[1]S. Saarim Fowlaath

[2]Dr. M. Syed Masood*

Development of a Customizable Cloud Storage using Raspberry Pi

Abstract: - This project presents a cost-effective and customizable cloud storage solution using a Raspberry Pi 5. Unlike traditional cloud services that often involve high costs and limited control, this self-hosted system allows users to manage their data efficiently. The backend is powered by Apache, JSON is data handling with a user-friendly interface developed using PHP, HTML, and CSS. For remote access, a subdomain is utilized, while manages secure public network access using IPv4 and IPv6. This setup ensures seamless and reliable file access from any location. The system supports read and write access for multiple users, making it suitable for personal use, small businesses, and educational institutions. By leveraging the flexibility and affordability of Raspberry Pi, this project offers an accessible alternative to mainstream cloud storage platforms. Additionally, it ensures data privacy and customization, providing users with full control over their files.

Keywords: Cloud Storage, Raspberry Pi 5, Self-Hosted Storage, Apache, HTML, PHP, CSS, JSON, Web-Based File Sharing, Secure File Access, Customizable Storage Solution.

I. INTRODUCTION

In today's digital world, cloud storage plays a crucial role in managing and accessing data from anywhere. However, many commercial cloud services come with high costs, privacy concerns, and limited customization. This project focuses on creating a personal, self-hosted cloud storage system using Raspberry Pi 5, offering a flexible and cost-effective alternative. The system is powered by an Apache server for backend operations, while PHP, HTML, and CSS are used to develop a simple and user-friendly interface. JSON is utilized for efficient data handling, ensuring smooth communication between the frontend and backend. For remote access, a subdomain is set up, and Cloudflare is used to enable secure connectivity with both IPv4 and IPv6 support. It useful for file sharing. Designed for personal users, small businesses, and educational purposes, this solution is scalable, efficient, and easy to deploy. It ensures users have complete control over their data while maintaining privacy and security.

II. LITERATURE SURVEY

Cloud storage has become an essential part of our daily lives, with services like Google Drive and Dropbox making it easy to store and access files online. However, these platforms often come with limitations such as subscription fees, data privacy concerns, and limited customization. Many users are now exploring self-hosted alternatives to have more control over their data. Researchers have shown that devices like Raspberry Pi can be an effective, low-cost solution for building personal cloud storage systems.

Other studies have also explored how to improve the performance and accessibility of self-hosted storage, providing reliable performance JSON is often chosen for data handling due to its lightweight nature and ease of integration with web-based interfaces. To allow remote access, Dynamic DNS services like DuckDNS are commonly used, along with platforms like Cloudflare for enhanced security and accessibility using IPv4 and IPv6. Many previous implementations have shown the importance of strong authentication and access controls to protect data.

[1] S. Saarim Fowlaath, IV Sem. MCA Student, Department of Computer Applications, B. S. Abdur Rahman Crescent Institute of Science and Technology, GST Road, Vandalur, Chennai -48, saarimfowlaath@gmail.com

[2] *Corresponding author : Dr. M. Syed Masood, Associate Professor and Head, Department of Computer Applications, B. S. Abdur Rahman Crescent Institute of Science and Technology, GST Road, Vandalur, Chennai -48, ms.masood@crescent.education

III. EXISTING SYSTEM

In the current scenario, most users rely on commercial cloud storage services such as Google Drive, Dropbox, and OneDrive to store and manage their data. These platforms offer convenient access to files from any location, along with features like automatic backups and file synchronization. However, they often come with limitations such as restricted storage space, subscription costs, and concerns over data privacy. Users have limited control over their data, as it is stored on third-party servers, making it vulnerable to data breaches or misuse.

The most significant drawback was lack of portability—users could only access their files when connected to the same local network, restricting remote accessibility. these Raspberry Pi 4-based systems often relied on ownCloud or Nextcloud, which, while functional, required high system resources and were often complex to configure. The performance limitations of Raspberry Pi 4, combined with the lack of an efficient way to extend access beyond the LAN, made these setups impractical for users needing remote file access.

Challenges in the Existing System:

- Limited Remote Access in Raspberry Pi 4-Based Systems.
- High Resource Consumption of OwnCloud and Nextcloud.
- Complex Setup and Maintenance.
- Dependency on Third-Party Cloud Services.
- Lack of Customization and Scalability.

