Raspberry Pi Digital Signage System - Complete Implementation with Fixes

This updated version includes all fixes discovered during deployment and testing.

🚀 Quick Start

```bash
# Clone the repository
git clone https://github.com/js-barg/digital-signage-system.git
cd digital-signage-system

# Extract files and install
python3 extract_files.py
sudo mv ~/digital-signage-system /home/admin/media-player
cd /home/admin/media-player
sudo ./scripts/install-media.sh --verbose
```

🔧 Critical Fixes Applied

1. Prometheus Metrics Import Fix

The original `flask-prometheus-metrics` package has import issues. Fixed in `app.py`:

python

```python
#!/usr/bin/env python3
import os
import yaml
import importlib
from flask import Flask, render_template, redirect, url_for
# from flask_prometheus_metrics import PrometheusMetrics  # COMMENTED OUT - Import issue
from utils.monitoring import get_system_stats

app = Flask(__name__)
# metrics = PrometheusMetrics(app)  # COMMENTED OUT

# Load configuration
with open('config.yaml', 'r') as f:
    config = yaml.safe_load(f)

# Dynamic mode loading - FIXED counter issue
enabled_modes = {}
mode_counter = 1
for mode_name, mode_config in config['modes'].items():
    if mode_config.get('enabled', False):
        try:
            module_name = f"modes.mode{mode_counter}_{mode_name}"
            mode_counter += 1
            ui_module = importlib.import_module(f"{module_name}.ui")
            app.register_blueprint(ui_module.blueprint, url_prefix=f"/admin/modes/{mode_name}")
            enabled_modes[mode_name] = {
                'module': module_name,
                'config': mode_config
            }
        except ImportError as e:
            print(f"Failed to load mode {mode_name}: {e}")

@app.route('/')
def index():
    return redirect(url_for('admin'))

@app.route('/admin')
def admin():
    return render_template('admin.html', modes=enabled_modes, stats=get_system_stats())

@app.route('/health')
# @metrics.do_not_track()  # COMMENTED OUT
def health():
    return {'status': 'ok', 'modes': list(enabled_modes.keys())}

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=False)
```

2. Nginx IPv6 Fix

Fixed nginx trying to connect to IPv6 localhost. Update `/home/admin/media-player/nginx/templates/media-player.conf.j2`:

```nginx
server {
    listen 80;
    server_name _;
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    server_name _;

    ssl_certificate /etc/ssl/certs/media-player.crt;
    ssl_certificate_key /etc/ssl/private/media-player.key;

    location / {
        proxy_pass http://127.0.0.1:5000;  # Changed from localhost to 127.0.0.1
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location /static {
        alias /home/admin/media-player/app/static;
        expires 1h;
    }

    location /media {
        alias /home/admin/media-player/media;
        expires 1d;
    }
}
```

3. Systemd Service Template Fix

Fixed the Jinja2 template error in `/home/admin/media-player/systemd/templates/media-mode.service.j2`:

```ini
[Unit]
Description=Digital Signage {{ mode_name }} Mode - Display {{ display_id }}
After=media-controller.service
Requires=media-controller.service

[Service]
Type=simple
User=admin
Group=admin
WorkingDirectory=/home/admin/media-player/app
Environment="DISPLAY=:0.{{ display_id }}"
Environment="PATH=/home/admin/media-player/app/venv/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
ExecStart=/home/admin/media-player/app/venv/bin/python -m modes.mode{{ mode_index }}_{{ mode_name }}.servi
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
```

