# **Progetto Raspberry Pi: Sistema di Riproduzione Video con Controllo da Remoto**

## ****Introduzione****

Questo progetto ha l'obiettivo di realizzare un sistema basato su Raspberry Pi in grado di riprodurre video su un display HDMI in risposta alla pressione di un pulsante fisico. Il sistema include anche un'interfaccia web accessibile tramite un hotspot WiFi generato dal Raspberry, che permette agli utenti di caricare e selezionare video da un dispositivo mobile o PC.

## ****Architettura del sistema****

Il sistema si compone di diverse componenti principali:

1. **Sottosistema Embedded**
   * Gestisce l’input dal pulsante fisico.
   * Utilizza la libreria **PI4J** per l'interfacciamento con il GPIO del Raspberry Pi.
   * Contiene la classe EmbeddedSubsystem, responsabile della gestione del pulsante e della comunicazione con il sistema di riproduzione video.
   * Invia un segnale all'applicazione principale (PlayVideo) per avviare la riproduzione quando il pulsante viene premuto.
2. **Modulo di Riproduzione Video**
   * Implementato in PlayVideo, che utilizza un framework specifico per la gestione della riproduzione multimediale (da definire).
   * Mostra un'immagine statica sullo schermo HDMI quando non viene riprodotto alcun video.
   * Alla pressione del pulsante, avvia la riproduzione del video selezionato.
   * Quando il video termina, ritorna all’immagine statica.
3. **Web Server e Interfaccia Web**
   * Il Raspberry Pi genera un **hotspot WiFi** per permettere agli utenti di connettersi direttamente alla sua rete.
   * Un server web integrato gestisce le richieste HTTP (potrebbe essere realizzato con Flask, Node.js o un altro framework).
   * L’interfaccia web permette agli utenti di:
     + **Caricare nuovi video** nella memoria del Raspberry.
     + **Visualizzare la lista dei video disponibili**.
     + **Selezionare il video attivo**, che sarà quello riprodotto alla pressione del pulsante.
4. **Gestione della Memoria e File System**
   * I video vengono salvati in una cartella specifica sulla memoria del Raspberry.
   * Un database leggero (ad esempio **SQLite o un semplice file JSON**) può essere utilizzato per tenere traccia dei video disponibili e del video attualmente selezionato.

## ****Funzionalità del sistema****

1. **Avvio e Schermata Predefinita**
   * All’accensione del Raspberry, viene mostrata un’immagine statica sul display HDMI.
   * Il web server viene avviato automaticamente, permettendo la connessione alla rete WiFi del Raspberry.
2. **Gestione del Pulsante**
   * Quando il pulsante viene premuto, EmbeddedSubsystem invia un segnale a PlayVideo.
   * PlayVideo carica e avvia il video selezionato sullo schermo HDMI.
   * Alla fine del video, torna alla schermata predefinita.
3. **Interfaccia Web per il Controllo Remoto**
   * L’utente può connettersi all'hotspot WiFi del Raspberry e aprire una pagina web per gestire il sistema.
   * La pagina web offre:
     + Un pulsante per **caricare nuovi video**.
     + Una lista dei video disponibili con un'opzione per **selezionare il video attivo**.
     + Un’anteprima dell’immagine statica attualmente visualizzata.
4. **Supporto a Formati Video**
   * Il sistema deve supportare almeno i formati più comuni (MP4, AVI, MKV).
   * Potrebbe essere utile integrare un sistema di conversione per garantire la compatibilità con il player video.
5. **Possibili Estensioni Future**
   * **Modalità slideshow**: invece di un solo video, il sistema potrebbe riprodurre una playlist di video in sequenza.
   * **Integrazione con Bluetooth**: per controllare il sistema anche con un telecomando o un’app mobile dedicata.
   * **Streaming su dispositivi remoti**: gli utenti potrebbero visualizzare i video anche dai loro dispositivi mobili senza doverli scaricare.
   * **Timer o programmazione automatica**: per avviare la riproduzione video a determinati orari.

