Fundamentos de sistemas embebidos

Proyecto Final:
Centro Multimedia

García Camargo José Daniel

26 de noviembre de 2025

1. Objetivo

Es crear un *centro multimedia* que tenga la capacidad de reproducir imágenes, videos y música desde medios extraibles, así también como ser capaz de abrir aplicaciones de streaming y poder usarlas.

2. Lista de materiales

2.1. ¿Que es un kiosco digital?

A continuación se enlistan los componentes necesarios para realizar este proyecto:

•Raspberry Pi 4 Model B.

•Adaptador microHDMI a HDMI.

•Memoria USB.

•Pantalla smart-TV.

•Cable de alimentación.

3. Consideraciones

Al usuario se le sugiere:

•No usar cables desgastados ya que pueden presentar corto circuito.

•No sobrecargar la toma de corriente.

•Mantener alejados los componentes de materiales inflamables.

3.1. Sobre la Raspberry Pi

Según las especificaciones de la tarjeta Raspberry Pi 4 Model B, hay algunas advertencias y recomendaciones para su uso. Las mas importantes para su uso como centro multimedia son:

•Se recomienda el uso de una fuente de alimentación adecuada, mínimo de 3A.

•Se debe mantener la Raspberry Pi con una ventilación adecuada y evitar temperaturas altas (hasta 50°C) para que no sucedan daños físicos o sobrecalentamientos.

•Antes de realizar cualquier cambio-integración física, apague su dispositivo.

Si se desea revisar las especificaciones completas de la tarjeta por favor de acceder a la siguiente liga [1].

4. Funcionamiento

4.1. Conexiones físicas

En el siguiente diagrama podemos observar la que puertos de la tarjeta se usaran para poder usarse el centro multimedia. 1 - Aquí se deberá conectar el cable de alimentación 2- Aquí se deberá conectar el cable microHDMI y el extremo de HDMI a su pantalla o monitor 3 - Son las entradas de USB.

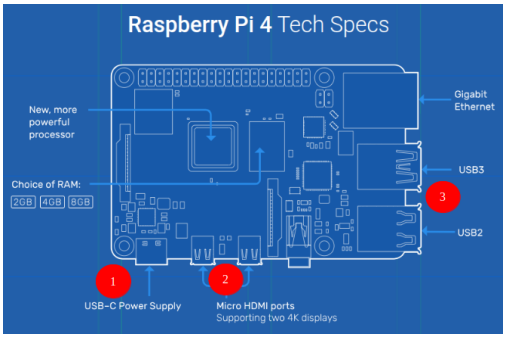

Figura 1: Menu principal del Centro Multimedia

4.2. Librerías

Para el funcionamiento adecuado del centro multimedia se usaron las siguientes librerías.

•vlc
•pygame
•sys
•threading
•os
•subprocess

La interfaz gráfica del proyecto, requirió el uso de una librería esencial llamada pygame, la cual está integrada e inicializada en todos los módulos de la siguiente manera:

```
        # Inicializa pygame
        pygame.init()
        sleep(1)

        #Creamos un hilo para los medios extraibles
        menu = MenuMediosExtraibles()
        streaming = ServiciosStreaming()
        musica = MusicaStreaming()
```

```
        red = WifiScanner()
        #red = MenuConexiones()

        # Configura el tamano de la ventana y los colores
        COLOR_FONDO = (0, 0, 0)  # Negro
        COLOR_TEXTO = (0, 0, 0)  # Blanco
        COLOR_SELECCION = (255, 64, 0)  # Rojo para resaltar la opcion seleccionada
        COLOR_OPCIONES= (255, 64, 0) #Gris para la interaccion

        # Configura la ventana de pygame
        pantalla = pygame.display.set_mode((0, 0),pygame.RESIZABLE)
        ancho_ventana, largo_ventana = pantalla.get_size()
        pygame.display.set_caption("Centro de Entretenimiento")

        fuente = pygame.font.SysFont("Arial", 60)
        fuente2 = pygame.font.SysFont("Arial",30)

        # Lista de opciones del menu
        fondo=pygame.image.load('images/fondo.jpg')
        icon_streaming=pygame.image.load('images/streaming.png')
        icon_music=pygame.image.load('images/music2.png')
        icon_usb=pygame.image.load('images/usb.png')
        icon_wifi=pygame.image.load('images/wifi3.png')
```

Para la reproducción del video simplemente se hace el llamado a la "funcion play_media"para que se reproduzca el video al ejecutar el programa.

4.3. Modulos

El centro multimedia esta compuesto de los siguientes módulos.

