DATA DUPLICATION REMOVAL IN CLOUD COMPUTING BASED ON FILE CHECKSUM

James ADEGBOYE & Folahan JIBOKU

The Department of Computer Science,
Federal Polytechnic, Ilaro, Ogun State, Nigeria.
olujoba.adegboye@federalpolyilaro.edu.ng
folahan.jiboku@federalpolyilaro.edu.ng

Abstract

In recent times, data deduplication has gained attention and it is becoming well known in commercial storage systems. This is as a result of increased in digital content. It takes away superfluous data at from the file and sub-file level and contents that are being repeated by using the file checksum algorithm, which is found to be more efficient in terms of computation than classic compression methods in huge storage systems. Data deduplication is becoming more of a necessity for cloud storage providers as the size of data and the number of customers continue to grow at an exponential rate. Providers of cloud computing drastically cut their data storage and transfer costs by keeping a copy of repeated data. Addressing rising storage demands is a difficult and time-consuming process that necessitates a huge computational infrastructure for effective data processing and analysis. This work provides a breakdown of cloud computing, storage, and inclusion also considering cloud file services. It also considers deduplication storage efficiency by reviewing current data deduplication algorithms, implementations, and processes for the benefit of all stakeholders in cloud computing. It uses file checksum techniques to provide an efficient way of identifying and eliminating duplicates by computing file digest, which requires less time when compared with other approaches that have been used in the past.

Keywords: ***Data Deduplication, Cloud Computing, File checksum, Redundancy, Storage***

Introduction

A new generation of information and communication technology is gaining popularity which is known as the Cloud and all the services it provides. All users have some data to keep in a secure location that is easily accessible. The benefits of storing data in the cloud for sharing information with friends, transferring files between multiple smartphones, and backing up for small businesses cannot be exaggerated, as the cloud computing notion is essential to make life easier through the alternatives emanating with it (IDC 2010). Cloud computing helps several organizations in time adding suitability and cutting cost. Thus, the scope of cloud storage cannot be specifically bordered because these organizations can store their data without having to take cognizance of the gadgets. The key benefits of Cloud Computing to the end-users include cost savings, being able to access the data regardless of where they might be then, and lastly security. With more humans accessing their data online, cloud storage has become a significant aspect of file sharing today. (Bowers et al, 2009).

Due to this "data deluge, "In the big data era, figuring out how to manage storage cost-effectively has become one of the most difficult and critical challenges in mass storage systems. According to workload studies undertaken by Microsoft (Meyer and Bolosky, 2011), around 85% of the data in their production main and secondary storage systems are redundant, respectively (Wallace et al, 2012). It was discovered that about 80% of businesses said they were looking at data de-duplication technology in their storage systems to decrease redundant data and thereby increase or boost their overall performance (DuBois et al, 2011).

Duplication (FalconStor, 2009) is quite easy; when data is not stored as a backup, it is repeatedly duplicated; Duplication is not a novel concept; in fact, it is a subset of compressing data. Data de-duplication is a cost-effective data reduction technique that also saves storage space (El-Shimi et al, 2012;), it lowers data that has been repeated in situations whereby there are network issues (Shilane, 2012). In the broader sense, a chunk-level data (Quinlan and Dorward, 2011) de-duplication system divides the input stream of data (for example, backup files and images of a virtual machine, and so on) into data "chunks" that are well duplicate-detected and independently recognized by a well-secured platform for an example the SHA-1 also known as a fingerprint (Muthitacharoen et al, 2001). These chunks of data may have a size that cannot be changed, like file blocks, or variable-sized units defined by the content

itself (Quinlan and Dorward, 2011). Typical compression algorithms have been known to operate at the byte or string level, de-duplication discovers and clears redundancy (e.g., 8KB) or file level. Second, de-duplication bypasses the traditional approach of byte-by-byte comparisons by constructing fingerprints to identify redundant material (files or chunks). By calculating and indexing the fingerprints of chunks or files, Deduplication may easily be employed for global data minimization in commercial storage systems because of these two features. Since de-duplication fingerprints are orders of magnitude less than the original data. Several approaches to de-duplication include file checksum, parity bit, repetition code, cyclic redundancy checks, etc. The Verhoeff algorithm and Damm algorithm are all checksum schemes that are specifically designed to detect errors made by humans when writing down or remembering important digits such as identification numbers. The file checksum concept evaluates if a system has redundant data by making comparisons with an entering file chunk with previously-stored files using several file properties such as User ID, Size, Extension, FileName, data parameters, and Checksum. The purpose of this research is to develop a system that detects and removes duplicate data on the cloud using the file checksum algorithm. To achieve this, existing systems were examined to check for their weaknesses, and then a data duplicate and removal detector was developed using a file checksum algorithm.

