Designing a Web-Based Cloud Storage Management System to Optimize Data Retrieval, Storage Efficiency, Security, and Accessibility

Research Project
MSc Cloud Computing

RAJU GUMPULA
Student ID: x23220708

School of Computing
National College of Ireland

Supervisor: Abubakr Siddig

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	RAJU GUMPULA		
Student ID:	X23220708		
Programme:	MSc in Cloud Computing	**Year:**	2024
Module:	Research Project		
Supervisor:	Abubakr Siddig		
Submission Due Date:	29/12/2024		
Project Title:	Designing a Web-Based Cloud Storage Management System to Optimize Data Retrieval, Storage Efficiency, Security, and Accessibility		
Word Count:	8087	**Page Count**	22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Raju Gumpula

Date: 29/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Designing a Web-Based Cloud Storage Management System to Optimize Data Retrieval, Storage Efficiency, Security, and Accessibility

Raju Gumpula
X23220708

Abstract

This project involves the Designing of the cloud based File sharing and storing Application utilizing AWS services to increase effectiveness and dependability in file transmitting and user messaging. Due to this, the system incorporates AWS Amplify, AppSync as well as DynamoDB to support data storage, access and synchronizing. AWS SNS and Lambda handle notifications, making the system capable of providing real time notification on the events such as, when a file has been uploaded or updated. In order to give robust security, Amazon Cognito is used for user sign on and Amazon GuardDuty for malware scan of files to be uploaded. It utilizes CloudFront for distributing content and CloudWatch for tracking system and logs performances. Largely, this application focuses on scalability, security and usability issue that one can associate with most cloud application solutions. The project offers a sound framework for safe and efficient notification solutions and offers space for optimisation in the form of broadening and diversifying notifications and adding bespoke analysis features.

1 Introduction

1.1 Research Background

The raw quantity and speed of data generation that today's digitized world presents have contributed to the need for efficient file storage and management systems. Modern complex and dynamically changing environments have presented many challenges to traditional storage solutions and their key parameters such as availability, security and, especially, the ability to scale up when required by the nature of the business process, have failed to meet modern expectations especially where it comes to storing massive amounts of data that need to be accessed and shared by users located in different geographical locations (Singh et al., 2021). Web-based options have risen as more appropriate options that provide flexibility, affordability, and compatibility with other online products. Nevertheless, the difficulties, which differ from those typical for a developed web-application, remain: data protection, stable work of system under high loads, and convenience of interface for users.

Studies show that popular cloud applications such as Google Drive and Dropbox are currently up to industrial standards, but no one has effectively incorporated better security, data analysis, and performance into such systems (Kadaskar and Kamthe, 2024). Currently, most platforms have little or no protocols for managing threats like unauthorized access and data breaches. Moreover, it lacks scalability of the increases of user load and vast data in some cases.

That is why this project tries to fill in these gaps by implementing and deploying the cloud-based file sharing and storage application using the Amazon Web Services (AWS). With characteristics such as feature services S3 for secure storage, CloudFront for faster content delivery and IAM for accurate access control, this solution will offer a modern storage solution needed by users (Kewate et al., 2022).

1.2 Aim

The main aim of this study is to improve web-based cloud management system functionality by integrating it with AWS services, attaining security and scalability to a higher level, and also improving performance.

1.3 Objectives

- To design an intuitive user interface for easy access to data.
- To optimize cloud infrastructure for advanced storage management and scalable solutions.
- To safeguard data through secure protocols with potential confidentiality and privacy.

1.4 Research Question

How can a web-based cloud storage management system be designed to optimize data retrieval speeds and storage efficiency while ensuring robust security and user accessibility?

1.5 Problem Statement

However, the need for dependable and most of all elastic cloud-based file storage and sharing platforms and solutions has been realized in the contemporary world. However, many currently existing platforms are experiencing certain critical issues that act as barriers towards fulfilling users' needs. Problems including low data protection, poor site's extendibility at the usage rush, slow information sharing, and low user restriction. Identification, unauthorized access and overall poor systems' performance have raised major risks that are associated with the commonly implemented platforms, to both individual and organizational users in their online computing systems. Furthermore, with data storage and sharing demands continually on the rise around the world, traditional architectures fail to integrate at optimal measures, thereby encountering performance issues and degrading the user experience. Furthermore, no coherent systems for controlling possible security threats or risks in real-life are also added to these difficulties. Customers often experience challenges in trying to achieve both, keep their data secure and still have the data easily accessible.

These issues are worked out in this project by designing and implementing a cloud-based file sharing and storage application using AWS technology. With the help of such features as S3 for the safe storage of files, GuardDuty, for real-time threat detection, CloudFront, for content delivery at high speed, the solution plans to become safe and scalable, secure, and capable of meeting the dynamic demands of modern data storage and sharing environments due to user-friendly platform.