1. CentroMultimedia.py
2. MediosExtraibles.py
3. MenuConexiones.py
4. MenuMediosExtraibles.py
5. ServiciosStreaming.py

Cada uno de estos representa una ventana distinta del centro multimedia desde el punto de vista de la interfaz gráfica.

4.3.1. CentroMultimedia.py

El módulo principal de este proyecto es el archivo CentroMultimedia.py, que contiene el menú principal y todas las llamadas a funciones necesarias para navegar dentro del centro multimedia.

Figura 2: Mapa de conexiones (Imagen de recuperada de raspberrypi.com [1])

Para la contrucción del menú, se creó una función llamada dibujar_menu() dentro de prueba1.py en la cual se establece el fondo de la ventana, la dimesión y colores.

```
    def dibujar_menu():
       #Funcion para establecer el fondo de la ventana
       def Background_sky(fondo):
         size=pygame.transform.scale(fondo,(2048,1500))
         pantalla.blit(size, (0,0))
       textoFlechas = fuente.render(" - Moverse entre las opciones", True, COLOR_OPCIONES)
       pantalla.blit(textoFlechas,(100,400))
       textoSeleccion = fuente.render("ENTER - Seleccionar opcion", True, COLOR_OPCIONES)
       pantalla.blit(textoSeleccion,(100,450))
       y = 100  # Posicion inicial vertical para las opciones
       Background_sky(fondo)
       textoFlechas = fuente2.render(" - Moverse entre las opciones", True, COLOR_OPCIONES)
       pantalla.blit(textoFlechas,(100,960))
       textoSeleccion = fuente2.render("ENTER - Seleccionar opcion", True, COLOR_OPCIONES)
       pantalla.blit(textoSeleccion,(100,1000))
       pantalla.blit(icon_streaming,(1100,15))
       pantalla.blit(icon_music, (1100,195))
       pantalla.blit(icon_usb,(1100,375))
       pantalla.blit(icon_wifi, (1100,555))
       y = 30
       # Dibuja cada opcion
       for i, opcion in enumerate(opciones):
          if i == indice_opcion:
             texto = fuente.render(opcion, True, COLOR_SELECCION)  # Opcion seleccionada en
                 naranja
          else:
             texto = fuente.render(opcion, True, COLOR_TEXTO)  # Otras opciones en blanco
             pantalla.blit(texto, (600, y))
             y += 180  # Espaciado entre opciones

    pygame.display.flip()  # Actualiza la pantalla
```

4.3.2. ServiciosStreaming.py

El módulo de servicios de streaming está dividido en dos submódulos: video y música, donde se implemento el acceso a estas plataformas mediante llamadas a funciones que nos permite acceder al navegador web. Para poder realizar esto, se implementó llamadas a funciones (respectivamente en cada módulo) que permiten acceder al navegador web y las URL's de las plataformas.

```
        for evento in pygame.event.get():
            if evento.type == pygame.QUIT:
               self.ejecutando = False
               sys.exit()
            elif evento.type == pygame.KEYDOWN:
               if evento.key == pygame.K_DOWN:
                  self.opcion_seleccionada = (self.opcion_seleccionada + 1) % len(self.
                      opciones)
               elif evento.key == pygame.K_UP:
                  self.opcion_seleccionada = (self.opcion_seleccionada - 1) % len(self.
                      opciones)
               elif evento.key == pygame.K_RETURN:
                  if self.opciones[self.opcion_seleccionada] == "SPOTIFY":
                     self.open_service(self.opciones[self.opcion_seleccionada])
               elif self.opciones[self.opcion_seleccionada] == "DEEZER":
                  self.open_service(self.opciones[self.opcion_seleccionada])
               elif self.opciones[self.opcion_seleccionada] == "AMAZON MUSIC":
                  self.open_service(self.opciones[self.opcion_seleccionada])
              elif self.opciones[self.opcion_seleccionada] == "VOLVER":
                  self.ejecutando = False
              elif evento.key == pygame.K_ESCAPE:
```

```
                    self.ejecutando = False  # Agregar accion para el ESC

        pygame.display.flip()
```

La función más importante de ambos módulos, es la que inicia el menú de selección y donde se puede elegir entre las diferentes opciones. La interfaz gráfica nos permite movernos mediante las flechas arriba/abajo y ENTER para volver al menú principal. Ambas ventanas (Musica y Streaming) funcionan de esta misma manera.

4.3.3. MediosExtraibles.py

Dentro del módulo de medios extraíbles, se desarrollaron funciones que le permite al usuario reproducir contenido multimedia, incluyendo videos, imágenes y música, almacenados en un medio extraíble o USB. El modulo monta la usb y manda un mensaje a pantalla cuando se inserto correctamente. Todo esto funciona mediante VLC, la cual permite reproducir y gestionar el contenido multimedia almacenado en la USB.