2.0 LITERATURE REVIEW

A computer data storage concept in which digital information is organized into logical pools is referred to as cloud storage. The physical storage is often owned and maintained by a hosting business, and it spans numerous servers (possibly in various countries) (Wikipedia, 2022). These cloud storage providers are responsible for keeping data secure and available, as well as the environment safe and functional, and effective. People and corporations buy or lease storage capacity from vendors to store user, organization, or application data. Organizations will need to connect how they use and manage their data from the beginning to the end of mankind due to the rapid expansion of data and the requirement to keep it safer and longer. Now we have the option of storing all our data on the internet. Through the Internet, other companies offer and manage that off-site storage. Cloud storage provides access to a big pool of storage with three key features: Web-based access, a connection that isn't stable, and rapid existence of extremely large amounts of storage, all while charging only for what you use.

2.1 CLOUD STORAGE EVOLUTION

A type of cloud computing service that uses standard network storage and hosted storage to provide storage is referred to as cloud storage. One of the numerous benefits of cloud storage is being able to access your data from anywhere and at any time (Osuolale, 2019). Cloud storage providers may store anything from a tiny bit of data to an entire organization's warehouse. Cloud storage providers can charge subscribers based on how much storage they use and how much data they move to the cloud. Cloud storage is a concept hidden behind a user interface. On-demand access to the storage is possible. Cloud storage capacity is scattered around the globe. Cloud storage design enables scalable and multi-tenant delivery of on-demand services.

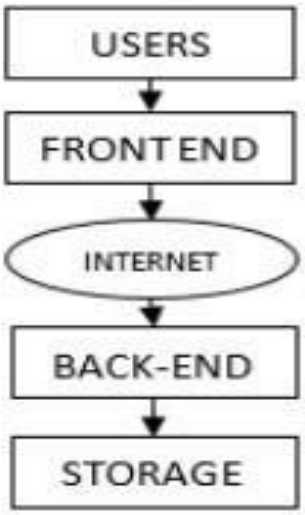

***Figure 1*. Cloud Storage Architecture (Stephen, 2007).**

2.2 ADVANTAGES OF CLOUD STORAGE

Compatibility: All of the cloud storage providers we've looked at so far feature desktop folders on all of their platforms. Users may move files between their local storage and the cloud using this method.
Bandwidth: Instead of emailing files to someone, you may send a web link to them via email.
Accessibility: Files stored on the server may be viewed from any computer with an Internet connection.
Recovering from disaster: It is strongly advised that firms have a backup strategy in place at all times. Businesses may utilize cloud storage as a backup strategy by storing a second copy of crucial files. These files are kept on a distant server and may be viewed using an internet connection. (Osuolale, 2019).

2.3 DISADVANTAGES OF CLOUD STORAGE

Storage Utilization: Be careful while moving files into a cloud folder. Your work will be relocated to the cloud storage location from its original folder permanently.
Bandwidth: Many cloud storage providers have a set amount of bandwidth available. If an organization exceeds the allotted budget, extra fees may be incurred.
Accessibility: It is impossible to view your data if you don't have a stable connection to the internet.
Security: Due to the relativity and the sensitivity of data, so many stakeholders find it difficult to trust vendors to keep their data safe from intruders.
Software: Without the software, it is impossible to edit files on all fronts (Osuolale, 2019).

2.4 OVERVIEW OF DATA DEDUPLICATION

Any cloud-based data security solution, for instance, must be willing and able to provide services at a cheaper rate to intending customers compared to what they can sum up or create on their own. Data deduplication among numerous clients is one way being utilized to achieve this aim, with the expenses of offering the service divided among the number of paying clients. Because there are several ways for eliminating duplicate data, the variations in the solutions, as well as the influence they may have on security and the rate at which data is being transmitted must be understood by service providers and their consumers.