4. File Extraction Script

Create `/home/admin/media-player/extract_files.py` for extracting files from the markdown:

python

````python
#!/usr/bin/env python3
import re
import os
import sys

def extract_files_from_markdown(markdown_file):
    """Extract all code files from the markdown documentation"""

    print(f"Reading {markdown_file}...")
    with open(markdown_file, 'r') as f:
        content = f.read()

    # Pattern to match file paths and code blocks
    pattern = r'### `(/home/admin/media-player/[^`]+)`\s*\n\s*```(?:[\w]+)?\n(.*?)```'
    matches = re.findall(pattern, content, re.DOTALL | re.MULTILINE)

    if not matches:
        print("No files found to extract!")
        return

    print(f"Found {len(matches)} files to extract")

    for filepath, code in matches:
        # Convert absolute path to relative path
        relative_path = filepath.replace('/home/admin/media-player/', '')

        # Create directory if needed
        dirpath = os.path.dirname(relative_path)
        if dirpath and not os.path.exists(dirpath):
            os.makedirs(dirpath, exist_ok=True)
            print(f"Created directory: {dirpath}")

        # Write file
        try:
            with open(relative_path, 'w') as f:
                f.write(code.strip())
            print(f"Created: {relative_path}")

            # Make scripts executable
            if relative_path.endswith('.sh') or relative_path.endswith('.py'):
                os.chmod(relative_path, 0o755)

        except Exception as e:
````

```
            print(f"Error creating {relative_path}: {e}")

    # Create empty __init__.py files
    init_files = [
        'app/__init__.py',
        'app/utils/__init__.py',
        'app/modes/__init__.py',
        'app/modes/mode1_recognition/__init__.py',
        'app/modes/mode2_video/__init__.py',
        'app/modes/mode3_kiosk/__init__.py',
        'app/modes/mode4_slideshow/__init__.py'
    ]

    for init_file in init_files:
        dirpath = os.path.dirname(init_file)
        if dirpath and not os.path.exists(dirpath):
            os.makedirs(dirpath, exist_ok=True)

        if not os.path.exists(init_file):
            open(init_file, 'a').close()
            print(f"Created: {init_file}")

    print("\nExtraction complete!")

if __name__ == '__main__':
    if len(sys.argv) > 1:
        extract_files_from_markdown(sys.argv[1])
    else:
        # Look for markdown file in current directory
        md_files = [f for f in os.listdir('.') if f.endswith('.md') and 'digital-signage' in f]
        if md_files:
            extract_files_from_markdown(md_files[0])
        else:
            print("No digital signage markdown file found!")
            print("Usage: python3 extract_files.py [markdown_file]")
```

5. Updated Requirements

Fixed `/home/admin/media-player/app/requirements.txt`:

```
Flask==2.3.2
Flask-HTMX==0.3.1
prometheus-flask-exporter==0.23.0
PyYAML==6.0
Pillow==10.0.0
pygame==2.5.0
psutil==5.9.5
Jinja2==3.1.2
requests==2.31.0
python-mpv==1.0.1
```

6. Installation Script Permission Fix

Updated `/home/admin/media-player/scripts/install-media.sh` to handle log permissions:

bash

```bash
#!/bin/bash
set -e

# Digital Signage Installer for Raspberry Pi 4
# Usage: ./install-media.sh [--dry-run] [--verbose] [--resume] [--auto-reboot]

SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
PROJECT_ROOT="$(dirname "$SCRIPT_DIR")"
LOG_FILE="$PROJECT_ROOT/logs/media-player-install.log"  # Changed to local directory

# Create logs directory
mkdir -p "$PROJECT_ROOT/logs"

# ... rest of the installer script remains the same
```

