTkinter."""
    return ctk.CTkButton(
        master_widget,
        text=texto_boton,
        fg_color="#111111",  # Fondo del botón
        text_color="#FFFFFF",  # Color del texto del botón
        border_color=color_borde_btn,
        border_width=2,
        corner_radius=15,
        font=FUENTE_BOTON_PRINCIPAL,
        hover_color=color_hover_btn,
        command=comando_ejecutar
    )
```

Redimensionamiento de Imágenes: Las imágenes se redimensionan dinámicamente utilizando `PIL` con `Image.Resampling.LANCZOS` para mantener la calidad visual en diferentes tamaños (cartelera, detalle, logo).

5.6. Escalabilidad y Mejoras Futuras

Base de Datos: Para una aplicación de producción, la gestión de reservas debería migrarse a una base de datos real (SQLite, PostgreSQL, MySQL) para manejar un mayor volumen de datos, concurrencia y relaciones más complejas (usuarios, transacciones, etc.).

Gestión de Usuarios: Implementar un sistema de autenticación de usuarios para personalizar la experiencia y rastrear compras por usuario.

Selección Múltiple de Asientos: Actualmente solo permite la selección de un asiento a la vez; se podría permitir seleccionar múltiples asientos antes de confirmar la compra.

Tickets y QR: Generación de tickets o códigos QR para las compras.

Filtrado y Búsqueda: Añadir opciones para filtrar películas por género o buscar por título en la cartelera.

Admin Panel: Un panel de administración para añadir/editar películas, funciones, precios y gestionar reservas.

5.7. Instrucciones de Ejecución

Asegúrate de tener Python 3 instalado.

Instala las librerías necesarias: `pip install customtkinter Pillow`

Coloca el archivo `.py` del código, las imágenes de las películas (ej. `Una voz silenciosa.png`, `risas.png`, etc.) y el logo de Cinemax (`logo_cinemax.png`) en la misma carpeta.

Ejecuta el script desde la terminal: `python tu_archivo_principal.py` (reemplaza `tu_archivo_principal.py` con el nombre de tu archivo).