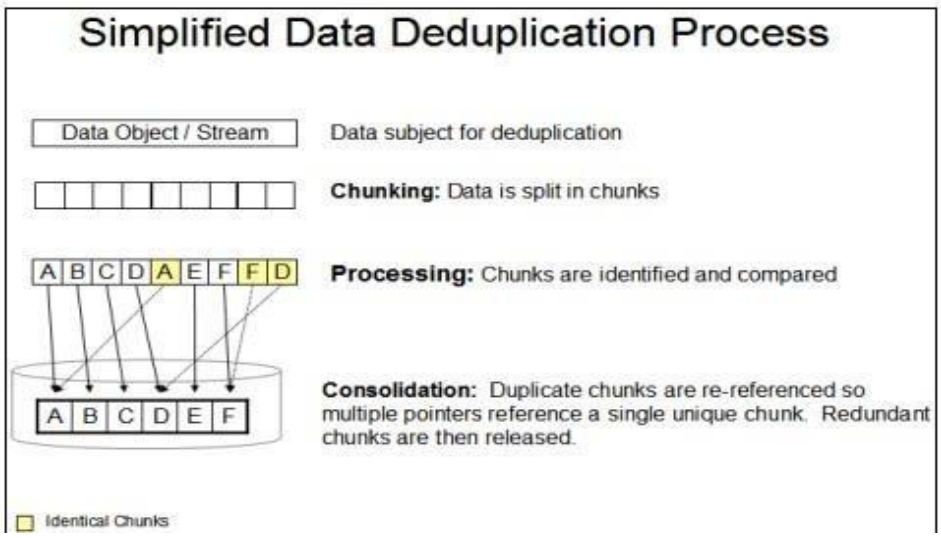

Figure 2. **Deduplication processing (Source: Osuna et al, 2011).**

2.5 CHUNKING OF DATA STREAM

A technique for identifying redundancies by separating data into predetermined pieces can be referred to as chunking. It depends on the deduplication process's technology and location; these units might be files or more detailed

elements like blocks. When compared to other ways, file-level deduplication is less adaptable. If the deduplication system is aware of the features, it may be able to distinguish specific components inside specific files. The many ways in which data can be chunked are listed in the table below. The data deduplication ratio is affected by each approach.

TABLE 1
***Data Chunking Approaches (Compiled by Author*)**

File-based	Each chunk is made up of a single file. Typically, file-based deduplication is employed with any gadget that is capable of seeing file systems
Block-based	The information item is being broken down into smaller pieces of varying sizes. Block-based Block storage devices are the most common users of deduplication.
Format aware	This technique recognizes specified data formats and chunks of data objects based on them. Format aware chunking, for instance, might split a PowerPoint presentation into many slides.
Format agnostic	This technique is a search algorithm that searches for logical gaps and other patterns in data items that tend to correlate.

Data deduplication algorithms are often used in different computer system contexts such as the following ;

Secondary storage: Secondary storage systems which include archive storage and backup technology, have a lot of duplicates (Wallace et al, 2012). Data domain, a storage business that is now a part of EMC, believes that "disk-based deduplication storage has emerged as the new-generation storage solution for enterprise data protection to replace tape libraries" based on this discovery(Zhu et al, 2008). For a fact, data deduplication is already proven to reach a factor of data reduction of approximately 5~40 in such systems (Shilane et al, 2012), resulting in considerable storage space and hardware cost reductions. Post-deduplication delta reduction has recently been used as a complement to data deduplication to reduce non-duplicate but equivalent data chunks. Such post-deduplication procedures have been shown to achieve an extra data parameter of 2.5 by imposing additional processing and I/O overheads on top of data deduplication (Xia et al, 2014).

Primary Storage: Data deduplication is capable of producing a data reduction factor which sums up to about 40%-60% in main storage (Meyer et al, 2011), notably for server file systems, according to recent research (Koller et al, 2010). Data deduplication for main storage not only decreases the space used in storing file requirements but also removes identical I/O on the important I/O path (Koller et al, 2010), improving disk I/O performance. Deduplication has recently been included in various file systems that are open source for main storage such as OpenDedupe, (Koutoupis, 2011).

