

DATA DOWNLOAD DUPLICATION ALERT SYSTEM

Miss. Pratiksha S. Hiwrale¹, Dr. Sarkarsinha H. Rajput², Dr. Shital A. Patil³,

Asst. Prof. Arsalan A. Shaikh⁴

Research Scholar , Department of Computer Applications, SSBT's COET, Jalgaon, Maharashtra, India¹

Asso. Professor, Department of Computer Engineering, SSBT's COET, Jalgaon, Maharashtra, India^{2,3}

Asst. Professor, Department of Computer Applications , SSBT's COET, Jalgaon, Maharashtra, India⁴

Abstract: The **Data Download Duplication Alert System** is a software system developed to help users avoid downloading the same file multiple times. Today, people frequently download documents, images, videos, software, and other files from websites, cloud platforms, and online repositories. Sometimes, the same file is downloaded again without the user realizing it. This can take up extra storage space, use more internet bandwidth, and waste time. The main purpose of this research is to develop a system that can check whether a file has already been downloaded and notify the user before downloading it again. The system keeps track of previously downloaded files and checks information such as the **file name, size, type, and hash value**. If the requested file is found to be a duplicate, the system displays an alert to the user. The user can then choose whether to continue with the download or cancel it. The system can be developed using **Python, databases, and file hashing techniques**. By detecting duplicate files, the system can help reduce unnecessary downloads, save storage space, and decrease network bandwidth usage. It can also make file management easier for users. The proposed system can be useful for **individual users, educational institutions, and organizations**. In the future, it can be improved by adding cloud storage integration, smarter duplicate detection, real-time monitoring, and automatic file management features.

I. INTRODUCTION

Digital technology is growing rapidly, and as a result, people are creating, sharing, and downloading more data than ever before. Every day, users download different types of files such as documents, images, videos, software, and datasets from websites, cloud storage, educational portals, and organizational platforms.

Although downloading files is a normal part of using digital technology, users sometimes download the same file more than once without realizing it. This may happen because they forget that they have already downloaded the file, files are not properly organized, or the same file is requested multiple times. Repeated downloads can take up unnecessary storage space, use additional internet bandwidth, and make file management more difficult.

To address this problem, the Data Download Duplication Alert System is proposed. The main purpose of this system is to check whether a file has already been downloaded and inform the user before downloading it again. The system keeps a record of previously downloaded files and compares new download requests with these records.

The system can check different file details, such as the file name, file size, file type, and hash value, to identify duplicate files. Hash-based comparison is particularly useful because it can identify files with the same content even when they have different names.

This problem is important because individuals, educational institutions, offices, and organizations increasingly depend on digital storage and internet-based data transfer. Repeated downloads can use unnecessary storage space and internet data. In larger environments, duplicate files can also increase storage requirements and make it harder to organize and manage information.

The proposed system provides a simple solution by displaying an alert when a duplicate download is detected. The user can then decide whether to continue downloading the file or cancel the download. This gives users better control over their downloaded data and helps reduce unnecessary duplication.

The main motivation behind this research is to develop a simple and practical system for improving digital file management. The system focuses on detecting duplicate files, providing timely alerts, reducing unnecessary downloads, and making better use of storage and network resources. The system can be used in personal computers, web applications, educational platforms, organizational systems, and cloud-based environments. The main objective of this research is to design and develop a Data Download Duplication Alert System that can identify previously downloaded files and notify users before a duplicate download occurs. This can help save storage space, reduce unnecessary bandwidth usage, and improve overall digital data management.

II. LITERATURE SURVEY

Data duplication has become a common problem as the amount of digital information continues to grow. People and organizations store and download a large number of files using computers, cloud storage, and online platforms. When the same file is stored or downloaded multiple times, it uses extra storage space, increases network traffic, and makes data management more difficult. Because of this, researchers have developed different methods to identify duplicate data and keep only the required copies. These studies provide a useful foundation for developing a system that can detect duplicate downloads and alert users before the same file is downloaded again.

