

# Proyecto final- Centro multimedia

Mendoza Acevedo Jonathan  
Torres Anguiano Azael Arturo  
25 de noviembre de 2025

## 1 Objetivo

El alumno elaborará un centro multimedia con todos los conocimientos adquiridos durante el semestre.

## 2 Introducción

El propósito fundamental de este reporte es implementar un centro de entretenimiento para la reproducción de películas, videos, música y fotografías.

## 3 Antecedentes

VLC es un software de reproductor de archivos multimedia multiplataforma, gratuito y de código abierto, además de ser portátil e incorporar un servidor de transmisión o streaming desarrollado por el proyecto VideoLAN..<sup>1</sup>

## 4 Desarrollo

El primer paso es esencial, se debe configurar el sistema operativo:

1. Cambiar la imagen de arranque de rasbian.

```
$ sudo cp (ubicacion)/splash.png/usr/share/pliesmouth /themes/phix
```

2. Configurar la Raspberry en modo consola.

```
$ sudo raspi-config
```

3. Configurar script de arranque.

```
$ sudo nano /etc/rc.local
```

4. Instalar librerías.

```
$ pip install tk  
$ python3 -m pip install pyautogui  
$ pip install keyboard  
$ sudo apt-get install python-pygame  
$ pip install opencv-python-headless
```

---

<sup>1</sup>VideoLAN. (s. f.). Official download of VLC media player, the best Open Source player - VideoLAN. Recuperado el 25 de noviembre de 2025 <https://www.videolan.org/vlc/>

```
$ pip install python-vlc
```

Tenemos que desarrollar toda nuestra interfaz gráfica:

1. Crear ventanas y botones para seleccionar el menú.
2. Configurar wifi.
3. Leer y reproducir fotos.
4. Leer y reproducir videos.
5. Leer y reproducir música.
6. Apagar sistema.

### ADVERTENCIA

Ten mucho cuidado con la configuración de la Raspberry, se pueden quemar componentes si no se tiene conocimiento sobre el armado.

## 5 Resultados Experimentales

### Crear ventanas y botones para seleccionar el menú.

Para poder crear ventanas y botones se utiliza la librería Tkinter, es necesario leer la documentación para entender la implementación.

```
1      # Creación de los botones
2      botonNetflix = tk.Button(window, image=img_netflix,
3                               borderwidth=0, bg=colorFondo,
4                               cursor="X_cursor",
5                               command = funcion_Netflix)
6      botonAmazon = tk.Button(window, image=img_prime,
7                               borderwidth=0, bg=colorFondo,
8                               cursor="X_cursor",
9                               command = funcion_Amazon)
10     botonDisney = tk.Button(window, image=img_disney,
11                              borderwidth=0, bg=colorFondo,
12                              cursor="X_cursor",
13                              command = funcion_Disney)
14     #Creación de los label
15     labelBienvenida = Label(window, text="Ingresando al entorno",
16                              fg="#fff",      # Foreground
17                              bg=colorFondo,   # Background
18                              font=("Verdana_Bold", 60))
19     labelTitulo = Label(window, text="Centro Multimedia",
20                           fg="#fff",      # Foreground
21                           bg=colorFondo,   # Background
22                           font=("Verdana_Bold", 60))
```

Listing 1: Código parcial de la creación de botones (para evitar exponer muchas líneas repetidas) y de ventanas).

## Configurar wifi

Para configurar la red necesitamos modificar el archivo `/etc/wpa_supplicant/wpa_supplicant.conf` con `sudo`, cuando se modifique hay que reiniciar el sistema.

```
1      #Funcion que conecta a una red WiFi
2      def conectarRed(ssid, key):
3          global labelConfig
4          arch = '/etc/wpa_supplicant/wpa_supplicant.conf'
5          subprocess.call(['sudo', 'chmod', '777', arch])
6          with open(arch, 'w') as fp:
7              fp.write('ctrl_interface=DIR=/var/run/wpa_supplicant_GROUP=netdev')
8          with open(arch, 'a') as fp:
9              fp.write('\nupdate_config=1')
10         with open(arch, 'a') as fp:
11             fp.write('\ncountry=MX')
12         with open(arch, 'a') as fp:
13             fp.write('\nnetwork={}_\n'.format(str(ssid)))
14         with open(arch, 'a') as fp:
15             fp.write('\tpskey={}_\n'.format(str(key)))
16         with open(arch, 'a') as fp:
17             fp.write('\tkey_mgmt=WPA-PSK')
18         with open(arch, 'a') as fp:
19             fp.write('\n')
20         labelConfig = Label(window, text="Aplicando configuraci\u00f3n\nRequiere reiniciar",
21                             fg="#fff",      # Foreground
22                             bg=colorFondo,   # Background
23                             font=("Verdana_Bold", 60))
24         labelConfig.place(relx=0, rely=0, relwidth=1, relheight=1)
25         window.after(4000, reinicio)
```

Listing 2: Código de ejemplo que muestra la configuración del wifi.