Cloud Storage: Cloud storage: In recent times, the whole premise of the cloud has become an essential platform used for storage in computers (Vrable, 2009). Because the fundamental performance constraint of cloud storage is restricted network capacity in the wide-area network, data deduplication can assist speed up data synchronization between client and cloud by detecting data that is unmodified. Meanwhile, deduplication assists in reducing cloud storage expenditure. Data deduplication is now used by DropBox, SkyDrive (formerly known as OneDrive), Google Drive, and other cloud storage services to improve their services (Drago et al, 2013).

3.0 METHODOLOGY

3.1 ALGORITHM FOR THE SYSTEM

The file checksum technique is used in this project to gauge the file's hash value as the primary prerequisite for removing duplicates. The method takes a random message as input and returns a 128-bit "message digest" of it. It is

considered to be practically impossible to produce two messages with a matching message digest, or a message with a pre-specified goal message digest. The MD5 algorithm was created for digital signature systems where a large file must be safely "compressed" before being encrypted with a private (secret) key using a public-key cryptosystem.

3.2 FILE CHECKSUM

Ways to assess hashes and checksum to eliminate duplicates from a selection were demonstrated in File Checksum. Data duplication removal is examining storage devices, such as a hard drive or a server, seeking duplicate data, and selectively removing them. While the following material discusses picture files, any file type on a computer may be opened using the same approach. This is because picture files take up a lot of space, and being able to remove all but one instance might drastically reduce the amount of storage required, especially in circumstances where there isn't a stringent file handling protocol in place when adding files. The process of using an algorithm to confirm the validity of a computer file, which is included in the file checksum, is known as file verification.

3.3 FILE VERIFICATION

The process of using an algorithm to confirm the validity of a computer file, which is included in the file checksum, is known as file verification. This may be implored by intermittently comparing two files, but it requires two copies of the same file and may ignore systemic errors that impact both files (Morris 2015). A file's integrity can be endangered, causing the file to become corrupted. A file can be destroyed in a variety of ways, including bad storage media, transmission problems, copying or relocation errors, and software flaws. Comparing the hash value of a file to a previously determined value, verification that is hash-based gives the assurance that the file is free from errors. The file is assumed to be unaltered if there's a match in the value. ".sha1" extension denotes a checksum file in sha1sum format that contains 160-bit SHA-1 hashes. A checksum file with a ".md5" extension, contains 128-bit MD5 hashes in md5sum format. A checksum file with the suffix ".sfv" contains 32-bit CRC32 checksums in a basic file verification method.

3.4 CHECKSUM ALGORITHM

Parity byte (also known as parity word): The most basic checksum technique is the check for parity over time, it breaks data into n-bit "words"each, and The exclusive or (XOR) of all the other terms is then computed. As an additional word, the result is attached to the message. The recipient evaluates the exclusive of all the message's words, including the checksum, to assess the message's integrity; if the output is not a term consisting of n zeros, the recipient knows there was an error in transmission. Any transmission fault that flips a small piece of the message, or an odd number of bits, is identified as an invalid checksum by this checksum. The probability of a two-bit error being undetected is 1/n if the damaged bits are picked at random.

Modular Sum: The checksum is computed by adding all the "words" as numbers that are unassigned, eliminating any extra bits, and multiplying the total by the complement of the two. The recipient repeats the process, including adding all of the words in the same sequence, to verify the message. An error has occurred if the checksum does not return a word full of zeros. This version also detects single-bit faults, but not multiple-bit faults.

Message Digest: MD5 File Verification is a program that can generate a checksum file. Checksums will be utilized. The main purpose is to collect all checksums for the collection of images from where we would like to erase repetitions, sort them by checksum values, and then rapidly locate any duplicates using a spreadsheet calculation technique. We may use the spreadsheet as a checklist to delete duplicate files if there are any.

Position-dependence: The above-mentioned basic checksums do not mention several typical flaws that impact several bits in one go, such as altering the order of data word ings or adding or removing words with all bits set to zero. Cyclic redundancy check and Fletcher's checksum are among the most widely used checksum algorithms, they address these flaws by taking into account the value of each word and its location in the series. The cost of computing the checksum rises as a result of this feature.

