IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 6, May 2026
Impact Factor: 8.2

DDAS: Data Download Duplication Alert System

Kishor Laxman Thorat[1], Ankita Umesh Tiwari[2], Nishant Upadhyay[3], Nirmala Shinge[4]
Department of BCA, School of Computational Sciences
Faculty of Science and Technology JSPM University, Pune, Maharashtra, India
kishorlaxmanthorat@gmail.com[1] ankitatiwari2127@gmail.com[2]
nishantupadhyayofficial@gmail.com[3] nss1.scos@jspmuni.ac.in[4]

Abstract: *Today, organisations and educational institutions manage enormous volumes of shared digital files. When many users access the same repository without any coordination mech- anism in place, the same files get downloaded over and over— quietly draining storage, bandwidth, and system responsiveness. The Data Download Duplication Alert System (DDAS) is a web- based solution built specifically to tackle this problem.*

The system combines SHA-256 cryptographic hashing with Levenshtein distance and Jaccard similarity algorithms to detect both exact and near-duplicate downloads in real time. Every download event is logged with its associated metadata, and as soon as the system spots a repeated file it immediately raises an alert and notifies both the user and the administrator. DDAS runs on a three-tier architecture: a responsive single-page frontend, a Python Flask backend, and an SQLite database operating in Write-Ahead Logging (WAL) mode for high-concurrency perfor- mance. Security is handled through JWT-based authentication and role-based access control, while Google Generative AI adds intelligent analytical insights on top of the core system.

Development followed the Software Development Life Cycle, and the finished system passed all stages of testing—unit, inte- gration, performance, security, and user acceptance. The results confirm a meaningful reduction in redundant downloads and noticeably better visibility into how organisational data resources are actually being used.

Keywords: Data Duplication, Download Monitoring, SHA-256 Hashing, Alert System, Flask, JWT Authentication, Web Application, Data Management

I. INTRODUCTION

File duplication is one of those problems that organisations rarely notice until its cost has already quietly accumulated. In shared digital environments—whether a university server, a corporate repository, or a government dataset portal—users routinely access the same files independently of one another. Without any coordination mechanism, the same dataset can be downloaded dozens of times by different people, or even by the same person across multiple sessions. Each individual download seems trivial, but together they represent a real waste of bandwidth, storage capacity, and system processing time.

The Data Download Duplication Alert System (DDAS) was built to close this gap. Rather than relying on policy commitments or user discipline, it introduces an automated technical layer that monitors every download, fingerprints each file using SHA-256 cryptographic hashing, and cross-checks it against a centralised log of prior downloads. If a match is found, the system raises an alert and, depending on the configured rules, may block the redundant transfer entirely.

Beyond simple duplicate detection, DDAS integrates fuzzy matching algorithms to catch near-duplicates—files that are functionally the same but differ slightly in name, format, or metadata. The system also includes an AI-powered chat interface built on Google Generative AI, allowing users to ask plain-language questions about download history and repository contents. Real-time filesystem monitoring via the Watchdog library ensures the system reacts the moment activ- ity occurs, rather than waiting for periodic batch jobs.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 6, May 2026 Impact Factor: 8.2

This paper describes the complete design, implementation, and testing of DDAS. The system was developed in response to Problem Statement ID 1659 issued by the Indian National Centre for Ocean Information Services (INCOIS) under the Ministry of Earth Sciences, which identified redundant dataset downloads as a genuine operational problem in large institu- tional environments.

II. MOTIVATION AND PROBLEM DEFINITION

The impetus behind DDAS comes directly from a concrete institutional challenge. Research organisations and government repositories often maintain large shared datasets that many teams need to access. Because there is typically no mechanism to tell a user that someone else has already retrieved a given file, teams end up downloading independent copies of identical data. Over time this leads to fragmented storage, inconsistent file versions, and inflated network usage—all of which carry real operational costs.

The problem is not confined to research institutions. In uni- versity settings, hundreds of students may download the same lecture notes or assignment files from a shared portal over the course of a semester. In corporate environments, project teams working across departments unknowingly duplicate files across shared drives and cloud storage. In each case, the absence of a monitoring layer allows the duplication to continue unchecked. There is also a security dimension to uncontrolled down- loads. Repeated access to the same sensitive file—especially without logging—creates audit gaps. An organisation may have no reliable way to reconstruct who accessed a file, how many times, or whether that access was appropriate. DDAS addresses this directly by maintaining a complete, timestamped record of every download event.

