Title: An Offline ESP32-Based WiFi File Server with OLED Feedback and Web-Based File Management Interface

Abstract:
This research presents the design and implementation of a cost-effective, offline, ESP32-based file server capable of handling local file transfers between multiple devices without the need for internet connectivity. The system creates a local WiFi access point using ESP32's softAP feature, allowing devices to connect and interact with an intuitive web interface for file management. Key features include SD card integration, an OLED display for live feedback, file analytics, multi-user support, multilingual compatibility, and a built-in offline code editor. This paper explores the technical architecture, implementation challenges, comparative analysis with cloud-based and LAN-based solutions, and the practical utility of the solution in educational, rural, and emergency communication environments.

---

1. Introduction

With the rapid advancement in IoT and edge computing, file sharing systems have predominantly moved toward cloud-based solutions. However, many environments lack consistent internet access. This project proposes an innovative solution using the ESP32 microcontroller to create an offline file-sharing platform with a responsive web interface, managed entirely within a localized WiFi network. The project provides a unique combination of embedded hardware and intuitive front-end software to facilitate secure, efficient file sharing.

---

2. Motivation and Problem Statement

Transferring files across different device platforms—especially between Android, iOS, and desktop systems—often requires internet connectivity or platform-specific applications. This becomes problematic in exam halls, disaster zones, rural areas, or even offline labs. A low-cost, portable file server capable of operating offline over WiFi would significantly improve cross-device accessibility and operational independence.

---

3. System Architecture and Components

3.1. Hardware Components:

- **ESP32 Dev Board**: Core controller providing WiFi access point and SPI interface.
- **MicroSD Card Module**: Connected via VSPI for storage.

- **1.3” SH1106 OLED Display**: Provides real-time operational feedback.
- **Buttons**: For physical authentication, reset, and mode switching.
- **In-built Battery**: Ensures portability and continuous operation.

3.2. Software Components:

- Arduino Framework with ESP32 Core
- WebServer Library for HTTP services
- Adafruit SH110X and GFX libraries for OLED rendering
- HTML/CSS/JS UI with embedded scripts
- Optional: mDNS for dynamic hostname support (jai.local)
- Client-side Monaco Editor for in-browser code editing

---

4. Features of the Proposed System

4.1. File Management:

-

Upload, download, rename, delete files/folders via web UI.

- Display current directory and storage usage.
- Create nested folders and navigate intuitively.

4.2. OLED Feedback:

- Shows current AP IP address and SSID.
- Upload progress, operation success/failure messages.
- Dynamic updates of browsing status.

4.3. Security and Access Control:

- User creation and login system.
- Password-protected file access.
- Button-triggered physical access unlock.

4.4. Analytics and Language Support:

- File usage statistics and system logs.
-

Interface supports multiple languages.

4.5. Embedded Code Editor:

- Built-in editor using Monaco or CodeMirror.
- Edit HTML, JS, TXT, CPP files directly in browser.
- Save changes back to ESP32.

4.6. Local DNS via mDNS:

- Access server via hostname (e.g., http://jai.local) instead of IP.

---

5. Use Cases

- **Educational Labs:** Offline file distribution and collection.
- **Rural Schools:** Low-cost digital content sharing.
- **Emergency Response:** Sharing files in disaster-hit or no-internet zones.
- **Cross-Platform Sharing:** Android to iOS to PC without cloud.

---

6. Results and Discussion

Benchmarks show a transfer speed of approximately 50-60 KB/s for ESP32 systems, which is acceptable for small-to-medium file sizes. With Raspberry Pi as base, speed can increase 5–10x, though at the cost of power and size. OLED improves UX significantly, giving real-time assurance of operations.

Challenges Faced:

- Unicode (Hindi) rendering on OLED unsupported.
- mDNS reliability varies across devices.
- Offline Monaco editor integration required storing JS libraries on SD card.

---

7. Cost Analysis

Component	Unit Cost (INR)
ESP32 Dev Board	300
OLED Display	200
SD Card Module	50
32GB SD Card	400
Buttons & Connectors	15
Li-ion Battery	150
Total per Unit	**1115 INR**

---

8. Conclusion

This project demonstrates a novel offline file sharing solution using ESP32. The system is practical, economical, and highly adaptable for educational and emergency applications. It opens pathways for further extensions including peer-to-peer architecture, external displays, and integration of data security protocols.

References

1. Espressif Documentation: https://docs.espressif.com

2. Adafruit OLED Libraries

3. Monaco Editor: https://microsoft.github.io/monaco-editor

4. Arduino Forum Discussions

5. Existing patents on file server and local wireless transmission