3.5 MD5 CHECKSUM ALGORITHM

This algorithm also known as the algorithm of MD5 message-digest collects messages of any length as input and creates a 128-bit message digest.

It is thought to be computationally impossible to produce two messages with a matching message digest, or any message that the message digest has known beforehand.

We begin by assuming that we have an n-bit message and that we wish to extract its message digest. n is a non-negative arbitrary integer that is capable of being "0", not a multiple of "8", and be any size. We picture the following parts of the message being written down: m_0 m_1 ... m_{n-1}

To calculate the message digest, complete the following five steps.

Step 1: Attach the Padding Bits

The message is "stretched" (padded) to a length of 448 modulo 512 bits. In other words, the message is just 64 bits long, falling short of being a multiple of 512 bits. Padding is often executed, even if the length of the message is already equal to 448, modulo 512. Padding is accomplished by attaching a single "1" bit to the message, followed by "0" bits until the length of the stretched message in bits equals 448 modulo 512bits. A total of 512 bits is added, with a minimum and maximum of 1 bit.

Step 2: Add the Length
A 64-bit representation of b is added to the result of the previous step. If b is larger than 264 in this case that it is, just the low-order 64 bits of b are used. in which these bits are joined as two 32-bit words, low order word first, as is customary. At this point, the final message's length (after padding with bits and b) is an exact multiple of 512 bits. To put it another way, the message's length is a multiple of 16 (32-bit) words. M[0... N-1], noting that N is a multiple of 16.

Step 3. Initialize MD Buffer
The message digest is computed using a four-word buffer (A,B,C,D). Each of the registers A, B, C, and D is a 32-bit register. In hexadecimal, these registers are assigned to the following values (low-order bytes first):
Word A: [01] [23] [45] [67];
Word B: [89] [ab] [cd] [ef] ;
Word C: [fe] [dc] [ba] [98] ;
Word D: [76] [54] [32] [10];

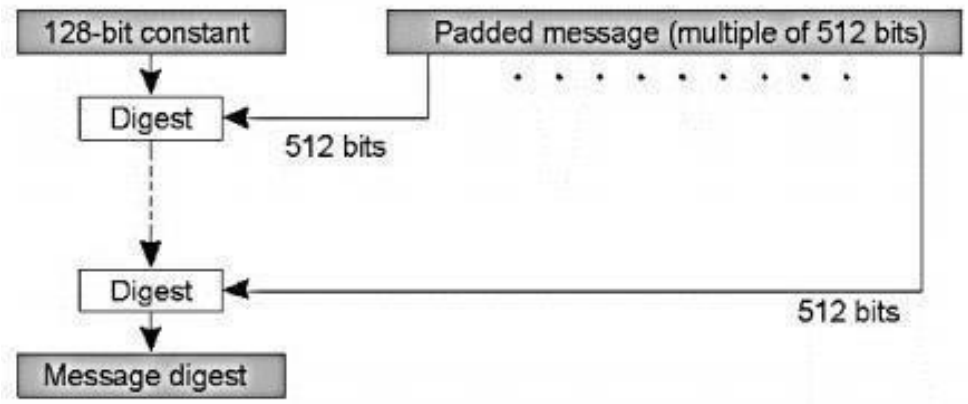

***Figure 3*: The Structure of the MD5 Algorithm (Source: Rivest)**

3.6 Architecture of the System

A comprehensive architectural view of the system, illustrating many of the system's elements from several architectural views. Its goal is to capture and discuss the key design decisions made by the system. It also focuses on the system characteristics, procedures, and techniques.

The User Interface layer submits requests to the Business Layer through a central interface which is the cloud server, this layer manages the request according to its logic guidelines like searching, sharing and the de-duplication process, and then captures the data returned from the database into the class format and makes it accessible to the User Interface Layer. The admin is capable of seeing all users and the activities they perform on the server.