The primary objectives the system was designed to meet are:

1) Monitor download activities in real time and maintain a complete log.
2) Detect duplicates—both exact and near-duplicate— using hashing and similarity algorithms.
3) Generate instant alerts whenever duplication is identi- fied.
4) Provide administrators with an interactive dashboard for monitoring and reporting.
5) Reduce redundant network and storage consumption in shared digital environments.

III. LITERATURE SURVEY

Research in data deduplication and download monitoring spans several disciplines. The following works directly shaped the design choices made in DDAS.

Work published in IEEE Transactions on Cloud Comput- ing (2022) on data deduplication in cloud storage systems established that SHA-256 hashing offers a reliable, collision- resistant mechanism for identifying duplicate files at scale [1]. Its treatment of hash functions and storage optimisation mod- els provided the theoretical basis for DDAS's core detection logic.

A study from the ACM Digital Library (2021) on real-time download monitoring systems highlighted the importance of time-series logging, database indexing, and anomaly detection [2]. Its framework for notifying administrators immediately upon detection of irregular download patterns directly in- formed the structure of DDAS's alert pipeline.

Research from Springer's Journal of Information Security (2020) on authentication and access control in web appli- cations contributed to the security architecture of DDAS, particularly its analysis of JWT-based token validation and role-based access control models [3].

A paper from the International Journal of Computer Net- works (2023) on bandwidth optimisation through duplication detection provided the theoretical underpinning for DDAS's network efficiency design, with its cache replacement al- gorithms and duplication probability functions guiding the detection engine [4].

Work from the Elsevier Journal of Software Engineering (2022) on alert systems in data management applications shaped DDAS's notification architecture, particularly its use of threshold functions and decision trees for rule-based noti- fications [5].

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 6, May 2026 Impact Factor: 8.2

More recent works—including event-driven duplicate de- tection papers from the Journal of Network and Computer Applications (Elsevier, 2025) [8] and lightweight monitoring research from IEEE Software (2024) [9]—contributed to the real-time processing architecture and informed the adoption of the Watchdog library for filesystem event handling.

IV. SYSTEM DESIGN AND ARCHITECTURE

A. Three-Tier Architecture

DDAS is organised as a three-tier application. The pre- sentation tier is a single-page application (SPA) built with HTML5, CSS3, and vanilla JavaScript, supplemented by Chart.js for data visualisation. It provides the interface for file uploads, alert viewing, history browsing, and the AI chat module. The logic tier runs a Python Flask backend that handles all application processing: authentication, duplication detection, alert generation, and API responses. The data tier uses SQLite operating in Write-Ahead Logging (WAL) mode, which allows concurrent reads and writes without locking— critical for a system that simultaneously logs events and responds to user requests.

B. Duplication Detection Engine

The detection engine works in two stages. In the first stage, the system computes the SHA-256 hash of the uploaded or monitored file and queries the Files table for a record with the same hash value. If one is found, the download is flagged as an exact duplicate. In the second stage, for cases where no exact match exists, the system applies Levenshtein distance to com- pare filenames and Jaccard similarity to compare tokenised content descriptors. This two-stage approach ensures that files which have been renamed or slightly modified are still caught as near-duplicates—a scenario that purely hash-based systems miss entirely.

C. Real-Time Filesystem Monitoring

The Watchdog library runs as a background observer that continuously monitors designated directories. The moment a file creation, modification, or move event occurs, Watchdog captures it and passes it to the detection engine. This gives DDAS a genuinely live quality—it does not wait for a user to upload something through the web interface; it responds to filesystem activity directly, making it suitable for environments where files arrive through multiple pathways simultaneously.

D. Security Architecture

Authentication is implemented using PyJWT, which issues signed bearer tokens upon successful login. Every API call must present a valid token; requests without one are rejected at the middleware level. Passwords are stored using bcrypt with PBKDF2 and 600,000 iterations, making brute-force attacks computationally infeasible. Role-based access control distin- guishes three user classes: administrators with full system access, registered users subject to duplication checks, and guest users with limited read-only access. The system also includes XSS protection, SQL injection prevention through parameterised queries, path traversal protection, and SSRF validation on URL-based imports.