## ****Tecnologie Utilizzate****

| **Componente** | **Tecnologia** |
| --- | --- |
| **Gestione GPIO e Pulsante** | PI4J (Java) |
| **Riproduzione Video** | Framework da definire (es. OMXPlayer, VLC, GStreamer) |
| **Web Server** | Flask (Python) / Node.js / Vert.x (Java) |
| **Interfaccia Web** | HTML, CSS (Bootstrap), JavaScript |
| **Hotspot WiFi** | Hostapd e DHCPD |
| **Gestione File Video** | File System Linux + SQLite/JSON |

## ****Conclusione****

Il progetto fornisce una soluzione completa per la gestione di un sistema di riproduzione video interattivo su Raspberry Pi. Grazie al pulsante fisico, l'utente può avviare facilmente la riproduzione del video selezionato, mentre l'interfaccia web consente di caricare nuovi video e cambiare le impostazioni in modo intuitivo. Il sistema può essere esteso con nuove funzionalità per renderlo ancora più versatile.

**Analisi del Progetto**

**Obiettivo**

Realizzare un sistema embedded su Raspberry Pi in grado di:

* **Riprodurre video su HDMI** quando viene premuto un pulsante.
* **Gestire i video tramite un'interfaccia web**, accessibile via hotspot WiFi.
* **Consentire il caricamento e la selezione dei video da remoto**.

**Requisiti**

**Requisiti Funzionali**

1. Il sistema deve mostrare un’immagine statica sullo schermo HDMI quando non è in riproduzione.
2. Alla pressione del pulsante, deve avviarsi il video selezionato.
3. Quando il video termina, il sistema deve tornare all’immagine iniziale.
4. Il Raspberry Pi deve generare un **hotspot WiFi**, permettendo la connessione di dispositivi mobili.
5. Un **web server** deve permettere agli utenti di:
   * Caricare nuovi video.
   * Visualizzare i video disponibili.
   * Selezionare il video attivo per la riproduzione.

**Requisiti Non Funzionali**

1. Il sistema deve essere **reattivo e leggero**, ottimizzato per il Raspberry Pi.
2. Il server web deve essere **scalabile**, per supportare future espansioni.
3. Il sistema deve funzionare **senza connessione Internet**, grazie all’hotspot WiFi.
4. Il codice deve essere **modulare**, per facilitare la manutenzione e l’aggiunta di funzionalità.

**Design del Sistema**

Il sistema sarà suddiviso in **4 moduli principali**:

**1️ Modulo Embedded (Gestione del Pulsante e Hardware)**

**Obiettivo**: Intercettare la pressione del pulsante e inviare un segnale al modulo video.  
**Tecnologia**: **PI4J (Java)**  
**Componenti**:

* **Classe EmbeddedSubsystem**: gestisce l’input dal pulsante GPIO.
* **Listener degli eventi**: attende la pressione del pulsante e comunica con il modulo video.

**2️ Modulo Video (Riproduzione e HDMI)**

**Obiettivo**: Gestire la riproduzione dei video e il passaggio all’immagine statica.  
**Tecnologia**: **Java con un framework video** (es. VLCJ, GStreamer o OMXPlayer)  
**Componenti**:

* **Classe VideoPlayer**: gestisce il video attivo.
* **Classe ScreenManager**: cambia tra l’immagine statica e il video.
* **Main PlayVideo**: avvia il sistema e coordina il tutto.

**3️ Modulo Web Server (Vert.x)**

**Obiettivo**: Fornire un’interfaccia per gestire i video e comunicare con il sistema.  
**Tecnologia**: **Vert.x (Java) con REST API**  
**Componenti**:

* **Server HTTP Vert.x**: gestisce le richieste dalla web app.
* **Gestore File (VideoManager)**: salva e recupera i video.
* **Database (video\_db.json o SQLite)**: memorizza la lista dei video disponibili.
* **API REST** per:
  + Ottenere la lista dei video (GET /videos).
  + Caricare nuovi video (POST /upload).
  + Selezionare il video attivo (POST /select).

**4️ Modulo Hotspot WiFi**

**Obiettivo**: Creare una rete WiFi a cui gli utenti possono connettersi.  
**Tecnologia**: **Linux (hostapd, dnsmasq)**  
**Componenti**:

* **Script di configurazione WiFi**: avvia automaticamente l’hotspot all’accensione.
* **Gestione DHCP**: assegna IP ai dispositivi connessi.