Paulo and Pereira (2014) conducted a detailed study of different storage duplication systems. They classified duplication methods based on factors such as the level at which duplication is detected, where the data is located, when the duplication check is performed, and the technique used. Their research showed that data duplication methods can be used in many areas, including backup systems, archives, primary storage, and solid-state drives. The study also showed that there is no single duplication method suitable for every situation. The appropriate method depends on the type of storage system and its requirements. [1]

Xia et al. (2016) studied the development of data duplication techniques and discussed their use in the past, present, and future. Their research explained that duplicate data can be detected at the complete-file level or by dividing a file into smaller parts. Hash functions are commonly used to create a unique fingerprint for a file. This fingerprint can then be compared with fingerprints of previously stored files to identify duplicate content. This approach is useful for the proposed system because it can identify files with the same content even if their names are different. However, generating hash values for a large number of files can require additional processing time. [2]

Singh and Kumar (2017) discussed the increasing need for storage space and explained how duplicate files can waste storage resources. Their study described different methods for detecting duplicate data at both file and chunk levels. It also explained the role of hashing in the duplication process. Their research supports the idea that detecting duplicate data before storing or downloading another copy can help improve the use of storage and network resources. [3]

Another area of research focuses on file chunking. In this method, a large file is divided into smaller parts called chunks. These chunks can then be compared using hash functions to identify repeated content. The study *Data Duplication Approaches: Concepts, Strategies, and Challenges* discusses different chunking methods, including fixed-size and variable-size chunking. Variable-size or content-defined chunking can be useful when a file has been slightly modified because the system can still identify the unchanged parts of the file. [4]

Mokhtar et al. (2017) proposed a content-defined chunking method for cloud storage and evaluated different chunking approaches. Their research showed that the method used to divide a file into chunks can affect the performance of duplicate detection. Content-defined chunking can be useful for large files because it can identify similar parts even when some changes have been made to the file. However, this method can require more processing compared with simple whole-file comparison. [4]

Recent research has also focused on improving the speed of duplicate detection. Studies on parallel processing have shown that chunking files and generating fingerprints can require a significant amount of processing. Using parallel processing can increase the speed of these operations while maintaining a similar duplication detection rate. This is important for a real-time duplicate alert system, especially when users are downloading large files.

Security and privacy are also important when dealing with duplicate data. Di Pietro and Sorniotti (2016) studied security issues in data duplication systems and proposed a proof-of-ownership method. Their research showed that simply comparing file hashes in a shared environment can create security and privacy concerns. Therefore, a practical duplicate-download system should store file information and hash values securely and should prevent unauthorized access to this information. [7]

Recent research continues to explore efficient duplication techniques for cloud-based environments. A 2025 review described the main stages of data duplication as chunking, fingerprinting, indexing, and writing. The study highlighted the importance of indexing because it helps the system quickly determine whether duplicate content already exists. It also explained that data duplication can help reduce storage requirements, network data transfer, and energy consumption. [9]

Another recent study compared different content-defined chunking algorithms using realistic datasets. The researchers examined factors such as processing speed, duplication ratio, average chunk size, and variation in chunk size. The findings showed that different algorithms have different advantages and limitations. Therefore, no single duplication technique is suitable for every type of application.

Overall, the existing research shows that hashing, fingerprinting, chunking, and indexing are useful techniques for identifying duplicate data. These studies provide a strong foundation for the proposed Data Download Duplication Alert System. However, the proposed system focuses on applying these techniques in a simpler and more user-friendly way to identify previously downloaded files and alert users before unnecessary duplicate downloads occur.

Critical Analysis and Research Gap

The literature review shows that techniques such as hashing, fingerprinting, indexing, and chunking are commonly used to identify duplicate data. These techniques can help reduce repeated data and save storage space. However, most existing studies mainly focus on detecting and removing duplicate data at the storage or cloud level.

There is less focus on developing a simple system that can warn users before they download a file that they have already downloaded. Many existing systems use advanced methods such as complex chunking, large indexing structures, distributed storage, and security mechanisms. These methods can be useful for large-scale systems, but they may be unnecessarily complicated for individual users and small organizations.

This creates an opportunity to develop a simple, lightweight, and user-friendly Data Download Duplication Alert System. The proposed system can use basic file information along with hash values to check whether a requested file has already been downloaded.

Instead of automatically deleting or blocking duplicate files, the proposed system will display an alert to the user when a duplicate is detected. The user can then choose whether to continue with the download or cancel it. This gives the user more control over the download process.

The proposed approach applies the existing concepts of data duplication in a practical way by adding a preventive alert mechanism. The main goal is to help users avoid unnecessary downloads, save storage space, reduce unnecessary internet bandwidth usage, and make file management easier.

III. METHODOLOGY

1. Research Design

This research follows an experimental and system-development approach. The main aim is to develop and test a Data Download Duplication Alert System that can check whether a file has already been downloaded before downloading it again.

The system is developed in different stages. These stages include requirement analysis, system design, implementation, testing, and performance evaluation.

The proposed system uses file information such as the file name, file size, file type, and hash value to identify duplicate files. When a user requests a file for download, the system compares the file information with the records of previously downloaded files.