- Fotos

```
                def analizar_contenido_images(self, ruta_montaje):
                    extensiones_imagen = ('.jpg', '.jpeg', '.png', '.gif', '.bmp')
                    image_files = []
                    try:
                        for root, dirs, files in os.walk(ruta_montaje):
                        for file in files:
                            if file.lower().endswith(extensiones_imagen):
                                ruta_completa = os.path.join(root, file)
                            if os.path.isfile(ruta_completa):
                                image_files.append(ruta_completa)
                    except Exception as e:
                        print(f"Error al buscar imagenes: {str(e)}")

                    if not image_files:
                        self.mostrar_mensaje_temporal("No se encontraron imagenes", self.
                            ROJO, 2)
                    return []

                return image_files
```

Ademas se programo una función para que el usuario pueda navegar en sus fotos:

```
                  for evento in pygame.event.get():
                      if evento.type == pygame.KEYDOWN:
                          if evento.key == pygame.K_ESCAPE:
                              corriendo = False
                              esperando = False
                          elif evento.key == pygame.K_RIGHT:
                              esperando = False  # Siguiente imagen
                          elif evento.key == pygame.K_LEFT:
                              index_actual = max(0, index_actual - 2)  # Imagen anterior
                              esperando = False
                          elif evento.key == pygame.K_SPACE:
                              esperando = False  # Tambien espacio para siguiente
```

- Videos La función para los videos funciona de la misma manera que la de fotos, solo con la diferencia que las extensiones de archivos son diferentes.

```
                  def analizar_contenido_images(self, ruta_montaje):
                      extensiones_imagen = ('.jpg', '.jpeg', '.png', '.gif', '.bmp')
                      image_files = []

                  if not image_files:
                      self.mostrar_mensaje_temporal("No se encontraron imagenes", self.
                          ROJO, 2)
```

```
                return []

            return image_files
```

- Música

```
    def analizar_contenido_musica(self, ruta_montaje):
        extensiones_musica = ('.mp3', '.wav', '.ogg', '.m4a', '.flac')
        music_files = []
        try:
            for root, dirs, files in os.walk(ruta_montaje):
                for file in files:
                    if file.lower().endswith(extensiones_musica):
                        ruta_completa = os.path.join(root, file)
                        if os.path.isfile(ruta_completa):
                            music_files.append(ruta_completa)
        except Exception as e:
            print(f"Error al buscar musica: {str(e)}")
```

4.3.4. WifiScanner.py

En el módulo de configuración de red, utilizamos el código de MenúConexiones para conectar nuestro dispositivo a una red WiFi. El proceso de conexión se inicia con un menú, donde podemos elegir entre iniciar la conexión o regresar al menú anterior.

Figura 3: Interfaz para la configuración de red

4.4. Implementación

4.4.1. Medios extraíbles

Video muestra. Medios extraibles

4.4.2. Streaming

Video muestra. Streaming
Video muestra. Streaming musica

4.4.3. Configuración de red

Video muestra. Configuracion de red

4.4.4. Código fuente

Se adjunta el enlace del repositorio donde se desarrolló el proyecto: Proyecto Final FESem.

5. Conclusiones

Este proyecto representó una experiencia sumamente enriquecedora que permitió el aprendizaje y la aplicación de los diversos conceptos que se aprendieron durante le curso, con un enfoque en el desarrollo de software. Fue un reto el poder configurar el entorno para que funcionara correctamente este proyecto pues incluyó la creación de interfaces gráficas en Python—una novedad para algunos miembros del equipo—utilizando diversas bibliotecas cuya documentación fue esencial para una correcta implementación en un entorno sin interfaz gráfica.

A lo largo del proceso, se adquirieron habilidades valiosas en la configuración de sistemas embebidos, la integración de aplicaciones conocidas como Netflix y HBO, y la gestión eficiente de los módulos con los códigos. Un aspecto fundamental fue el aprendizaje en el trabajo colaborativo, ya que el equipo aprendió a trabajar de manera remota y eficiente, utilizando herramientas de control de versiones como GitLab para gestionar los cambios durante las pruebas. Para varios integrantes, esta fue su primera experiencia trabajando en equipo dentro de esta plataforma, una habilidad que sin duda será de gran utilidad en su futuro profesional.

Referencias

[1] Raspberry Pi Holdings plc. “Raspberry pi 4 tech specs”. aspberrypi.com. Accedido el 26 de noviembre de 2025. [En línea]. [En línea]

[2] tecseris. “¿Qué es un kiosco digital y cómo funciona?” tecseris.com. Accedido el 26 de noviembre de 2025. [En línea]