IV. PROPOSED SYSTEM

This project aims to create a customizable, self-hosted cloud storage system using Raspberry Pi 5. Unlike earlier setups with Raspberry Pi 4, the upgraded hardware provides better speed and performance, making file storage and access more efficient. The system uses an Apache server for backend operations, while PHP, HTML, and CSS are used to design a simple and user-friendly interface. JSON is applied for effective data management, keeping the path of stored files. For remote access, the system uses a DuckDNS subdomain connected with Cloudflare, ensuring secure and reliable access through both IPv4 and IPv6. This makes it easy to access files from any network without complicated configurations. Additionally, the system supports open read and write access within a trusted environment, making it suitable for home networks, small offices, or educational purposes. With a focus on affordability and customization, the proposed solution offers a flexible alternative to commercial cloud storage platforms.

V. METHODOLOGY

The development of this customizable cloud storage system follows a structured approach to ensure efficiency, security, and ease of use. The methodology consists of several key phases, including system design, implementation, configuration, and testing, to create a seamless and reliable self-hosted storage solution.

A. System Architecture Overview:

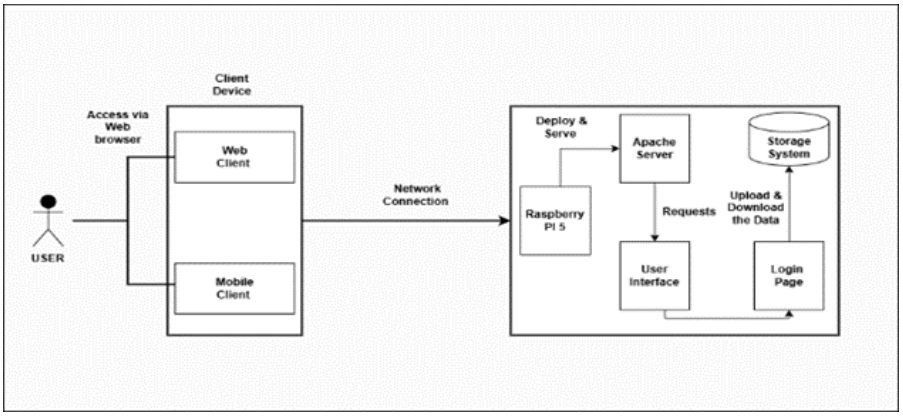

Fig. 1 System Architecture

The architecture of this customizable cloud storage system is designed to provide users with a seamless, self-hosted, and portable file management solution using Raspberry Pi 5. The system is structured into two main components: Client Side and Server Side, connected via a network to facilitate secure data access and storage.

This diagram shows how a self-hosted cloud storage system works using a Raspberry Pi 5. The system is designed so users can easily access and manage their files remotely. It has two main sections: the client side and the server side. On the client side, users can access the cloud through a web browser using their laptops, desktops, or mobile phones. The interface is simple and accessible on both mobile and desktop devices, making it easy to upload, download, and manage files using HTML, CSS, JSON and PHP.

For secure remote access, a subdomain is created using DuckDNS and configured with Cloudflare. This allows users to connect to their cloud storage from any location. Both IPv4 and IPv6 addresses are used to ensure compatibility across different networks. Users simply log in using the provided login page, which ensures that only authorized people can access the system. Once logged in, they can navigate through files, upload documents, or download content.

On the server side, the Raspberry Pi 5 runs an Apache Server that handles all incoming requests. When a user performs an action, the server processes the request and manages the data stored on the connected storage. The Apache Server ensures smooth communication between the user interface, storage system, and login page. It also ensures that files are uploaded or downloaded without delays, providing an efficient and responsive experience.

The storage system consistently connected to the Raspberry Pi. It serves as the primary location where files are stored. JSON is used for login data, storing path information. This format ensures data is well-organized and quickly accessible, providing a smooth experience for users. JSON also makes data management simple and reliable.

With this system, users have complete control over their data without relying on third-party cloud providers. The Raspberry Pi setup makes it a cost-effective and energy-efficient solution. The project is ideal for personal use, small businesses, or educational purposes. Additionally, it offers flexibility in managing files, making it a great alternative to commercial cloud storage.