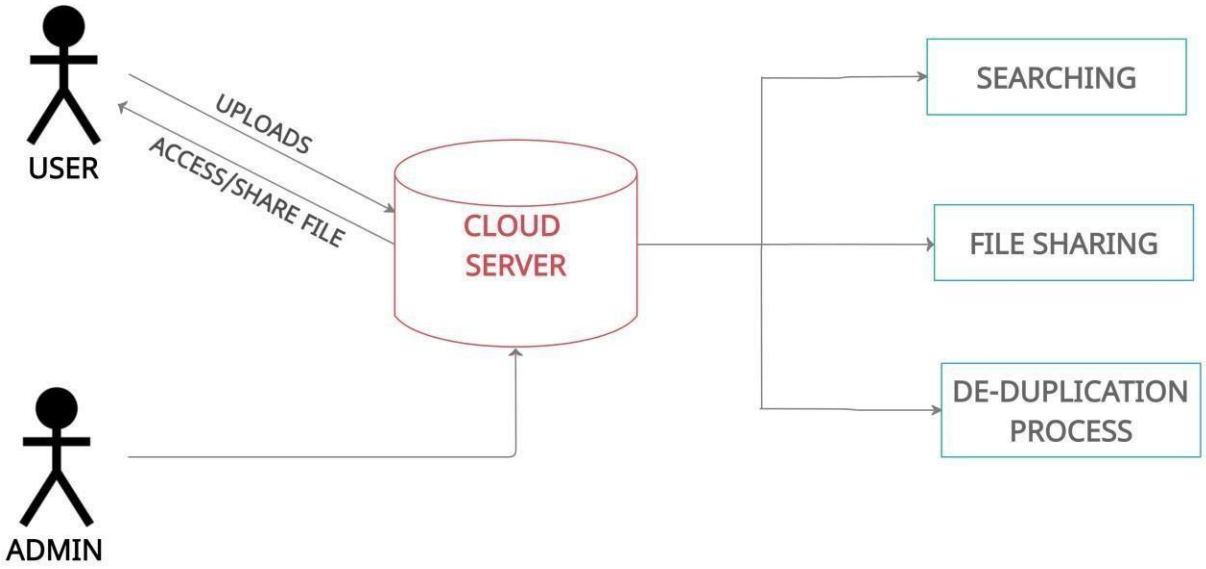

***Figure 4*. Overall architecture of the system**

3.7 DESIGN OF THE INPUT

The data that should be supplied into the system serves as a basis for the intended output of the system. The new system's input is designed to gather data from the subsystem.

3.8 DESIGN OF THE OUTPUT

When identifying the input design, it's equally crucial to include the output design. The output is the data that the system produces after the input data has been entered into the system. The ".md5" file extension denotes a checksum file in md5sum format that contains 128-bit MD5 hashes.

3.9 FILE DESIGN

Database Design

The database architecture for the website is made up of three tables, one is utilized to store Binary Large Objects. If the files are saved in the file system, the field can be erased. There are three fields; one for the file's main details, another for file structure utilizing folders, and a third for users and file sharing within that group.

Table 2: User Table

S/N	FIELD	FIELD TYPE	FIELD SIZE
1	ID	INT	11
2	EMAIL	VARCHAR	50
3	PASSWORD	VARCHAR	200
4	NAME	VARCHAR	50
5	ACTIVE_STATUS	TINY INT	1

Table 3*: File Table*

S/N	FIELD NAME	FIELD TYPE	FIELD SIZE
1	ID	INT	11
2	USER_ID	VARCHAR	50
3	FILE_ID	TEXT	256

Table 4: *File Uploads*

S/N	*FIELD*	*FIELD TYPE*	*FIELD SIZE*
1	*ID*	*INT*	*11*
2	*FILE*	*VARCHAR*	*50*
3	*HASH*	*TEXT*	*256*
4	*BLOB*	*TEXT*	*NULL*
6	*ACTIVE_STATUS*	*TINY INT*	*1*