E. Database Schema

Five tables support the system. The Users table holds cre- dentials and role assignments. The Files table stores metadata and hash values for every file the system has seen. The Downloads table records every download event, linking users to files with timestamps and status fields. The Alerts table captures each detected duplication event with severity and resolution status. The Logs table maintains a full system event journal for audit purposes. All tables are indexed on their most frequently queried fields to ensure fast retrieval under concurrent load.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 6, May 2026
Impact Factor: 8.2

V. SYSTEM REQUIREMENTS

Table I — Hardware Requirements

Component	Specification	Purpose in DDAS
Processor	Intel Core i5 or equiva- lent	Hashing,detection, backend processing
RAM	8 GB (expandable to 16 GB)	Handles concurrent re- quests smoothly
Storage	256 GB SSD	Stores application, database, and log files
Network	Wi-Fi / LAN	Data transfer and real- time monitoring
System Type	Laptop or Desktop (Windows/Linux)	Runs frontend, backend, and database locally
Backup Device	External Drive / Local Storage	Data safety and recovery support

TABLE I: HARDWARE REQUIREMENTS

Table II — Software Requirements

Component	Technology	Purpose / Function
Frontend	HTML5,CSS3, JavaScript, Chart.js	Responsive UI with real- time analytics and alert vi- sualisation
Backend	Python 3.11+, Flask, Werkzeug	Duplication detection logic, APIs, and business rules
Database	SQLite (WAL mode)	Stores metadata, logs, and user activity with opti- mised performance
Hashing & Algo- rithms	hashlib SHA-256, Levenshtein, Jaccard	Exact and near-duplicate detection
Authentication	PyJWT, Bcrypt (pbkdf2:sha256)	Secure login, token-based auth, password protection
File Monitoring	Watchdog	Detects real-time filesys- tem changes
AI Integration	Google Generative AI SDK	Intelligent analysis and decision support
Version Control	Git / GitHub	Source code management and collaboration

TABLE II: SOFTWARE REQUIREMENTS

VI. IMPLEMENTATION

A. Development Methodology and Timeline

Development followed the Software Development Life Cy- cle over a nine-week schedule. Weeks one and two covered requirement gathering and system design—architecture plan- ning, database schema design, and flowchart creation. Weeks three and four focused on backend development: Flask API setup, SQLite integration, SHA-256 hashing, and JWT authen- tication. Week five was dedicated to building the frontend SPA, Chart.js dashboards, and the upload and alert visualisation modules. Week six handled integration of all layers. Weeks seven and eight ran the full testing suite—manual testing, inte- gration verification, performance benchmarking, and security validation. Week nine finalised documentation and prepared the submission package.

B. Duplication Detection Algorithm

Algorithm 1 — Duplication Detection. The system re- ceives a file, computes its SHA-256 hash using Python's hashlib module, and queries the Files table for a matching hash. If a match exists, it triggers Algorithm 2 (alert dispatch) and logs the duplication event. If no exact match is found, the filename and tokenised metadata are compared against recent entries using Levenshtein distance and Jaccard similarity. If either similarity score exceeds the configured threshold, the file is flagged as a near-duplicate and alert dispatch begins. Otherwise, the file is treated as new: its metadata and hash are written to the database and the download is permitted.

Algorithm 2 — Alert Notification. Upon confirmation of duplication, the system creates an entry in the Alerts table with the file ID, user ID, timestamp, and severity level. It then dispatches a notification to the user via the in-app

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 6, May 2026 Impact Factor: 8.2

notification system and updates the administrator dashboard in real time. If email or WhatsApp notification is configured, those are sent as well. If the rule for the detected severity level calls for restriction, the download is blocked and the user receives an explanatory message.

C. Team Structure and Responsibilities

Kishor Thorat served as backend developer, implementing the core detection logic, Flask APIs, database integration, and authentication modules. Ankita Tiwari acted as project manager and tester, coordinating milestones, conducting man- ual testing, and preparing documentation. Nishant Upadhyay handled frontend development, building the SPA interface, Chart.js visualisations, dashboard components, and the alert display system.