B. Flow Overview:

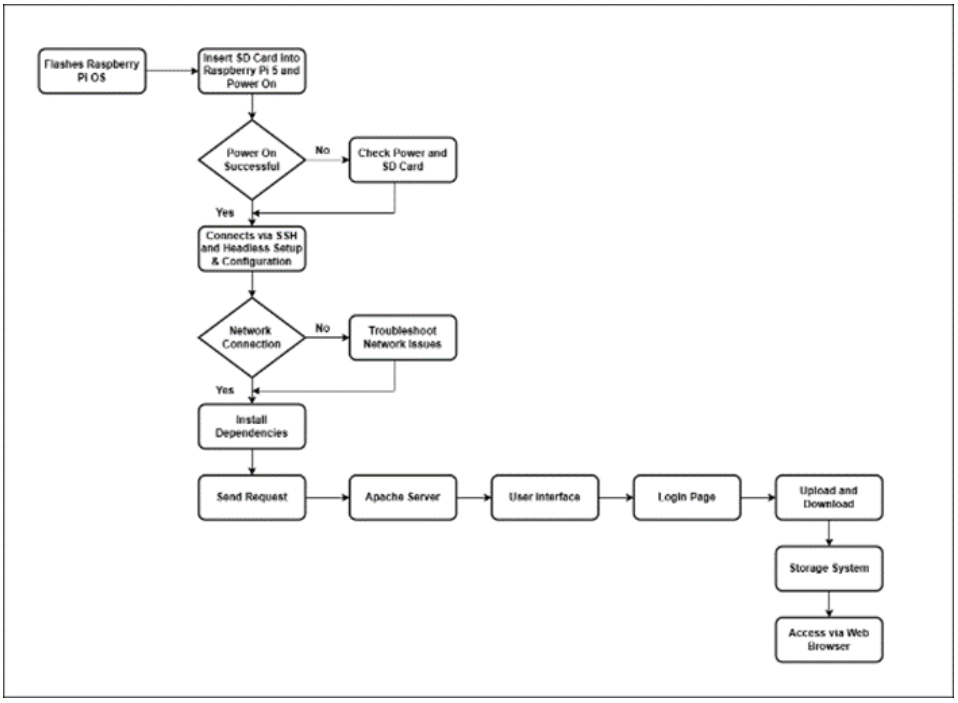

Fig. 2 Flow Diagram

This flowchart shows the step-by-step process of setting up a cloud storage system using a Raspberry Pi 5. It begins with flashing the Raspberry Pi OS onto an SD card using software like Raspberry Pi Imager. After the OS

is installed, the SD card is inserted into the Raspberry Pi, and the device is powered on. If the Pi doesn't start, basic troubleshooting is done by checking the power supply and ensuring the SD card is properly inserted. Once powered on, the Raspberry Pi is accessed remotely using SSH for a headless setup, meaning no monitor is needed. If network issues arise during this step, troubleshooting is carried out to fix the connection.

After successfully establishing the connection, necessary software dependencies are installed. This includes Apache Server for backend management, PHP for server-side scripting, and HTML, CSS, and JSON for the user interface and data handling. With the server ready, users can send requests to access the storage system. The web-based interface allows users to log in and manage their files. They can upload, download, and organize data using a simple and user-friendly design.

The login page ensures secure access, validating user credentials before granting permissions. Once authenticated, users can manage files directly from their web browser. The storage system handles the actual file management, with the Raspberry Pi connected to an external HDD for data storage. JSON is used to store and manage user information and file data effectively. The Apache server processes all file management requests and ensures smooth operation.

For external access, the cloud storage is made available over the internet. A subdomain is created using DuckDNS, which is then linked to Cloudflare. By adding the public IPv4 and IPv6 addresses, users can securely access their files from anywhere.

C. Hardware and Software Setup:

Hardware Requirements:

• Raspberry Pi 5 – Serves as the main processing unit for hosting the cloud storage. It runs the software, manages file requests, and ensures smooth data transfer. Its low power consumption makes it ideal for continuous operation.

• MicroSD Card (at 64GB) – Stores the Raspberry Pi OS and system files necessary for booting. A higher capacity SD card improves performance and storage management. It ensures stable operation and smooth execution of tasks.

• External HDD/SSD – Provides additional storage capacity for user files, enabling large-scale data management. It allows flexible storage expansion as per user requirements. An HDD is cost-effective, while an SSD offers faster read/write speeds.