4.0 RESULTS

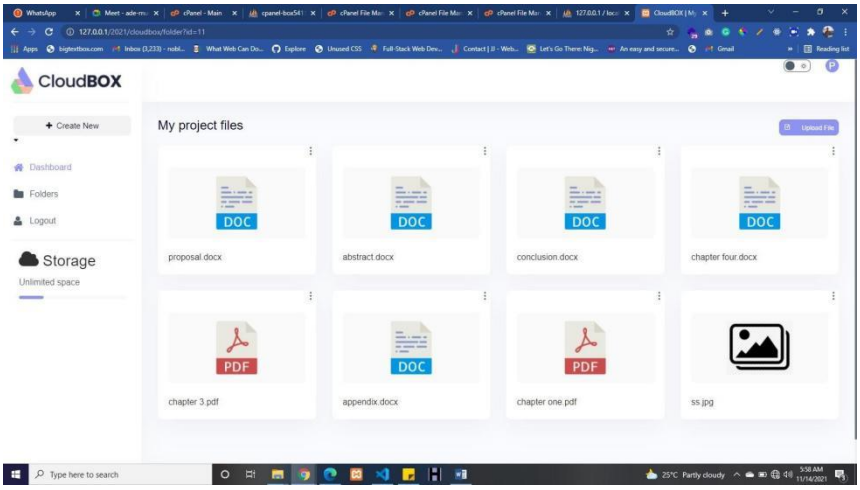

Figure 5. **Directory Files Page**

Figure 5 above shows a grid of files in a particular directory. Users can click on this file to see the full details of the file, as well as a share and downloadable link for the file. On this page, the user can also move a file from one folder to another folder. Additionally, clicking on the Upload File button will bring a dialog for uploading the file.

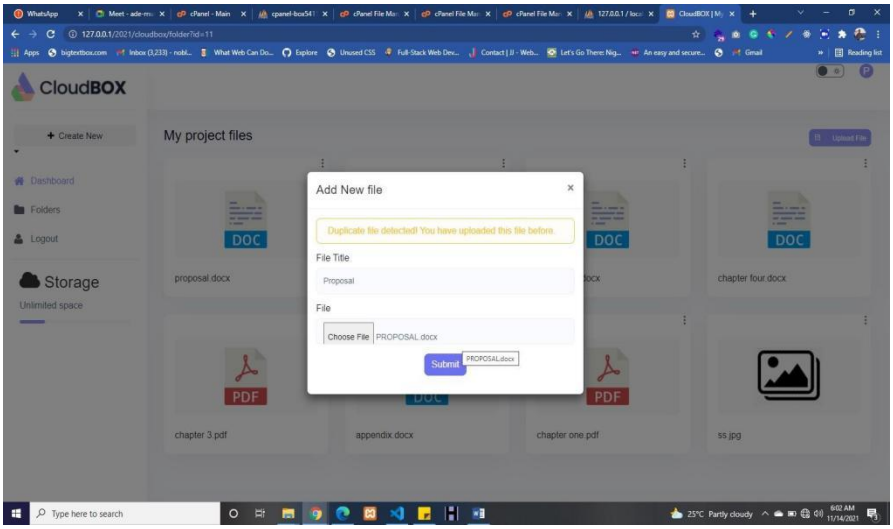

Figure 6: **File Upload Dialog and Duplicate Checker**

Figure 6 depicts the dialog box for uploading files to a folder, as well as the prompt that occurs when a user attempts to submit an already-existing document (duplicate file); this system will halt the action and provide the user with feedback.

5.0 CONCLUSION

The creation of a web application that uses file checksums to perform all activities needed in erasing cloud data duplicates. This system that has been built can help businesses that deal with highly redundant activities that require standard data copying and storing for future reference or recovery. It allows businesses to preserve data often and provides reliable, fast, and cost-effective data recovery solutions. Creating a backup for a file every 5 days generates a lot of duplicates and takes up a lot of space. A file checksum analysis can be used to eliminate duplicate data groups and maintain only what is unique and vital, freeing up a lot of storage space. The main concern will be ensuring that the database is free of duplicate files. The system developed is of great educational value since it developed a unique method for effectively locating redundant data in cloud computing, exchanging data, and erasing data duplicates using file checksums. Based on the issues faced in the course of this research, it is now recommended that interested web developers with competent records conduct more succinct work to simplify the system and expand the database so that it can hold as much data as feasible. Replicated data should be erased to free up space for a more dependable system.

REFERENCES

Bowers, K. D., Juels, A., & Oprea, A. (2009, November). HAIL: A high-availability and integrity layer for cloud storage. *In Proceedings of the 16th ACM conference on Computer and communications security* (pp. 187-198).