1.6 Research Significance

The importance of this study therefore resides in meeting major concerns relating to file storage and sharing in a cloud environment. Due to the tremendous increase in the flow of digital data, conventional file storage systems fail to deliver adequate protection, flexibility, and convenience. This present study provides a solution to this research problem by proposing a cloud-based application that utilizes AWS services to improve the availability, security, and efficiency of centralized file storage and sharing systems using S3, GuardDuty, and CloudFront services (Borra, 2024).

The features proposed in the application include threat identification in real time, fast file delivery, and secure access to servers which can prevent risks associated with information leak, unauthorized access, and slow performance. The AWS environment as a large highly scalable and secure ecosystem ensures that users can adapt to workloads variations safely without risks to performance (Rai, 2024).

Moreover, this work will help to expand the knowledge in the field of cloud computing, as well as the ability to show how the contemporary technologies can be effectively applied to overcome the existing deficits. It gives a clue for further developments of other secure applications based on cloud model, so that it will be instrumental for developers, researchers, and organizations (Gupta et al., 2021). In conclusion, this work contributes to better data protection, organization performance, and user satisfaction, which makes this work valuable and contributes to the Designing of the file sharing and storage system domain of cloud based.

1.7 Research Motivation

The purpose for this work is the trend towards using digital data and problems users meet when it comes to secure data storage and sharing in large scale. Most conventional file storage systems do not have the capacity needed to serve today's needs, including data storage and protection, and durability when accessed frequently (Park et al., 2022)**.** Further, the rise in data violation and unauthorized access has put emphasis on the security aspect of the cloud storage solutions.

The opportunities provided by cloud computing, especially based on AWS look like an opportunity to solve these problems. S3, IAM, and CloudFront are the services that provide scalability, secure data access and optimized performance thus AWS is perfect solution for designing next generation file sharing/storage application (Mansouri and Buyya, 2020). The drive to use AWS for this project is therefore informed by the common occurrence of scalable, economical, and easy to use solutions on AWS.

This research is also motivated by the possibility of contribute with solutions to existing challenges in cloud computing such as scalability, security, and usability. Therefore, as it is presenting a revolutionary solution, it is its goal to equip the users with a safe and optimized environment that corresponds to the requirements of the twenty-first century.

2 Related Work

2.1 Cloud Computing in File Storage and Sharing

Cloud computing has viewed itself as being as a platform that has changed the storage, processing and sharing of data in a way that is very unique in terms of its ability to scale, reliability, and accessibility. The use of the solutions based on files hosting and sharing in the cloud is motivated by the challenge of the growing volume of data that has to be processed and store and have to be made easily accessible and secure (Kotha et al., 2022). Traditional Storage systems are mostly on-premise while cloud platform offered users a way of storing humongous amount of data without the need to invest in the infrastructure. This is especially the case at a time when data creation is rapidly growing because of technology and the Internet of Things.

In file storage and sharing the cloud platform features are like, centralization of storage and sharing, real time collaboration, remotely accessible. These capabilities are very important for organizations and individuals who would like to store and share files over multiple groups, offices or areas. Google drive, Dropbox and One Drive are some of the examples that has followed good user interface with good back-end system. Although, these platforms have some peculiarities like limited opportunities for personalization, a critical dependence on the vendor's technologies, and risky for leakage of sensitive information (Prajapati and Shah, 2022).

Later, cloud service providers such as AWS have brought new enhancements to this discipline as developers can create custom solutions. AWS such as S3 comes with almost inexhaustible storage ability thus making user data highly available and durable. It should also be noted that cloud-based file sharing systems can incorporate use CDN, to facilitate faster file access across geographical regions (Yang et al., 2023). Such improvements prove the viability of cloud computing in meeting the increasing needs for file storage and sharing products.

2.2 Security Challenges and Solutions in Cloud-Based Applications

Security is a fundamental factor in all cloud-based file storage and sharing application solutions since users upload important and sometimes, sensitive information to these applications. Cloud systems are by nature vulnerable to security risks including invasions by unauthorized individuals, theft, and even denial of service attacks (Basu et al., 2018). The distributed nature of cloud environments aggravates these risks as follows, where data may traverse multiple locations and networks.

A major security concern is that sensitive information can only be available to those who ought to view it: An example of this security concern is where data is to be made available to a specific group of persons only. It is solved by means of access control and identity management as this challenge. Leverages such as AWS Cognito make it easier for developers to provide user authentication and control of user access to materials by implementing role-based security. In addition, there is equally the need to augment the encryption technologies that act as the shield to data both stored and moving from one system to another. Cloud service providers also come equipped with rich set of services that deal with encryption, including the server-side encryption provided by AWS S3 to guard stored data against unauthorized access and others (Sanne, 2022).