VII. RISK MANAGEMENT

TABLE III RISK ANALYSIS

Risk Type	Likelihood	Impact	Mitigation Strategy
Hash collision	Low	Medium	SHA-256 makes acciden- tal collisions computation- ally negligible
Database over- load	Medium	High	Indexing, caching, and regular backup routines
Network latency	High	High	Load balancing and asyn- chronous alert dispatch
Unauthorised access	Low	High	JWT authentication with role-based access control
User resistance	Medium	Medium	Intuitive UI design and user awareness training

Hash collisions pose a theoretical but practically negligible risk with SHA-256: its 256-bit output space makes accidental collisions computationally impossible under current hardware. Database overload is mitigated by indexing all frequently queried columns and running SQLite in WAL mode, which separates read and write operations and prevents table-level locking. Network latency is addressed by making alert dispatch asynchronous—the detection result reaches the user immedi- ately while the notification pipeline runs in a separate thread. Unauthorised access is prevented by validating JWT tokens at every API endpoint before any data is returned or written.

VIII. SOFTWARE TESTING

A. Testing Approach

Six categories of testing were applied to DDAS. Unit testing verified each module independently—hashing correct- ness, metadata comparison logic, and alert triggering were each tested in isolation before integration. Integration testing confirmed that data flows correctly through the full pipeline: from the frontend upload interface through the Flask backend to the SQLite database and back to the alert system. System testing validated the complete end-to-end workflow under realistic operating conditions. Performance testing measured response times under simulated loads of 500, 1,000, and 2,000 concurrent requests. Security testing verified authentication enforcement, role-based access restrictions, and protection against common web vulnerabilities. Finally, user acceptance testing was conducted with sample users to evaluate dashboard usability and alert clarity.

Table IV — Test Cases and Results

ID	Description	Expected Output	Actual Result	Status
UT01	SHA-256 hashing logic	Correct hash generated	Hash correct	Pass
UT02	Metadata comparison	Duplicate flagged	Duplicate flagged	Pass
UT03	Alert generation	Alert trig- gered	Alert trigered	Pass
IT01	Frontend to DB flow	Metadata stored, check done	Flowcorrect	Pass
IT02	Alert notification	Alert sent to user	Alert sent	Pass
ST01	End-to-end work- flow	Detect + alert + log	Workflow executed	Pass
PT01	Load (100 re- quests) 0	Detection < 2 sec	1.8 sec	Pass

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 6, May 2026
Impact Factor: 8.2

PT03	Stress (2000 re- quests)	System sta- ble	Stable	Pass
SEC01	Authentication check	Access de- nied	Denied	Pass
SEC02	Role-based access	Access re- stricted	Restricted	Pass
UAT01	Dashboard usability	Easy navi- gation	Intuitive	Pass
UAT02	Alert accuracy	Alert cor- rect	Alert cor- rect	Pass

TABLE IV: TEST CASES AND RESULTS

B. Performance Summary

Under a simulated load of 1,000 concurrent download requests, the system completed duplication detection and alert generation in an average of 1.8 seconds—well within the 2- second target. Stress testing at 2,000 simultaneous requests confirmed system stability with no unhandled exceptions or database lock failures. Security testing confirmed that invalid credentials, unauthorised role escalation attempts, and direct log-access attempts were each blocked correctly at the mid- dleware level.

IX. RESULTS AND DISCUSSION

The deployed DDAS prototype was evaluated against a controlled dataset simulating realistic institutional download activity. Over the evaluation period, the system tracked 32 registered datasets and successfully blocked 78 duplicate download attempts—a detection rate consistent with the levels of redundancy typically observed in shared academic and organisational repositories. Every blocked attempt generated a correctly formatted alert containing complete metadata: the original downloader's identity, the timestamp of the first download, the file hash, and the current user's identity.

The administrator dashboard rendered all data in real time. Chart.js visualisations made it straightforward for non-technical administrators to interpret download trends and du- plication frequency without needing any database knowledge. The AI chatbot module, powered by Google Gemini, allowed users to ask natural-language questions—such as "which files were downloaded most frequently this week?"—and receive coherent summaries drawn directly from the live database.

The Duplicate Detector module, which scans local direc- tories rather than waiting for upload events, demonstrated the system's ability to work entirely outside the browser interface. During testing, it correctly identified multiple copies of the same JavaScript file spread across different subdirectories, reporting the hash, location, and storage wasted by each copy. One limitation observed during testing was increased de- tection latency for very large files—above 500 MB—because SHA-256 hashing of the full file occurs server-side before the download decision is made. For typical institutional files (documents and datasets under 100 MB), this latency was im- perceptible. For larger files, client-side pre-hashing or progres- sive hash computation during transfer would be a meaningful optimisation in future versions.