📁 Complete Project Structure

```
/home/admin/media-player/
├── app/
│   ├── __init__.py
│   ├── app.py              # Fixed Prometheus import and mode counter
│   ├── config.yaml
│   ├── requirements.txt    # Updated with correct packages
│   ├── utils/
│   │   ├── __init__.py
│   │   ├── display.py
│   │   └── monitoring.py
│   ├── modes/
│   │   ├── __init__.py
│   │   ├── mode1_recognition/
│   │   │   ├── __init__.py
│   │   │   ├── ui.py
│   │   │   ├── renderer.py
│   │   │   ├── config.json
│   │   │   └── templates/
│   │   │       └── index.html
│   │   ├── mode2_video/
│   │   │   └── ... (same structure)
│   │   ├── mode3_kiosk/
│   │   │   └── ... (same structure)
│   │   └── mode4_slideshow/
│   │       └── ... (same structure)
│   ├── templates/
│   │   ├── base.html
│   │   └── admin.html
│   └── static/
│       └── style.css
├── media/
│   ├── backgrounds/
│   ├── images/
│   ├── videos/
│   └── text/
├── logs/                   # Added for local logging
├── scripts/
│   ├── install-media.sh    # Fixed log path
│   ├── uninstall.sh
│   ├── update.sh
│   └── preflight_check.py
├── systemd/
│   └── templates/
│       ├── media-controller.service.j2
│       └── media-mode.service.j2  # Fixed Jinja2 template
├── nginx/
│   └── templates/
│       └── media-player.conf.j2   # Fixed IPv6 issue
├── docs/
│   ├── API.md
│   ├── TROUBLESHOOTING.md
│   └── templates/
│       └── playlist.csv
├── extract_files.py          # Added for easy extraction
└── README.md
```

🛠️ Manual Fixes After Installation

If you encounter issues after running the installer:

1. Fix Prometheus Import (if not using extraction script)

bash

```bash
# Edit app.py
nano /home/admin/media-player/app/app.py
# Comment out the two Prometheus lines as shown above
```

2. Fix Nginx IPv6 Issue

bash

```bash
# Edit nginx config
sudo nano /etc/nginx/sites-available/media-player.conf
# Change 'localhost' to '127.0.0.1' in proxy_pass
sudo nginx -t
sudo systemctl reload nginx
```

3. Fix Permission Issues

bash

```bash
# Fix ownership of virtual environment
sudo chown -R admin:admin /home/admin/media-player/app/venv/

# Fix media directory permissions
sudo chown -R admin:admin /home/admin/media-player/media/
```

4. Create Systemd Services Manually (if template fails)

bash

```
sudo tee /etc/systemd/system/media-controller.service << 'EOF'
[Unit]
Description=Digital Signage Controller Service
After=network.target

[Service]
Type=simple
User=admin
Group=admin
WorkingDirectory=/home/admin/media-player/app
Environment="PATH=/home/admin/media-player/app/venv/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
ExecStart=/home/admin/media-player/app/venv/bin/python app.py
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
EOF

sudo systemctl daemon-reload
sudo systemctl enable media-controller.service
sudo systemctl start media-controller.service
```

🧪 Testing Commands

After installation, verify everything is working:

```bash
# Check services
sudo systemctl status media-controller.service
sudo systemctl status nginx

# Test endpoints
curl http://localhost:5000/health
curl -k https://localhost/health

# Check logs
sudo journalctl -u media-controller.service -n 50

# Get IP for web access
hostname -I | cut -d' ' -f1
```

🚨 Common Issues and Solutions

Issue: "No module named 'yaml'"

```bash
cd /home/admin/media-player/app
source venv/bin/activate
pip install -r requirements.txt
deactivate
sudo systemctl restart media-controller.service
```

Issue: Modes not loading

```bash
# Check for missing ui.py files
find /home/admin/media-player/app/modes -name "ui.py"
# If missing, copy from the fixed code sections above
```

Issue: Port 5000 already in use

bash

```bash
# Find what's using port 5000
sudo lsof -i :5000
# Kill the process or restart the service
sudo systemctl restart media-controller.service
```

Issue: Certificate warnings

This is normal with self-signed certificates. In your browser:

1. Click "Advanced"
2. Click "Proceed to site (unsafe)"
3. The site will load normally

🎯 Next Steps

1. Access the web interface at `https://<your-pi-ip>`
2. Add media files to the appropriate directories
3. Configure each mode through the web interface
4. Start displaying content on your HDMI screens!

📝 Development Notes

- Each mode is a separate Flask Blueprint
- Modes can be developed independently
- The renderer.py files contain the actual display logic
- The ui.py files handle the web interface
- Templates must extend 'base.html'
- All static files go in mode-specific static/ directories

This updated version includes all fixes discovered during the deployment session. For the latest updates, check the GitHub repository.