Another major problem relates to identification and handling of security threats in realtime. Logs and flow traffic analysis by using AWS GuardDuty in order to detect new threats. The mentioned tools ensure that system risks are prevented even before showing their negative consequences on system integrity. Also, activities such as auditing, logging, or compliance monitoring are critical in contributing to transparency and enhancing the security standards (Mykhaylova et al., 2024). However, it is still possible to sustain good security reinforcement level, one needs to be proactive and constantly update security. With the revolutionary threats, the cloud applications are expected to employ new security features like multipronged authentication and a zero-trust model.

2.3 Scalability and Performance Optimization in Cloud Infrastructure

That is why scalability is one of the essential characteristics of cloud structure, which guarantees the capacity to maximize systems' performance within the changeable workloads and the augmented volume of data. For file storage and sharing applications, availability of sufficient capacity is critical for file, users and concurrent access (Sharma and Chaturvedi, 2021). Cloud services, in general, and AWS in particular include solutions and architectural approaches that perfectly address these scalability needs.

A method that is widely recognized as efficient in reaching the goal of scalability is serverless computing. These self-service, pay-as-you-go services provide the capability to run code without the need to set up or coordinate servers, guaranteeing that all resources are in constant contention for usage. As such, this approach is highly suitable for operations such as file uploads where we work with files or create file previews in real time. Also, in delicate services such as AWS EC2, auto-scaling benefits guarantee that virtual machines can expand the level of horizontal scaling by adding or removing instances depending on workloads containing metrics (Saini and Behl, 2020).

Scalability is an obvious companion of performance optimization because user satisfaction is a direct function of the ability to access files quickly and efficiently. AWS CloudFront Content Delivery Networks to be more precise have assets that need to be cached at the edge so as to improve user performance. This minimizes the quest time to the ultimate desirer and allows easy retrieval of files irrespective of traffic congestion. Database optimization is also a significant area because metadata about the files stored should be retrieved and changed as needed. Some of the Databases known as NoSQL such as Dynamo DB support low latency data access which comes in handy when using to store the files metadata as well as access control (Sajid et al., 2024).

In addition to scalability, cloud applications' performance is boosted by monitoring and optimization tools. For example, AWS CloudWatch, gives the developers information about the system usage and system metrics such as CPU, latency, and throughput which help them to determine where the problem could be in system and how resources can be properly assigned (Singh and Aggarwal, 2024). When architectural principles are complemented with best practices for performance optimization, cloud hosted file sharing and storage applications can meet and sustain consumer expectations.

2.4 Integration of AWS Services for Cloud Application Development

AWS services are a gift for every application developer who wants to create secure, easily scalable, and efficient cloud solutions. AWS provides a broad spectrum of Services that can altogether be integrated to meet a number of needs of file sharing systems. Everything, starting with infrastructure storage and network security and ending with content delivery and analytics, are solutions made by AWS that can help to simplify the overall application building process and operation (Poornalinga and Rajkumar, 2016).

Revolved around file storage applications, there is Amazon S3, an object storage with almost unlimited scalability and designed for maximum durability and availability. S3 makes it easy to store objects in a bucket and access them when needed; this cloud storage solution provides versioning, lifecycle and server-side encryption. When integrated with DynamoDB, S3 helps to store file metadata like file name, size and access permission.

AWS Cognito is the best method on ensuring secure and scalable user authentication and authorization. Cognito user pools helps in handling users' registration and login while Cognito identity pools provide secure access to other AWS services. This make sure that use information and application services are guarded so that they are not accessed by wrong key holders.

Real-time processing is achieved through AWS Lambda and DynamoDB streams, event driven architecture also improves the system response. For instance, then an object is uploaded to S3, a DynamoDB attribute can invoke a Lambda function to inform the users or update the properties. It also minimizes interaction and guarantees that the system performs well without delays.

So as to provide content quickly for customers around the world, AWS CloudFront serves as a global CDN which caches files at the edge locations. It also means that the users can be confident in accessing the files closely without necessarily having to be sited in a specific area. Also, AWS GuardDuty and AWS CloudWatch offer perfect means for monitoring and security that allow developers to identify threats, improve performance and ensure security compliance.

AWS CloudFront, a CDN service of AWS, stores objects at edge locations to lower the time users have to wait when they request an object. Along with AWS GuardDuty and CloudWatch it provides a comprehensive reporting and security solution which helps in early detection of a threat, achievement of optimum utilisation of the system and be compliant with industry benchmarks.

2.5 Literature Gap

However, much research is still left to be conducted, as well as many shortcomings missed in present experimental counterparts of file sharing and storage solutions in cloud computing environments. They are the following: The problem of combining high-frequency data transfer in a real-time environment with a highly secure system has not yet been studied thoroughly. Services like AWS provide numerous features that allow for easy storage of files; however, many existing platforms fail to efficiently combine them to allow for transparent, secure, and real-time data access especially under high load.

Secondly there is absence of understanding the situatedness of the work, as well as the lack of focus on scalability and cost-effective solutions. Although the literature discusses the advantages of the serverless computing and auto-scaling models, there appears to be a lack of research regarding how such technologies may be implemented to support large scale file sharing applications without stratospheric costs. One of the less researched fields is how to achieve the best performance-to-cost ratio, especially in terms of repeatedly invoked operations such as file upload if the files are very large, or distributed access in a multiuser environment.

