RESEARCH ARTICLE OPEN ACCESS

Personal Cloud Storage System Using Docker Containerization

Mahek Bagwan*, Anjali Mane**, Avez Chadchan***, Shafi Hudewale****, Aishwarya Hosale*****

*(CSE AGPIT, Solapur, Email: mahekbagwan0106@gmail.com)
** (CSE, AGPIT, Solapur, Email: anjaliajitmane22@gmail.com)
***(CSE AGPIT, Solapur, Email: avezchadchan@gmail.com)
****(CSE, AGPIT, Solapur, Email: shafihudewale@gmail.com)
*****(CSE, AGPIT, Solapur, Email: aishwaryakeshi@gmail.com)

----------**********----------

Abstract:

Cloud storage services such as Google Drive and Dropbox provide easy access to files, but they also introduce concerns regarding data privacy, dependency on third-party providers, and recurring subscription costs. To overcome these limitations, personal cloud solutions have gained popularity, allowing users to store and manage data on their own infrastructure. This paper presents the design and implementation of a Personal Cloud Storage System using Docker containerization technology. The system is deployed using Docker containers to host the cloud application, database services, and web server environment in an isolated and scalable manner. Docker Compose is utilized to manage multi-container deployment efficiently. The proposed system provides features such as file upload/download, secure user authentication, file sharing, and access through a web-based interface. The implementation demonstrates that container-based deployment is lightweight, portable, and easy to maintain compared to traditional virtualization approaches. The developed personal cloud offers improved privacy, cost efficiency, and flexibility for individuals and small organizations. The results indicate successful deployment with stable performance and the ability to scale services based on user requirements.

Keywords: Docker, Personal Cloud, Containerization, Cloud Storage, Nextcloud, Docker Compose, Private Cloud.

----------**********----------

I. INTRODUCTION

Cloud computing has transformed the way individuals and organizations store and access data. Cloud storage services provide users with the ability to upload, manage, and retrieve data from anywhere using internet connectivity. Popular cloud platforms such as Google Drive, Microsoft OneDrive, and Dropbox are widely used due to their convenience, accessibility, and ease of collaboration. However, these services are owned and maintained by third-party organizations, which can raise concerns related to privacy, control, and data security.

In many cases, users store sensitive information such as personal documents, academic files, and organizational data on public cloud platforms. Although public cloud providers offer security measures, users still have limited control over where and how their data is stored. Additionally, most services provide limited free storage and require paid subscriptions for higher capacity. This creates a dependency on service providers and increases long-term costs.

A personal cloud is a private cloud storage system hosted on user-controlled infrastructure such as a personal computer, server, or Raspberry Pi. It enables users to store data locally while still providing remote access similar to public cloud platforms. Personal cloud systems are cost-effective, customizable, and privacy-focused.

Docker is a containerization platform that allows applications to run in lightweight containers. Unlike virtual machines, containers share the host operating system kernel, making them faster and more efficient.

Docker simplifies application deployment by packaging the application along with its dependencies, ensuring portability across different systems.

This project focuses on building a personal cloud storage system using Docker. The system is designed using Docker containers for modular deployment, enabling services such as file storage application, database management, and web access to work efficiently. The solution provides a secure, scalable, and easy-to-deploy personal cloud alternative.

II. PROBLEM STATEMENT

Most users rely on public cloud storage services for file management and remote access. These services introduce issues such as limited free storage, subscription costs, dependency on third-party providers, and privacy risks. There is a need for a personal cloud storage system that provides complete user control over data storage, improved privacy, and flexible deployment at minimal cost. This project aims to design and implement a Docker-based personal cloud storage system that provides secure access, file sharing, and efficient storage management.

III. OBJECTIVES

The main objectives of this project are as follows:

1. To design and develop a personal cloud storage system for private file management.
2. To deploy cloud services using Docker containers for lightweight and scalable deployment.
3. To implement user authentication and secure access to stored data.
4. To provide file upload, download, sharing, and deletion functionality.
5. To ensure persistent storage using Docker volumes.
6. To enable remote access using a browser-based interface.
7. To reduce dependency on public cloud providers while improving data privacy.

IV. PROPOSED SYSTEM

A. The proposed system is a Personal Cloud Storage solution deployed using Docker containers. The cloud application provides user access through a web interface where users can log in, upload files, download files, share documents, and manage stored data.

B. The system is composed of multiple components running inside Docker containers, ensuring isolation and modularity. The architecture includes:

- Cloud storage application container (Nextcloud / OwnCloud)
- Database container (MySQL / MariaDB)
- Web server container (Apache / Nginx, included in Nextcloud image)
- Persistent storage volumes for file storage and database data
- Docker Compose for orchestration and container management

C. The system allows multiple users to access the cloud environment, each having personal storage space and file management privileges. Data is stored on local storage while the user interface is accessible via browser.

V. SYSTEM ARCHITECTURE

The architecture of the proposed system is shown conceptually below:

User → Web Browser → Cloud Application (Nextcloud) → Database (MariaDB) + Storage Volume

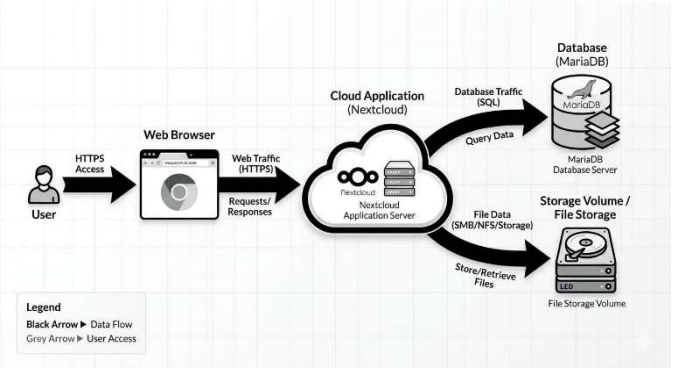

Fig. 1 Architecture of the system

A. Architecture Description

1. **Client/User:** Users access the system through a web browser or mobile interface.
2. **Docker Host:** The machine running Docker Engine hosts all the containers.
3. **Nextcloud Container:** Handles the application logic, user interface, file upload/download, and file sharing features.
4. **Database Container:** Stores user credentials, file metadata, permissions, and logs.
5. **Storage Volume:** Stores uploaded files persistently even if containers restart.
6. **Network Layer:** Docker provides internal networking between containers.