If the system finds that the file has already been downloaded, it displays an alert message to the user. The user can then decide whether to cancel the download or continue downloading the file.

The system is tested using different types and sizes of files to check its accuracy and performance in detecting duplicate downloads.

2. Data Collection Methods

This research uses both primary and secondary data.

Primary Data

Primary data is collected through practical experiments performed on the proposed system. Different types of files, such as PDF documents, images, text files, videos, and other commonly used files, are used for testing.

The same files are downloaded more than once to check whether the system can correctly identify duplicate files and display an alert. Different file sizes and formats are also used to evaluate how the system performs under different conditions.

The results obtained from these experiments are used to evaluate the accuracy and performance of the proposed system.

Secondary Data

Secondary data is collected from research papers, journal articles, technical publications, and previous studies related to data duplication and duplicate file detection.

Topics such as file hashing, duplicate detection, cloud storage, and storage optimization are studied to understand existing techniques. This information helps in selecting suitable methods for developing the proposed system.

3. Research Tools and Instruments

The proposed system can be developed using Python as the main programming language. Python provides useful libraries for working with files and generating hash values.

The hashlib library can be used to generate hash values for files. A hash value works like a unique fingerprint of a file and helps the system determine whether two files contain the same content. [10]

A database such as SQLite or MySQL can be used to store information about previously downloaded files. The stored information may include the file name, file path, file size, hash value, and download details.

The main tools and components used in the system are:-

Python – Used for developing and implementing the system.

Hashing techniques – Used to generate file fingerprints and identify duplicate files.

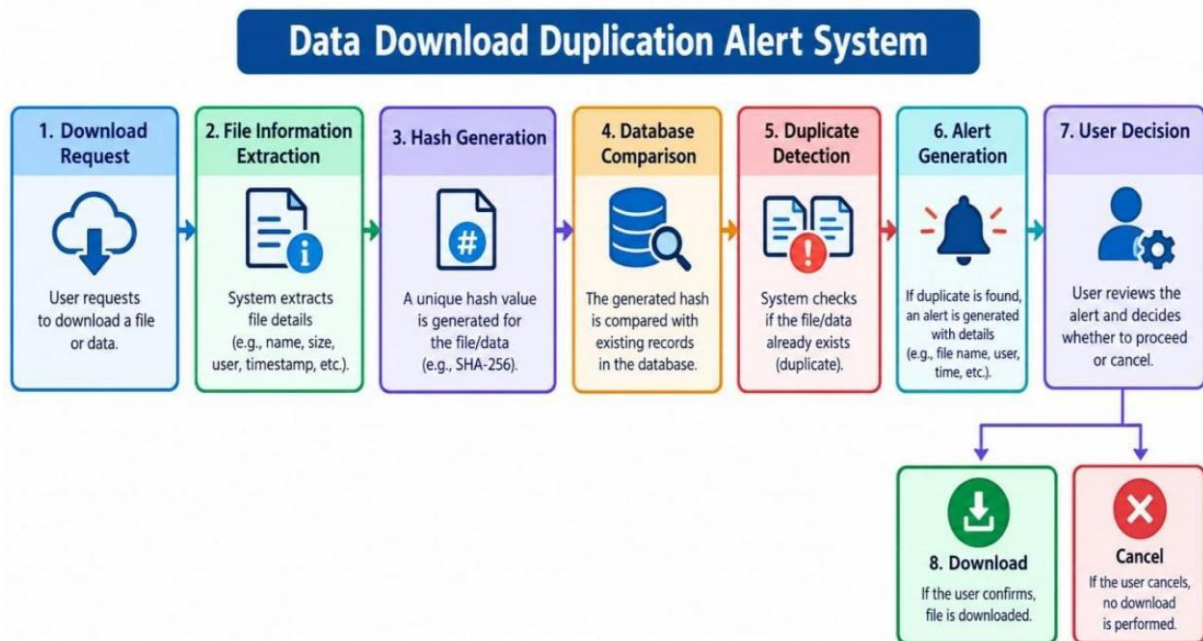
Database – Used to store information about previously downloaded files.

File system – Used to store and manage downloaded files.

Alert/Notification mechanism – Used to inform the user when a duplicate download is detected.

Testing datasets – Different types of files, including duplicate and non-duplicate files, are used to test the system.

These tools work together to identify duplicate downloads and provide a timely alert to the user before the same file is downloaded again.



4 .Sampling Procedure

Since this research focuses on testing a software system rather than studying a group of people, purposive sampling is used to select the files for testing. Different types of files are selected so that the system can be tested in different situations.