• Power Supply for Raspberry Pi 5 – Ensures stable power delivery for uninterrupted performance. A high-quality power adapter prevents voltage fluctuations and overheating issues. It is crucial for maintaining the reliability of the system.

• Laptop with Windows 11 – Used for the initial setup, configuration, and remote access of the Raspberry Pi. It allows SSH and VNC connections for headless operation. The laptop also assists in software installation and troubleshooting.

• Wireless Network – Provides network connectivity for local and remote access. It enables users to connect to the Raspberry Pi over Wi-Fi without requiring an Ethernet connection. This ensures portability and accessibility.

Software Requirements:

• Raspberry Pi OS (64-bit Lite) – A lightweight operating system that runs on the Raspberry Pi. It optimizes system performance and supports all required dependencies. The Lite version is preferred for better efficiency in a headless setup.

• Apache Server – A reliable web server that handles HTTP requests and serves the PHP-based user interface. It ensures fast and secure access to the cloud storage system, enabling users to upload, download, and manage their files.

• HTML, CSS, JSON – Used for designing the user-friendly web interface. HTML structures the pages, CSS enhances the design, and JSON manages data storage and exchange. These technologies make the cloud storage interface interactive and accessible.

• SSH and VNC – Enables headless setup and remote access to Raspberry Pi. SSH allows command-line control, while VNC provides a graphical interface. These tools ensure easy configuration and management without a dedicated monitor.

• Cloudflare and DuckDNS – Used for securing and managing domain access. DuckDNS provides a free subdomain for remote access, while Cloudflare enhances security through DNS management, ensuring reliable access with DDoS protection.

D. Networking and Remote Access Configuration

Networking and remote access are critical components of the cloud storage system, enabling users to access their files from any location using a web browser. The system is configured to ensure secure and reliable connectivity through a series of steps.

• Local Network Access – The Raspberry Pi 5 is connected to the local network using a mobile hotspot. The device is assigned a static IP address for easy identification within the network. This allows initial configuration, testing, and file management within the local environment.

• SSH for Headless Setup – Since the Raspberry Pi is set up without a monitor (headless setup), SSH (Secure Shell) is used for remote command-line access. SSH allows the user to configure the device, install software packages, and manage services directly from their laptop. Additionally, VNC (Virtual Network Computing) can be used for graphical access if needed.

• Dynamic DNS (DDNS) Setup – Since mobile networks often assign dynamic IP addresses, a DDNS service like DuckDNS is used. DuckDNS provides a stable subdomain that maps to the Raspberry Pi's current public IP address. It continuously updates to ensure users can access the system using a consistent domain name.

• Cloudflare for Enhanced Security – The DuckDNS subdomain is integrated with Cloudflare for additional security and performance enhancements. Cloudflare provides DNS management, SSL/TLS encryption, and DDoS protection. It ensures the system remains accessible and secure, even in the event of cyber threats.

• User Access and Authentication – Once the network configuration is complete, users can securely access the storage system through the web interface using the DuckDNS subdomain. A login page ensures authentication, preventing unauthorized access. File upload, download, and management operations can be performed seamlessly from any device with an internet connection.

VI. RESULTS AND DISCUSSION

The customizable cloud storage system using Raspberry Pi 5 was successfully developed and deployed, providing a reliable and budget-friendly storage solution. The user interface, designed using HTML, CSS, and PHP, offers a straightforward experience for users to manage files efficiently. JSON is used for seamless data handling, ensuring smooth interactions. With the implementation of DuckDNS for dynamic DNS and Cloudflare for security, remote access was established effectively. Performance tests demonstrated the stability of the Apache server under regular use, with minimal resource consumption. Security features such as authentication and SSL encryption ensured data privacy and protected against unauthorized access.