Also, there is an application like google drive and drop box that focus on usability factors but rarely there is a system that provides usability that is unique to the organizational goals or to the regulation. The fact that there is limited prior work in utilizing most AWS services in unison like AWS GuardDuty for threat detection, Amazon CloudFront for content delivery, amplifies the necessity for research in devising integrated solutions that address security, scalability challenges and use efficiencies. It is hoped that this project will also fill these gaps by developing a secure, scalable and cost-efficient cloud file sharing application using AWS services.

Author & Year	**Framework**	**Approach**	**Advantage**	**Limitation**
Viikari, V. (2022)	Monolithic to Microservice Migration	Integration of monolithic web services into a microservice-based cloud platform.	Enables scalability and agility for legacy systems.	Complexity and potential downtime during migration.
Zhang, J. Y., and Zhang, Y. (2024)	DevSecOps Metrics Framework	Quantitative analysis of DevSecOps metrics in cloud-based microservices.	Provides measurable insights for enhancing security and efficiency.	Requires alignment across diverse teams and tools.
Vadlamani, S., et al. (2024)	Cloud-Based Data Architectures	Integrating scalable data architecture solutions for enterprises.	Ensures scalability and optimized data management.	High initial setup cost and design complexity.
Yanamala, A.K.Y. (2024)	Data Storage Optimization in Cloud	Techniques and best practices for efficient cloud data storage.	Reduces storage costs and improves retrieval efficiency.	Limited applicability to non-standardized cloud systems.
Hashmi, S.A., et al. (2021)	IoT and Cloud-Based Energy Management	Energy management system leveraging IoT and cloud for smart grids.	Enhances energy efficiency and demand-side management.	Security and privacy concerns in IoT-cloud integration.
Alam, T. (2021)	Cloud-Based IoT for Smart Cities	Application of IoT in smart city development using cloud computing.	Facilitates real-time data processing and smart infrastructure.	Challenges in interoperability and data standardization.
Botonakis, S., et al. (2020)	iSWoT (Service-Oriented Architecture in the Cloud)	Semantic Web of Things using service-oriented architecture in cloud environments.	Enhances interconnectivity and data utilization in IoT.	High computational requirements and implementation complexity.
Abdullah, P.Y., et al.	Cloud-Based HRM for	Development of HR management systems	Affordable and scalable HR	Dependence on cloud providers and

(2020)	SMEs	for SMEs using cloud technology.	solutions for SMEs.	potential downtime issues.
Marinescu, D.C. (2022)	Cloud Computing Theory and Practice	Analysis of cloud computing concepts, theories, and practices.	Provides a holistic understanding of cloud systems.	Requires advanced technical knowledge for full comprehension.
Saura, J.R. (2021)	Data Sciences in Digital Marketing	Utilization of data science techniques for improved marketing metrics.	Enhances marketing performance through data-driven strategies.	Dependence on data availability and quality.

3 Research Methodology

3.1 Language and Libraries

This project is based on modern programming languages and libraries which can work freely, can expand easily and can also be integrated into cloud solutions. Python is particularly used as the primary back-end language because of its unique flexibility together with its rich library which makes the interaction with AWS services easier. Frameworks like Boto3, the AWS SDK for Python, allow users to perform operations on services like S3, DynamoDB, and Lambda without writing code, for example, to store files, update metadata or handle events. For the frontend, the React.js is opted due to the component-based approach that allows on building an interactive application at the frontend. Whereas communication with backend services, REST APIs and GraphQL endpoints are handled well by the React library (Biehl, 2018).

GraphQL applied also in the development of the work in that it allows for better querying and updating of data between the frontend and the backend. It reduces both the issues of over-fetching and under-fetching of data which is characteristic of older REST APIs. There are other libraries like Axios for making HTTP requests and Formik for forms which make our work easier. These languages and libraries collectively form a sound underpinning of the application, which make it relatively easy to implement and allow components of the system to communicate effectively with each other.

3.2 Tools Used

The project employs many facets of development through which integration and testing are done too. AWS Amplify is one of the preferred solutions to use to deploy and maintain services such as S3, DynamoDB, Cognito or AppSync. Amplify reduces the complications of configuration of these services so that the developers can easily work on the actual application than the service infrastructure.

For development, Visual Studio Code is used as the primary Integrated Development Environment (IDE) because of it is rich extensions offer and rich debugging capabilities. APIs are tested with Postman in order to make sure that backend endpoints successfully interact with the frontend. Git and GitHub are responsible for the version control, which helps in development and the management of the code deployment process (Ghodke and Chavan, 2024).