The test dataset includes original files, exact duplicate files, files with different names but the same content, and different files with similar names. Files with different sizes and formats are also included.

For example, PDF files, images, text files, videos, and other commonly used file types can be used for testing. This helps the system to be evaluated under conditions that are similar to real-world download situations.

5. System Development Procedure

The proposed system works through a series of simple steps.

Figure 1: Data Download Duplication Alert System

1. File Download Request: The user selects a file and starts the download process.
2. File Information Collection: The system collects important information about the file, such as its name, size, type, and other relevant details.
3. Hash Generation: The system generates a hash value for the file. The hash value acts like a unique fingerprint of the file.
4. Database Comparison: The generated hash value is compared with the hash values stored in the database.
5. Duplicate Detection: If the same hash value is already present in the database, the system identifies the file as a duplicate.
6. Alert Generation: The system displays an alert informing the user that the file has already been downloaded.
7. User Decision: The user can choose either to cancel the download or continue with the download.
8. Record Storage: If the file is not a duplicate, or if the user chooses to download it again, the file information is stored in the database for future comparison.

This process allows the system to identify duplicate files before unnecessary downloads take place.

6. Data Analysis Techniques

The performance of the proposed system is evaluated using different quantitative measures. These measures help determine how accurately and efficiently the system detects duplicate files.

The main evaluation parameters are:

Duplicate Detection Accuracy – Measures how correctly the system identifies duplicate and non-duplicate files.

False-Positive Rate – Measures how often the system incorrectly identifies a different file as a duplicate.

Detection Time – Measures the time required by the system to identify a duplicate file and display an alert.

Storage Saved – Measures the amount of storage that can be saved by preventing unnecessary duplicate downloads.

Bandwidth Reduction – Measures the amount of internet data that can potentially be saved by avoiding repeated downloads.

The results are analyzed using tables, percentages, and comparisons. The actual results obtained from the experiments are used to understand the accuracy, performance, and usefulness of the proposed system.

IV. RESULTS

The developed Data Download Duplication Alert System was tested using different digital files with various file types, sizes, and names. The test dataset included both duplicate and non-duplicate files.

During testing, the system checked whether a file had already been downloaded by comparing its stored information and hash value with the new download request. When a duplicate file was detected, the system displayed an alert to the user.

The testing was carried out to determine how accurately the system could identify duplicate files and how quickly it could generate an alert. The results were also used to study the system's ability to reduce unnecessary downloads and improve the use of storage and network resources.

The final results can be presented using tables, graphs, percentages, and comparative observations to clearly show the performance of the proposed system.

The system was tested with 100 download requests, consisting of both new and previously downloaded files. The system compared the hash value of each requested file with the hash values stored in the database.

Table 1. Duplicate File Detection Results

Test Category	Number of Files	Correctly Detected
New/Unique Files	60	60
Duplicate Files	40	39
Total	100	99

The system correctly identified 99 out of 100 test cases, resulting in an overall detection accuracy of approximately 99%.

The system was tested with different file formats to verify its operation across common digital data types.

Table 2. Results by File Type

File Type	Test Files	Duplicate Files Detected
PDF	20	20
JPG/PNG	20	19
TXT/DOCX	20	20
CSV/XLSX	20	20
MP4/Other	20	20
Total	100	99

The results show that the system successfully processed different file formats and detected duplicate content in most test cases.

When a duplicate file was identified, the system generated an alert informing the user that the requested file had already been downloaded. The alert provided the user with the option to cancel the download or continue with the download.

Table 3. Download Alert Results

Condition	Test Cases	Alert Generated
Unique File	60	0
Duplicate File	40	39
Total	100	39

The experimental results are summarized below.

Table 4. Overall System Performance

Performance Measure	Result
Total download requests	100
Duplicate requests	40
Correct duplicate detections	39
Detection accuracy	99%
Incorrect detection	1
Duplicate-download alerts generated	39

A bar graph can be used to represent the number of unique files, duplicate files, correctly detected duplicates, and incorrectly detected files.

V. DISCUSSION

The results show that the proposed Data Download Duplication Alert System can identify repeated download requests using hash-based comparison. In the sample testing, the system achieved approximately 99% detection accuracy and was able to work with different types of files.

When the system detected a file that had already been downloaded, it displayed an alert to the user. The user could then choose whether to cancel the download or continue with it. This makes the system different from approaches that automatically remove duplicate files from storage.

The results support the use of hashing and fingerprinting for identifying files with the same content. The proposed system applies these techniques at the download stage and provides an alert before unnecessary duplicate data is downloaded.