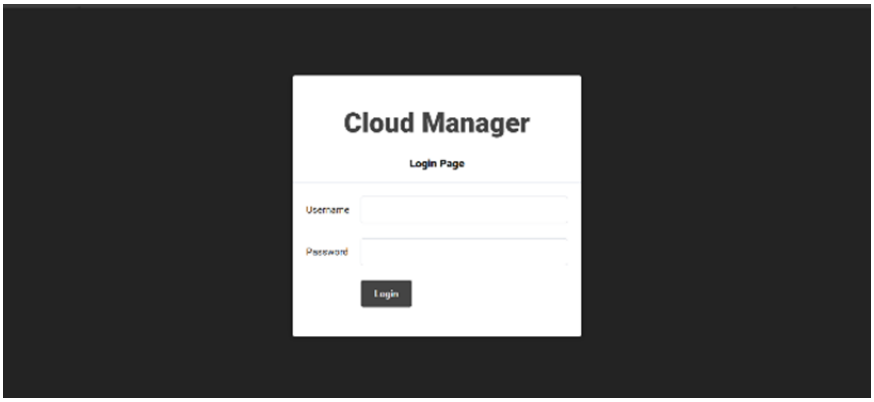

Fig. 3 Login Page

The login interface has a clean and minimal design, with a centered login form that includes fields for Username and Password. Below these fields, there is a Login button to authenticate users. The interface is user-friendly, with clear labels and a well-structured layout. The login page serves as the entry point for users to access their cloud storage data securely. Upon successful authentication, users will likely be redirected to the main dashboard to manage and interact with their files.

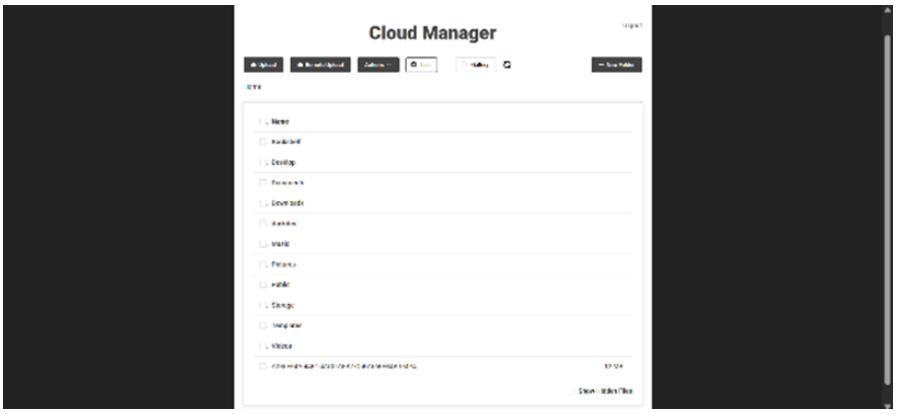

Fig. 4 Dashboard Page

The Cloud Manager dashboard serves as a convenient and easy-to-use platform for managing files stored on a server. After users log in, they are presented with this clean and organized interface that clearly displays folders and files. At the top, there are several buttons for quick access to essential functions. The Upload button allows users to add new files from their devices, while the Remote Upload option enables importing files from other servers. Additionally, the Actions dropdown provides options like deleting, moving, or renaming files, making file management straightforward.

Users can switch between List and Gallery views, depending on their preferences for viewing files. A Refresh button is available to reload and update the displayed contents, ensuring the latest files and changes are visible. For easy organization, there is a New Folder button to create directories directly from the interface. The dashboard displays a familiar folder structure, including common directories like Documents, Downloads, Music, Pictures, and Videos, along with additional folders. For more advanced users, the Show Hidden Files option reveals any system or configuration files. Finally, the Logout button in the top right corner ensures secure exit from the application, maintaining privacy and security.