B. Working Flow

1. User sends request through browser.
2. Nextcloud container receives the request and verifies user authentication.
3. File operations are performed and file metadata is stored in the database.
4. Actual files are stored in persistent Docker volume.
5. Output is displayed on the browser interface.

VI. MODULE DESCRIPTION

The personal cloud system is divided into the following modules:

A. Authentication Module

This module verifies user identity using username and password. It ensures only authorized users can access the cloud storage. It supports admin creation and user management.

B. File Management Module

This module supports:

- Uploading files
- Downloading files
- Renaming files
- Deleting files
- Creating folders

C. File Sharing Module

Users can share files with other users or generate public links with permissions such as read-only or edit access.

D. Database Management Module

The database stores:

- User information
- File metadata
- Sharing permissions
- Activity logs

E. Docker Container Management Module

This module handles deployment and orchestration using Docker Compose. It ensures services start automatically and remain isolated.

VII. IMPLEMENTATION DETAILS

The project is implemented using Docker and Docker Compose on a Linux-based system.

A. Tools and Technologies Used

- Docker Engine
- Docker Compose
- Nextcloud Docker Image
- MariaDB Database
- Linux OS (Ubuntu)
- Browser (Chrome/Firefox)

B. Hardware Requirements

- Raspberry pi 3/4/5
- Internet Connectivity for remote access

C. Software Requirements

- Ubuntu/Linux OS or Windows with Docker Desktop
- Docker installed
- Docker Compose installed

D. Docker Compose File (Sample)

Below is a sample docker-compose.yml used for deployment:

```
version: '3'

services:
  db:
    image: mariadb
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: rootpass
      MYSQL_PASSWORD: userpass
      MYSQL_DATABASE: nextcloud
      MYSQL_USER: nextclouduser
    volumes:
      - db_data:/var/lib/mysql
```



```
  app:
    image: nextcloud
    restart: always
    ports:
      - "8080:80"
    environment:
      MYSQL_PASSWORD: userpass
      MYSQL_DATABASE: nextcloud
      MYSQL_USER: nextclouduser
      MYSQL_HOST: db
    volumes:
      - nextcloud_data:/var/www/html
    depends_on:
      - db

volumes:
  db_data:
nextcloud_data:
```

E. Steps for Deployment

1. Install Docker and Docker Compose.
2. Create a project folder.
3. Create the docker-compose.yml file.
4. Run the following command:
   docker-compose up -d
5. Open browser and access: http://localhost:8080
6. Create admin account and configure database.
7. Cloud storage becomes ready for use.

VIII. RESULTS AND DISCUSSION

The system was successfully deployed using Docker containers. The Nextcloud container provided a user-friendly web interface similar to Google Drive. The MariaDB container stored user and file metadata efficiently. Persistent storage volumes ensured that files remained available even after restarting containers.

A. Output Screens;

The following outputs were observed:

- Docker containers running successfully.
- Nextcloud login and dashboard interface.
- Successful file upload and download.
- File sharing link generation.
- Multi-user login support.

B. Performance Analysis

The Docker-based system performed efficiently with minimal resource usage. Since containers share the host kernel, startup time was fast. The modular architecture also allows easy updates and maintenance.

C. Security Features

The system supports:

- User authentication
- Access control
- Secure sharing permissions
- Database isolation

Additionally, SSL encryption can be enabled in future using Nginx reverse proxy and Let's Encrypt.

IX. CONCLUSION

This project successfully demonstrates the implementation of a Personal Cloud Storage System using Docker containerization. The system provides secure and private file storage with features such as file upload, download, sharing, and multi-user access through a web-based interface. Docker ensures lightweight deployment, portability, and easy maintenance compared to traditional virtual machine-based systems. The proposed personal cloud solution reduces dependency on third-party cloud providers while improving privacy and cost efficiency. The results confirm that Docker-based personal cloud deployment is a reliable and scalable approach for individuals and small organizations seeking secure storage solutions.

ACKNOWLEDGMENT

The authors would like to express sincere gratitude to Project Guide, for valuable guidance, continuous support, and technical suggestions throughout the completion of this project. The authors are also thankful to the **Department of Computer Engineering, [AGPIT,Solapur]**, for providing the

required facilities and resources to carry out this work. Finally, the authors would like to thank all faculty members and peers who contributed directly or indirectly towards the successful completion of this research work.

REFERENCES

[1] Docker Documentation. [Online]. Available: https://docs.docker.com
[2] Nextcloud Documentation. [Online]. Available: https://docs.nextcloud.com
[3] MariaDB Documentation. [Online]. Available: https://mariadb.org/documentation/
[4] "Cloud Computing: Concepts, Technology and Architecture," Thomas Erl, Prentice Hall, 2013.
[5] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality," Proc. IEEE, 2009.
[6] IEEE Research on Containerization and Virtualization Technologies, IEEE Xplore Digital Library.