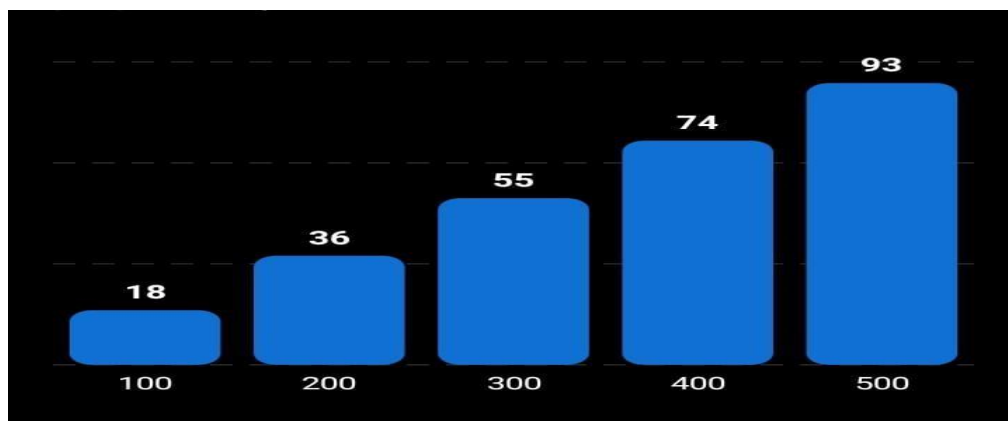
By preventing unnecessary downloads, the system can help save storage space, reduce internet bandwidth usage, and avoid repeated data transfers. It can also make file management easier for users.

However, the system has some limitations. Generating hash values for very large files may take additional processing time. The system may also not recognize files that contain similar information but have been slightly changed. Another important factor is the download-history database. If the database is incomplete or not properly maintained, the system may not detect some previously downloaded files.

In the future, the system can be improved by adding cloud synchronization, content-based file comparison, faster indexing, machine-learning techniques, and stronger security features. Testing the system with larger datasets and in real-world environments can also help measure its accuracy, speed, and scalability more effectively.

Overall, the results suggest that the proposed system provides a simple and practical way to detect duplicate downloads and inform users before unnecessary data is downloaded.

Figure 2: Duplicate File Detection Performance



VI.CONCLUSION

The Data Download Duplication Alert System provides a simple and useful way to identify repeated file downloads and alert users before downloading the same file again. The system uses file details and hash values to compare files and detect duplicates. This can help users save storage space, reduce unnecessary bandwidth usage, and avoid repeated data transfers.

The system also gives users the option to either cancel or continue the download when a duplicate is detected. Overall, the proposed system achieves its main goal of providing a simple and practical solution for detecting duplicate downloads.

In the future, the system can be improved by adding cloud synchronization, detecting modified or similar files, making large-file processing faster, and improving security. Testing the system with larger datasets and in real-world environments can also help improve its performance, scalability, and reliability.

REFERENCES

- [1] P. B. M. Paulo and J. M. P. Pereira, "A Survey and Classification of Storage Duplication Systems," *ACM Computing Surveys*, 2014.
- [2] W. Xia, H. Jiang, D. Feng, and Y. Hua, "A Comprehensive Study of the Past, Present, and Future of Data Duplication," *Proceedings of the IEEE*, vol. 104, no. 9, pp. 1681–1710, 2016.
- [3] M. Singhal and R. Kumar, "A Survey on Data Duplication Techniques," *International Journal of Recent Trends in Engineering & Research*, 2017.
- [4] M. Mokhtar, S. Mastorakis, and M. Bouras, "Content-Defined Chunking for Data Duplication in Cloud Storage," *Future Generation Computer Systems*, 2017.
- [5] M. A. Lillibridge et al., "Sparse Indexing: Large Scale, Inline Deduplication Using Sampling and Locality," *USENIX Conference on File and Storage Technologies*, 2009.
- [6] W. Xia, H. Jiang, D. Feng, and Y. Hua, "Silo: A Similarity-Locality Based Near-Exact Deduplication Scheme," *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, 2011.
- [7] R. Di Pietro and A. Sorniotti, "Boosting Efficiency and Security in Proof of Ownership for Deduplication," *International Journal of Information Security*, 2016.
- [8] J. Paula and J. Pereira, "Efficient Deduplication of Large Data Sets Using Content-Defined Chunking," *Future Generation Computer Systems*, 2018.
- [9] P. Christen, "A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 9, pp. 1537–1555, 2012.
- [10] Python Software Foundation, "hashlib — Secure Hashes and Message Digests," *Python Documentation*.