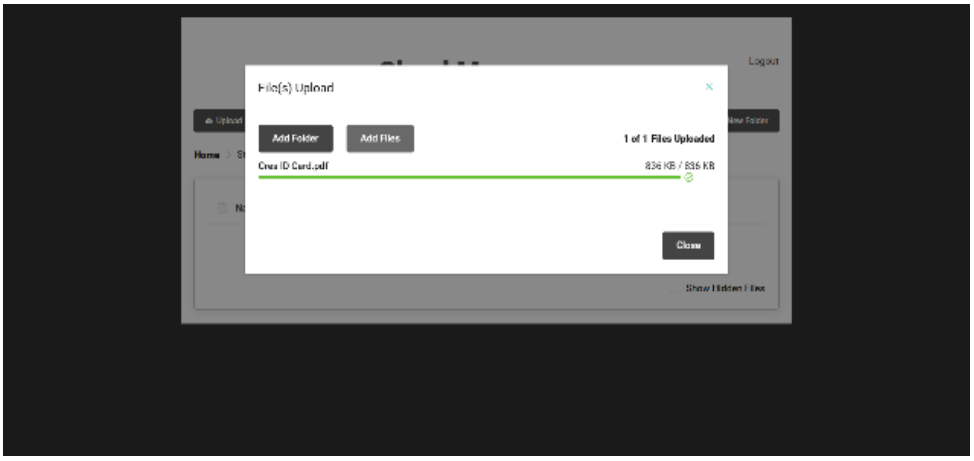

Fig. 5 Upload Page

The file upload window within a cloud management platform. It shows that a document has been uploaded successfully. The progress bar, marked in green, confirms the upload completion, and a small checkmark indicates everything went smoothly. The file size is displayed, both as the uploaded amount and the total size. Users can choose to add more files or folders using the "Add Files" or "Add Folder" buttons if needed. There's also a "Close" button available to exit this screen.

VII. CONCLUSION

The cloud storage system built using Raspberry Pi 5 provides an affordable and customizable solution for personal use. By using HTML, CSS, and PHP for the user interface, and JSON for efficient data handling, users can easily manage their files through a web-based platform. The integration of Apache server, DuckDNS, and Cloudflare ensures secure remote access, making the system accessible from anywhere. Throughout the implementation, the system has shown stable performance, reliable accessibility, and low resource consumption, proving its effectiveness as a practical self-hosted storage solution.

Moving forward, the system can be further enhanced by adding features like automatic data backups, multi- user support with customized access controls, and improved security measures. Additional optimizations in file transfer speeds and storage management can also be considered. Developing a mobile application for seamless file access would enhance usability. With these enhancements, the cloud storage system can offer a more advanced, scalable, and user-friendly solution for broader applications.

REFERENCES

[1] Dr. Girish N, Krupananda S, Jagadish P Patel, Kashifa Ruheen A, Khadija Tul Kubra, Assessing the Viability and Dependability of Next cloud Deployed on Raspberry Pi, International Journal of Engineering Research & Technology (IJERT), 2024.

[2] Noopur Malse, Aditya Lad, Atharv Mandpe, Nishant Lanjewar, Developing a Secure Private Cloud Storage System, International Journal for Research in Applied Science & Engineering Technology (IJRASET), 2024.

[3] Personal Cloud Storage using Raspberry PI, Rakesh Suryawanshi, Ruchita Amancha, Suhaib Sambulkhani, Vaishali Shinde, Priti Vyavahare, ResearchGate, 2023.

[4] Ms. N. Indhumadhi, Manikandan A and Balamurugan K, Cloud Storage Using Raspberry Pi, International Journal of Engineering Research & Technology (IJERT), 2023.

[5] Pratik Kumar singh, Prakhar Tibrewal, P j Mohammed shoaib, Naveen, Padmavathi M, Dr. T.C.Manjunath, A Raspberry Pi-based Private Cloud System for Remote Data Access, IJIRT, 2023.

[6] R Niyam, Dr. Gobi Natesan, Raspberry Pi as a Personal Cloud Server with Next-cloud, IJSRCSEIT, 2023.

[7] https://doi.org/10.1016/j.eswa.2023.120486

[8] Antonios Makris, Ioannis Kontopoulos, Evangelos Psomakelis, Stylianos Nektarios Xyalis, Theodoros Theodoropoulos, Konstantinos Tserpes, Performance Analysis of Storage Systems in Edge Computing Infrastructures, MDPI, 2022.

[9] https://www.raspberrypi.com/products/raspberry-pi-5/

[10] https://dash.cloudflare.com/0ed03acaa1cc74c7e401a1be9c8df61b/rasphub.duckdns.org

[11] https://www.duckdns.org/

[12] Timothy Sewe Ogode, M. Bhavana, P. Ananya, Dr. Vijay Kumar4, A Rasperry Pi Network Scanner & Cloud Storage, IJARCCE, 2021.

[13] S Prasath kumar, P Rayavel, N Anbarasi, Raspberry pi based secured cloud data, ResearchGate, 2021.

[14] https://doi.org/10.48550/arXiv.2107.14325

[15] Disari Chattopadhyay, Debasrita Ghosh, Abhilash Bhattacharya, Koushik Pal, Smart Home Automated System Using RaspberryPiand Google Cloud Services, (IJRASET), 2021.