## Leer y reproducir fotos

Utilizamos OpenCV y reproducir las imágenes automáticamente dentro de la usb `/media/artsol/imagenes/`.

```
1      #Funcion para reproducir fotos
2      def funcion_Fotos():
3          input_images_path = "/media/artsol/imagenes/"
4          files_names = os.listdir(input_images_path)
5          for file_name in files_names:
6              image_path = input_images_path + "/" + file_name
7              print(image_path)
8              image = cv2.imread(image_path)
9              if image is None:
10                 continue
11             image = cv2.resize(image, (width, height), interpolation=cv2.INTER_CUBIC)
12             cv2.namedWindow("WindowName", cv2.WINDOW_FULLSCREEN)
13             cv2.imshow("Image", image)
14             cv2.waitKey(1500)
15             cv2.destroyAllWindows()
```

Listing 3: Código de ejemplo que muestra la reproducción de imágenes.

## Leer y reproducir videos

Utilizamos la librería **VLC** para reproducir videos mediante la reproducción automática por medio de la usb.

```
1 #Funcion para reproducir video
2 def funcion_Video():
3     input_videos_path = "/home/artsol/Desktop/videos"
4     files_names = os.listdir(input_videos_path)
5     for file_name in files_names:
6         video_path = input_videos_path + "/" + file_name
7         print(video_path)
8         video = vlc.MediaPlayer(video_path)
9         if video is None:
10             continue
11         video.play()
```

Listing 4: Código de ejemplo que muestra la reproducción de video.

## Leer y reproducir música

Las siguientes funciones muestran la implementación de la reproducción automática.

```
1 # Funcion para agregar canciones de la memoria USB
2 def aadir():
3     canciones = filedialog.askopenfilenames(initialdir="/media/artsol/ARTSOL/canciones/"
4 ,title="Elige una cancion", filetypes=(("mp3", "*.mp3"), ("allfiles", "*.*")))
5     #cambiar la extension del nombre de la cancion for cancion in canciones:
6     for cancion in canciones:
7         cancion=cancion.replace("/media/artsol/ARTSOL/canciones/", "")
8         cancion=cancion.replace(".mp3", "")
9         #aadir cancion a la pantalla
10        pantalla.insert (END, cancion)
11
12        cancion= pantalla.get (ACTIVE)
13        cancion=f'{cancion}.mp3'
14        pygame.mixer.music.load(cancion)
15        pygame.mixer.music.play(loops=0)
16    #funcion para cambiar de cancion
17    def siguiente():
18        proxima = pantalla.cursorselection()
19        proxima = proxima[0]+1
20        #get title song
21        cancion= pantalla.get (proxima)
22        cancion= f'{cancion}.mp3'
23        pygame.mixer.music.load(cancion)
24        pygame.mixer.music.play(loops=0)
25        pantalla.selection_clear(0,END)
26        #activar nueva barra a la siguiente cancion
27        pantalla.activate(proxima)
28        last = None
29        pantalla.selection_set(proxima, last)
30
31    global paused
32    paused=False
33    def pause(is_paused):
34        global paused
35        paused = is_paused
36        if paused:
37            pygame.mixer.music.unpause()
38            paused=False
39        else:
40            pygame.mixer.music.pause()
```

```
39 paused=True
```

Listing 5: Código de ejemplo que muestra la reproducción de música.

## Apagar el sistema

Para apagar el sistema necesitamos invocar `'subprocess.call()'` que se encarga de forzar la salida.

```
1 #Funcion para apagar el sistema
2 def apagar():
3     subprocess.call(['shutdown', "-h", "now"])
```

Listing 6: Código de ejemplo que muestra despedida del sistema.

## 6 Conclusiones

Se cumplió el objetivo de crear un centro multimedia, todo esto fue posible gracias al conocimiento encaminado del profesor, es complejo desarrollar un sistema embebido, se comprendió todo lo que conlleva este sistema y la implementación se logró satisfactoriamente.

## 7 Referencias:

1. tkinter — Python interface to Tcl/Tk. (s. f.). Python Documentation. Recuperado de: <https://docs.python.org/3/library/tkinter.html>
2. PyAutoGUI. (2023, 24 mayo). PyPI. Recuperado de: <https://pypi.org/project/PyAutoGUI/>
3. keyboard. (2020, 23 marzo). PyPI. Recuperado de: <https://pypi.org/project/keyboard/>
4. Pygame Front Page — pygame v2.6.0 documentation. (s. f.). Recuperado de: <https://www.pygame.org/docs/>
5. opencv-python-headless. (2025, 7 julio). PyPI. Recuperado de: <https://pypi.org/project/opencv-python-headless/>

## 8 Video:

Los resultados de los experimentos estarán en un video de Youtube: <https://youtu.be/apLj4VdHGVA>

Repositorio del proyecto: <https://github.com/jonathanMA23/centro-multimedia>