The manner in which AWS CloudWatch is used to monitor the resources is through gain insight into the system performance and effectively take appropriate action to remove bottlenecks. Docker is used to mimic the environment that is to be deployed which avoids future incompatibilities in the aws environment. As a whole, these tools create conditions for effective development and dependable functioning of the system.

3.3 Environment Setup

Once again, the environment setup comes as a crucial element in the context of organisational implementation, as it has an impact on the interaction between components. Python is used to configure the backend environment combined with the use of Boto3 to control cloud resources while AWS CLI has been installed. Backend service is also set up in Amplify CLI to provision easily. For the frontend development Node.js and npm are installed for using React.js and other packages (Alfadel et al., 2020).

For storing and managing data, Amazon DynamoDB is set to store metadata information such as file names, sizes and hash value. S3 buckets are created for storing files and the objects stored in them can be assigned lifecycle policies as well as server-side encryption is set for extra security and to minimize the expenses. AWS Cognito controls user sign in and sign-up features, with user pools and identity pools that controls the users access roles and privileges.

The environment of application deployment is AWS Elastic Beanstalk for the deployment of infrastructure automatically. The local mock environment is used in Amplify throughout the development of the application to emulate backend services in order to avoid implementation changes to AWS. Containers, especially Docker Containers and other similar Object containers are also used to also replicate the environment in the local distributed development, the testing phase, and the production environment.

3.4 Data Collection

Assessment of the system's effectiveness and satisfaction levels of the users require data gathering. Data collections of the usage behavior, system performance, and optimization opportunities are collected from different AWS services. Cognito login logs are also used in analyzing user authentication trends and authorization of users into the system (Djonov and Galabov, 2020). Such logs assist in recognizing the potential security threats, and improving the mechanism of authentication.

S3 returns availability and request counts related to file uploads and downloads for metrics on storage in use and user activities. For example, the data concerning the frequency of file sharing and size of files which are shared might be useful for defining trends of storage configurations. information like file hashes and upload timestamp, is stored from the DynamoDB where necessary as it is used to evaluate the system's ability to detect and prevent duplicate uploads.

In addition, services that are offered by Amazon Web Services are monitored with the aid of CloudWatch that offers performance details such as response time, CPU usage, as well as data transfer rate. These metrics allow developers to understand where the system is likely to slow down, and that it will lag how one's expectations are when it is loaded with different levels of work. This is backed by survey and form data collected from the users which in addition to providing quantitative data also offer qualitative data on the user satisfaction and ideas for change.

3.5 Limitation

There are some basic peculiarities which somewhat limit the project's overall spectrum of options and capabilities, even though it possesses quite a powerful architecture and offers the majority of the enhanced functions. This paper has one major drawback – excessive reliance on AWS services that may be effective but require particular attention regarding high operating costs. For instance, S3 is charged per usage, as is Lambda and expenses pertaining to these services need to be tracked and controlled properly. This also makes it immobile by not allowing the utilization of other cloud service providers without having to redesign ways.

Other limitation is related to absence of the advanced analytics and visualization functionalities of the germ. Although the system has the capability to log voluminous amount of data on performance and user interactions, it lacks, explicit and inherent facilities for data analysis. As a result, this hinders the administrators' attempts to draw sufficient intelligence from current system measures other than going external.

Currently security is strong though features that can be added are End to End encryption for data transfer and MFA. While using Cognito, authentication and role-based access control are supplied by Amazon however, there is no prompt for an MFA this can put the system to the risk of cyber threats.

4 Design Specification

4.1 System Architecture

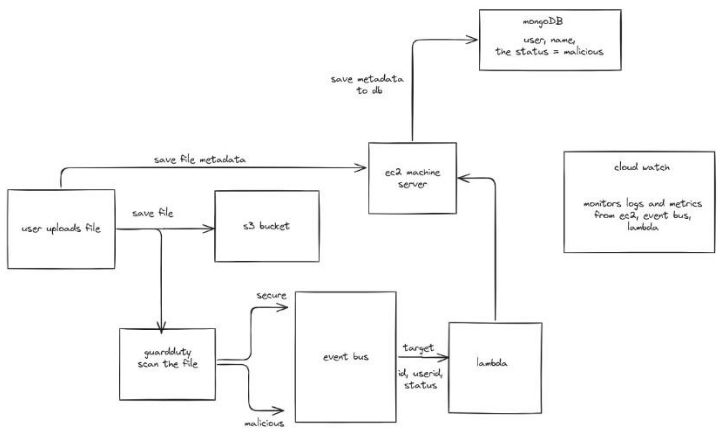


Figure 1: Architecture Diagram
(Source: Self-Developed)

This consists of a system architecture for cloud-based file sharing and storage in a seamless and secure manner. It works by stitching multiple AWS services together to provide high availability, performance, and security. File Upload via React Web Interface The user uploads files via a web interface. These files are uploaded to one S3 bucket directly and stored as a security method. An EC2 instance acts as a backend server, processing all of those requests and managing the AWS services interactions between them. MongoDB is used to store the file metadata, i.e., Details about the file, such as the user who uploaded it, the time of upload, and its status (Safe/Malicious)

AWS GuardDuty is used to analyze files that are uploaded to this architecture. When GuardDuty detects malicious files, it sends an event to an AWS EventBus Event Bus which then calls a Lambda function. This function updates the file status in the database and notifications to the user. To monitor system performance with metrics from the EC2 server, Event Bus and lambda function log. This architecture is both modular and scalable, providing an efficient method of handling the files used while maintaining a high level of data security and operational transparency.