Drago, I., Bocchi, E., Mellia, M., Slatman, H., & Pras, A. (2013). Benchmarking personal cloud storage. *Proceedings of the 2013 Conference on Internet Measurement Conference*. https://doi.org/10.1145/2504730.2504762

DuBois L., Amaldas M., and Sheppard E., "*Key considerations as deduplication evolve into primary storage," White Paper 223310*, Mar. 2011. [Online]. Available: http://www.bedrock-tech.com/wp-content/ uploads/2010/05/wp_key-considerations.pdf

El-Shimi A., Kalach R., and Kumar A., "Primary data deduplication-large scale study and system design," *in Proc. Conf. USENIX Annu. Tech. Conf.*, Jun. 2012, pp. 1–12.

FalconStor Software, Inc. 2009. Demystifying Data Reduplication: *Choosing the Best Solution. http://www. pexpo.co.uk/c content download/20646/3 53 74 7/fi le/DemystifyingDataDedupe* _ W P. pdf, White Paper, 2009-10-14,1-4.

IDC. (2010). The 2011 digital universe study. Retrieved 21 April 2022 from http://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf.

Koller, R., & Rangaswami, R. (2011). I/O Deduplication. ACM Transactions on Storage, 6(3), 1– 26. https://doi.org/10.1145/1837915.1837921

Koutoupis, P. (2011). Data deduplication with Linux. Linux Journal, 2011(207), 7.

Meyer, D. T., & Bolosky, W. J. (2011). A study of practical deduplication. *In Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (pp. 229–241).

Muthitacharoen, A., Chen, B., & Mazières, D. (2001). A low-bandwidth network file system. *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles*. https://doi.org/10.1145/502034.502052

Osuna, A., Balogh, E., de Carvalho, A. R. G., Javier, R. F., & Mann, Z. (2011). *Implementing IBM storage data deduplication solutions*. IBM Redbooks.

Osuolale A. (2019). Data Finding, Sharing, and Duplication Removal in the Cloud Using File Checksum Algorithm. *International Journal of Research Studies in Computer Science and Engineering (IJRSCSE) Volume 6, Issue 1, 2019*, PP 26-47 ISSN 2349-4840 (Print) & ISSN 2349-4859 (Online) DOI: http://dx.doi.org/10.20431/2349-4859.0601004

Quinlan S. and Dorward S., "Venti: A new approach to archival storage," i*n Proc. USENIX Conf. File Storage Technol.*, Jan. 2002, pp. 1–13

Rivest, R. (1992). RFC1321: The MD5 message-digest algorithm. Retrieved 23 April 2022 from http://www.ietf.org/rfc/rfc321.txt

Shilane, P., Huang, M., Wallace, G., & Hsu, W. (2012). WAN-optimized *replication of backup datasets using stream-informed delta compression. ACM Transactions on Storage*, 8(4), 1– 26. https://doi.org/10.1145/2385603.2385606

Stephen J. Bigelow, 2007 Data Deduplication Explained: Retrieved 4 May 2022 from http://searchgate.org;

The definition of Cloud storage retrieved from: https://en.wikipedia.org/wiki/Cloud_storage Accessed 26 May 2022

Vrable, M., Savage, S., & Voelker, G. M. (2009). *Cumulus. ACM Transactions on Storage,* 5(4), 1–28. https://doi.org/10.1145/1629080.1629084

Wallace, G., Douglis, F., Qian, H., Shilane, P., Smaldone, S., Chamness, M., & Hsu, W. (2012). Characteristics of backup workloads in production systems. *In Proceedings of the USENIX Conference on File and Storage Technologies (FAST)* (Vol. 12, pp. 4-4).

Xia, W., Jiang, H., Feng, D., Douglis, F., Shilane, P., Hua, Y., … Zhou, Y. (2016). A Comprehensive Study of the Past, Present, and Future of Data Deduplication. *Proceedings of the IEEE*, 104(9), 1681–1710. https://doi.org/10.1109/jproc.2016.2571298

Zhu, H., Wen, Y., & Ng, W. K. (2012). Private data deduplication protocols in cloud storage. *Proceedings of the 27th Annual ACM Symposium on Applied Computing - SAC '12.* https://doi.org/10.1145/2245276.2245361