X. CONCLUSION

DDAS demonstrates that the problem of redundant file downloads—long treated as an unavoidable side effect of shared digital infrastructure—can be addressed effectively through a well-designed automated monitoring system. By combining SHA-256 hashing for exact detection, Levenshtein distance and Jaccard similarity for near-duplicate matching, real-time filesystem monitoring via Watchdog, and a secure web application layer, the system delivers a comprehensive solution that works both proactively (blocking duplicates as they occur) and retrospectively (surfacing historical duplica- tion patterns through analytics).

The architecture is deliberately modular. Each component— the detection engine, the alert pipeline, the authentication layer, the analytics dashboard, and the AI interface—can be developed, tested, and extended independently. This makes

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 6, May 2026 Impact Factor: 8.2

DDAS straightforward to adapt for different organisational contexts, whether a university portal, a corporate file server, or a government dataset repository.

All six testing categories returned passing results. The system performed reliably under stress conditions and was rated positively by users during acceptance testing. DDAS v4.0 establishes a solid foundation for the more advanced capabilities described in the future scope, including full cloud- native deployment, mobile companion applications, and se- mantic duplicate detection using embedding-based AI models.

XI. FUTURE SCOPE

• Semantic duplicate detection using embedding-based AI models, enabling identification of files that are con- ceptually similar even when hash and string comparisons return no match.

• Cloud-native deployment on AWS, Azure, or Google Cloud with auto-scaling and high-availability configura- tions for enterprise-scale environments.

• Mobile companion apps for Android and iOS, delivering real-time alerts and dashboard access to administrators and users on the go.

• Predictive analytics and anomaly detection dashboards that forecast duplication trends and flag unusual down- load patterns before they escalate.

• REST and GraphQL API ecosystem to enable DDAS to integrate as a plug-and-play module within existing enterprise data management infrastructure.

• Dataset versioning and collaborative access tracking to support team-based environments and maintain complete audit trails for compliance.

• Enhanced authentication, including two-factor authen- tication, biometric login, and alignment with GDPR and ISO 27001 standards for regulated sectors.

• Multi-language support and region-specific customisa- tion to broaden accessibility across different countries and industries.

ACKNOWLEDGMENT

The authors express sincere gratitude to Mrs. Nirmala Shinge for her expert guidance, consistent encouragement, and technical direction throughout this work. Heartfelt thanks are extended to Dr. Anita Pisote (Project Coordinator), Dr. Santosh Gaikwad (Program Coordinator, BCA Department), Prof. G. A. Patil (Director), and Prof. R. S. Deshpande (Dean, School of Computational Sciences), Faculty of Sci- ence and Technology, JSPM University Pune, for providing the resources and support necessary to carry this project to completion.

REFERENCES

[1] L. Zhao et al., “Efficient file system monitoring and duplicate detection using real-time hashing strategies,” IEEE Access, vol. 10, pp. 112345– 112360, 2022.

[2] R. Singh and S. Kumar, “Optimized duplicate file detection in event-driven environments,” Journal of Systems Architecture, Elsevier, vol. 129, p. 101986, 2023.

[3] W. Chen and A. Gupta, “Event-driven filesystem monitoring frameworks for proactive alerting,” ACM Transactions on Storage, vol. 19, no. 3, Article 27, 2023.

[4] D. Roy, P. Sengupta, and T. Zhang, “Leveraging event-driven monitoring for security and intrusion detection,” International Journal of Cyber- Security and Digital Forensics, vol. 13, no. 2, pp. 45–58, 2024.

[5] S. Patel and A. Mehta, “Visual analytics dashboards for system health and storage monitoring,” Frontiers in Big Data, Springer, 2025.

[6] Ministry of Education, Government of India, “Smart India Hackathon 2023: Problem Statements and Evaluation Guidelines,” New Delhi, India, 2023.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 6, May 2026
Impact Factor: 8.2

[7] Ministry of Education, Government of India, "Smart India Hackathon 2024: Innovation Themes and Judging Criteria," New Delhi, India, 2024.
[8] R. Sharma, H. Verma, and P. Singh, "Event-driven duplicate detection and alerting systems for smart infrastructure," Journal of Network and Computer Applications, Elsevier, vol. 215, Article 103791, 2025.
[9] A. Malhotra and N. Joshi, "Lightweight monitoring and alert systems for cybersecurity applications," IEEE Software, vol. 41, no. 1, pp. 66–74, 2024.
[10] National Innovation Foundation, "Design principles for hackathon-ready intelligent monitoring systems," Government of India, 2025.