4.2 Database and Data Flow Design

MongoDB is selected for the database for its flexibility and scalability. It captures the metadata of structured files containing file name and size, upload timestamps, user details, and security information. Metdata is essential to locate and cluster files and the usage history of the user.

The Data Flow starts when the user uploads a file via the web interface. Its metadata is sent to MongoDB through the EC2 server while the file is sent to an S3 bucket. The file within the S3 bucket is scanned for malicious content by GuardDuty. If the file is secure, it updates its status in mongoDB. For example, in case of threat detection, GuardDuty generates an event to trigger a Lambda function. This function warns the user of the threat and adds the file to the database as malicious.

4.3 Functional Requirements and Non-Functional Requirements

Functional requirements of the system provide functionality including, uploading files, file storage, scanning for malicious content, and notifying the user result of a scan. It must also enable the file sharing, so the user can create a link to access the file providing the high level of security. The other important role is keeping an indexed database of uploaded files and their metadata.

The non-functional requirements are all about performance, scalability and security. It should not impact the performance when system has concurrent uploads happening on it and will have to handle large volumes and numbers of concurrent uploads supported and running well at the same time. And it should provide seamless integration with other AWS services so you can tap into more advanced functionalities, like analytics. While still a high-level overview, security is also taken into consideration here, leveraging things like server-side encryption for S3 files, role-based access control via AWS Cognito and CloudWatch for logging across the board.

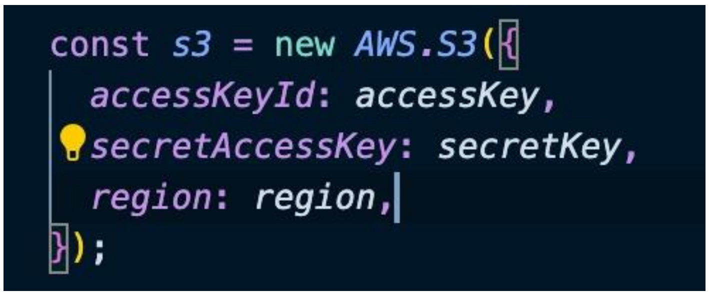
4.4 Security and Access Control

This cloud-based application is built on security and access control driven design. The project use AWS Cognito for authentication and authorization, so it is only possible for approved users to upload, or download files. That is, role-based access control, which in which you assign permissions according to the user roles, limits (or knows) the actions of the user in the system.

The encrypted files are stored in s3, side by side the project enable server-side encryption with AWS Key Management Service (KMS) in order to increase data security. GuardDuty runs scans for malware or other threats added to our bucket and when it identifies a problem, it acts through EventBus and a Lambda function. WAF (Web Application Firewall); Protects against web-based attacks to the system like SQL injection or cross-site scripting. Additionally, the application has complete logging through CloudWatch for all activities, such as file uploads, scans, and attempts to access the files. Such a layered approach provides strong security of user data and system itself.

5 Implementation

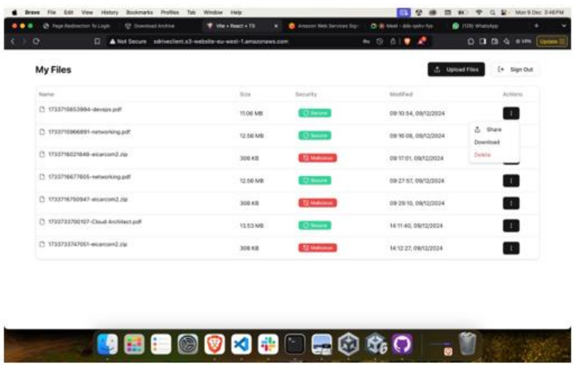
The cloud-based file sharing and storage application uses a variety of AWS services along with latest web development tools to develop a secure, reliable and scalable solution. The application features include: The development process involves multiple elements from establishing the cloud infrastructure to designing the interface for the user.



```
const s3 = new AWS.S3({
  accessKeyId: accessKey,
  secretAccessKey: secretKey,
  region: region,
});
```