+---------------------------------------------------+

| Raspberry Pi - Video System |

+---------------------------------------------------+

| [EmbeddedSubsystem] | [Web Server - Vert.x] |

| (PI4J - Java) | (REST API - Java) |

| - Pulsante GPIO | - Upload Video |

| - Listener Eventi | - Lista Video |

| - Segnale al Player | - Selezione Video |

+-------------------------+-----------------------+

| [Video Player - Java] |

| - Gestisce riproduzione video su HDMI |

| - Cambia tra video e immagine statica |

+---------------------------------------------------+

| [Hotspot WiFi - Linux] |

| - Rete WiFi privata |

| - DHCP e gestione connessioni |

|  |
| --- |
|  |

## ****Possibili Estensioni Future****

Supporto a più utenti contemporanei per la gestione video.  
Integrazione con **Bluetooth** per il controllo remoto.  
**Notifiche WebSocket** per aggiornare in tempo reale l’interfaccia web.  
Modalità **playlist** per riprodurre più video in sequenza.

Dispositivi smart room:  
Raspberry coordinatore:

1)hosta la LAN (ip statico)

2)hosta Mqtt broker

3)mqtt “client” che gestisce l interconnessione fra tutti I dispositivi

Raspberry Video:

1)Mqtt client

2)JavaFx + pi4j

Raspberry Audio:

Shelly Plug It:

Configurare sul suo http server

1)dandogli il wifi del broker

2) configurando mqtt dando ip broker

Comunica con mqtt on e off

Aggiunta dispositivo video

Configurazione iniziale:

Disattivare serivizi inutili (bluetooth,notifica usb…)

Disabilitare la luminosità e il risparmio

Aggiungere il servizio mqttVideo

Aggiungere la cartella contenente i 2 progetti (MqttClient e JavaFx + pi4j)

Risparmio energetico (stacca il servizio video lasciando solo mqtt, spegne tv, disattiva hdmi, rimuove ambiente grafico, limita cpu)

Salta la corrente elettrica: i raspberry si riaccendono in automatico quindi al loro riavvio partono i servizi e in ogni caso ritornano il modalita attiva, se un rasp video si accende prima del broker no problem perché prima di avviare la parte video aspetta la connessione con il broker

mosquitto\_pub -h 192.168.0.130 -t smartroom/pi1/cmd -m sleep

mosquitto\_pub -h 192.168.0.130 -t smartroom/pi1/cmd -m wake

sudo apt update

Samba

sudo apt install samba samba-common-bin

sudo nano /etc/samba/smb.conf

[Condivisa]

path = /home/pi/condivisa

browseable = yes

writeable = yes

guest ok = yes

public = yes

create mask = 0777

directory mask = 0777

force user = villasilvia

sudo systemctl restart smbd

java

sudo apt install openjdk-17-jdk

mvn

sudo apt install maven

RasPi

VillaRasp1

10.42.0.1

Diagramma sequenza

@startuml

actor Guida

participant BrokerMQTT as Broker

participant "Generic Shelly Plug" as Shelly

participant "Plafoniere" as Luci

participant "Raspberry-mp3" as Audio

participant "Stazione 1" as S1

participant "Stazione 2" as S2

participant "Stazione 3" as S3

participant "Stazione 4" as S4

== Avvio Stazione 1 ==

Guida -> S1 : trigger Sonar

S1 -> Broker : publish("stazione/1/triggered")

Broker -> Luci : setBrightness(50)

Broker -> Audio : setVolume(50)

S1 -> S1 : wait 3s

S1 -> S1 : playVideo()

== Fine video S1 ==

S1 -> Broker : publish("stazione/1/ended")

S1 -> S1 : autoDestruction()

== Avvio Stazione 2 ==

Broker -> S2 : accensione (via MQTT)

Guida -> S2 : trigger Sonar

S2 -> S2 : playVideo()

S2 -> Broker : publish("stazione/2/triggered")

Broker -> Broker : on timer (es. 5s)

Broker -> Shelly : on()

Broker -> Broker : off timer (es. 20s)

Broker -> Shelly : off()

== Fine video S2 ==

S2 -> Broker : publish("stazione/2/ended")

S2 -> S2 : autoDestruction()

== Stazione 3 ==

Broker -> S3 : accensione (via MQTT)

S3 -> Broker : publish("stazione/3/triggered")

Broker -> S3 : playVideo()

S3 -> Broker : publish("stazione/4/start")

== Stazione 4 ==

Broker -> S4 : accensione

S4 -> Broker : publish("stazione/4/triggered")

Broker -> S4 : playVideo()

== Fine stazione 4 ==

S4 -> Broker : publish("stazioni/fine")

Broker -> Luci : setBrightness(100)

Broker -> Audio : setVolume(100)

@enduml