

Centralized Video Playback System for Showroom TVs

Project Overview

Maruti Suzuki showrooms across multiple locations have TVs displaying promotional and informational videos. Currently, there's no centralized system to manage what each TV plays. Our goal is to build a central hub that decides, distributes, and monitors video playback across all showroom TVs, regardless of brand or OS.

1. Key Requirements

R1	Centralized Control	Central hub decides which video plays on each TV.
R2	Offline Resilience	TV continues local playback when disconnected from internet.
R3	Health Monitoring	Each TV reports playback and storage status periodically.
R4	Dashboard Visualization	Dashboard displays live status and video metrics.

2. System Architecture

The system consists of three layers — Central Hub, Communication Layer, and Edge (TV Module) Layer. Below are detailed roles and data flows.

Step 1 – Central Hub Layer (*Updated*)

The central hub handles both content distribution and status collection. It hosts video uploads, detects new files, assigns them to TVs, pushes them via secure protocols (HTTP, MQTT, or SFTP), and receives tracking data (heartbeat, playback info, storage health) from each TV agent. The data is stored for dashboard visualization and alerting on playback issues.

Step 2 – Communication Layer

Maintains secure, persistent communication between Hub and TV modules via WebSocket or MQTT. Fallback HTTP polling ensures updates when network reliability is low.

Step 3 – Edge Layer (TV Module)

Raspberry Pi modules act as agents connected to TVs. They receive and store videos locally, use Kodi or Python-VLC for playback, and continue looping videos offline if disconnected. Agents periodically send status reports to the hub with logs and telemetry.

3. LibreELEC + Kodi Integration (Recommended)

LibreELEC provides a lightweight Linux OS for media playback on Raspberry Pi. Kodi acts as the playback engine and exposes a JSON-RPC API to control playback, playlists, and logs. We can create a Kodi Python add-on that polls the central hub for new assignments, downloads videos, and plays them automatically.

Component	Role
LibreELEC	Minimal Linux-based media OS with SSH and Python add-on support.
Kodi	Media player software supporting playlists, looping, and remote APIs.

Integration Benefits

- No dependency on Smart TV OS versions.
- Auto-boot and auto-resume playback after reboot.
- Lightweight, uniform deployment for all brands.
- Python-based control for easy updates and remote logs.

4. Implementation Plan

Phase	Task	Tools / Libraries	Outcome
1	Build central hub backend	FastAPI, Watchdog, SQLAlchemy	Video distribution and tracking server.
2	Deploy LibreELEC on Pi modules	LibreELEC Image Creator	TV boots directly to Kodi player.
3	Develop Kodi Python Add-on	Kodi API, Requests, MQTT	Plays videos, sends telemetry.
4	Build Monitoring Dashboard	React, Streamlit, Grafana	Visualizes live TV and status data.
5	Testing and Resilience	Network simulations	Ensures stable offline playback and recovery.

5. Dashboard Features

Dashboard will show:

- Online/offline status of TVs.
- Current video name and duration.
- Storage usage and last report time.
- Manual controls to assign or restart playback.

6. Final Tech Stack Summary

Layer	Technology
Central Hub	FastAPI, Watchdog, PostgreSQL, MQTT broker (Mosquitto)
Edge Device	LibreELEC + Kodi Python Add-on
Media Playback	Kodi Player (autoplay loop mode)
Dashboard	React + Streamlit / Grafana
Monitoring	MQTT logs → Prometheus / Grafana metrics

This refined design ensures scalable, brand-independent, and fault-tolerant media playback across all showroom TVs. It leverages open-source Python-compatible tools, supports offline operation, and provides real-time monitoring from a central dashboard.