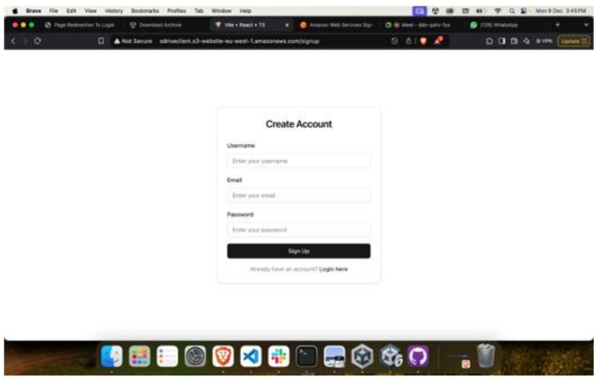
5.1 Frontend Development

React is used to develop the frontend.js, a component-based framework that works hand-in-hand with state management, offers a user-friendly interface for uploading files, viewing files you have saved, and managing your account. Features like a dashboard where the user can view files that were uploaded and their security status, upload files, and file-sharing options.



My Files

Name	Size	Security	Modified	Actions
1733716653994-devops.pdf	11.06 MB	Secure	09:10:54, 09/12/2024	⋮
1733716966891-networking.pdf	12.56 MB	Secure	09:16:08, 09/12/2024	Share / Download / Delete
1733716021848-eicarcom2.zip	308 KB	Malicious	09:17:01, 09/12/2024	⋮
1733716677605-networking.pdf	12.56 MB	Secure	09:27:57, 09/12/2024	⋮
1733716750947-eicarcom2.zip	308 KB	Malicious	09:29:10, 09/12/2024	⋮
1733733700107-Cloud Architect.pdf	13.53 MB	Secure	14:11:40, 09/12/2024	⋮
1733733747051-eicarcom2.zip	308 KB	Malicious	14:12:27, 09/12/2024	⋮

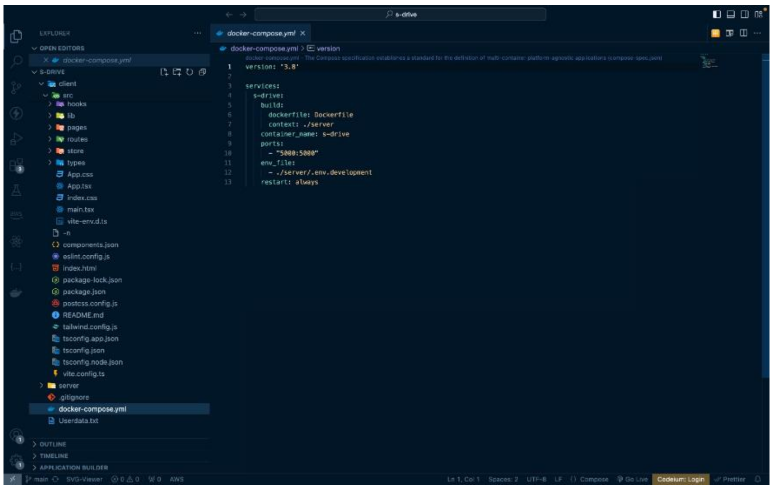


Create Account

Username: Enter your username
Email: Enter your email
Password: Enter your password

5.2 Backend Development

The backend is hosted on AWS EC2 instance which powers up the logic of the application and the API endpoints. Developed using Node. On the backend, js manages file metadata processing, authentication, and AWS service communication. Backend is run in a Docker container so that it can be deployed in the same way no matter which env it is running on.

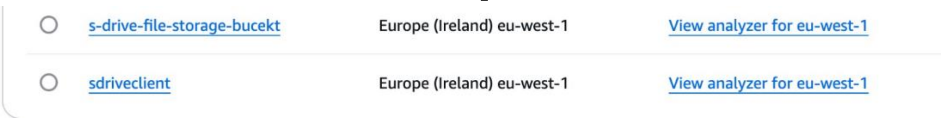


```yaml
version: '3.8'

services:
  s-drive:
    build:
      dockerfile: Dockerfile
      context: ./server
    container_name: s-drive
    ports:
      - "5000:5000"
    env_file:
      - ./server/.env.development
    restart: always
```

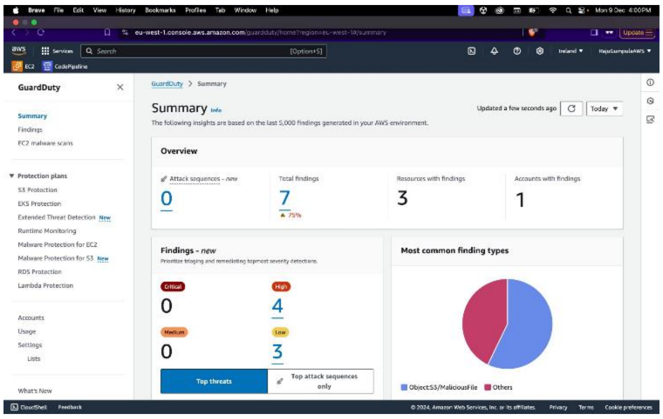
5.3 AWS Services Integration

- S3 Buckets: There are two S3 buckets configured, one for storing the uploaded files and another one for hosting static frontend website. Uploaded files are stored securely with their metadata stored for possible access later.

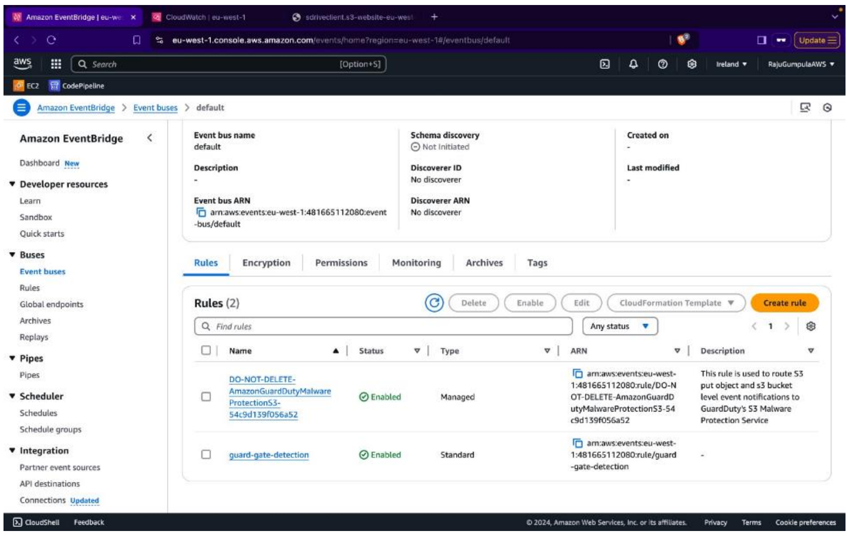


s-drive-file-storage-bucekt	Europe (Ireland) eu-west-1	View analyzer for eu-west-1
sdriveclient	Europe (Ireland) eu-west-1	View analyzer for eu-west-1

- GuardDuty: This service detects threats by scanning uploaded files for malicious content. Malicious files are reported, an event is raised, the database is updated, and the users are informed.



- EventBus and Lambda: Event generated from the GuardDuty are sent to EventBus and then hits to Lambda function. It updates the file status in the database and triggers notifications to users.



Event bus name
default

Description
-

Event bus ARN
arn:aws:events:eu-west-1:481665112080:event-bus/default

Schema discovery
Not Initiated

Discoverer ID
No discoverer

Discoverer ARN
No discoverer

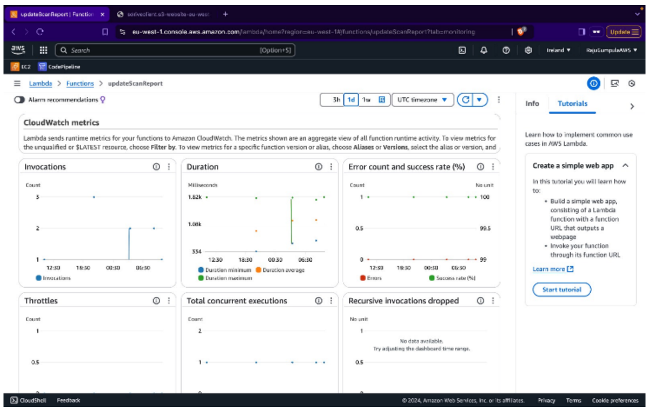
Created on
-

Last modified
-

Rules (2)

Name	Status	Type	ARN	Description
DO-NOT-DELETE-AmazonGuardDutyMalwareProtectionS3-54c9d139f056a52	✓ Enabled	Managed	arn:aws:events:eu-west-1:481665112080:rule/DO-NOT-DELETE-AmazonGuardDutyMalwareProtectionS3-54c9d139f056a52	This rule is used to route S3 put object and s3 bucket level event notifications to GuardDuty's S3 Malware Protection Service
guard-gate-detection	✓ Enabled	Standard	arn:aws:events:eu-west-1:481665112080:rule/guard-gate-detection	-

- CloudWatch: All the components are configured with CloudWatch to monitor the logs and the metrics. In identifying the performance bottlenecks and security issues.


updateScanReport | Function
sdriveclient.s3-website-eu-west
eu-west-1.console.aws.amazon.com/lambda/home?region=eu-west-1#/functions/updateScanReport?tab=monitoring
Lambda > Functions > updateScanReport
Alarm recommendations
3h
1d
1w
UTC timezone
Info
Tutorials
Invocations
Count
Invocations
Duration
Milliseconds
Duration minimum
Duration average
Duration maximum
Error count and success rate (%)
Errors
Success rate (%)
Throttles
Total concurrent executions
Recursive invocations dropped
No data available.
Try adjusting the dashboard time range.
Start tutorial
Learn more

CloudWatch metrics

Lambda sends runtime metrics for your functions to Amazon CloudWatch. The metrics shown are an aggregate view of all function runtime activity. To view metrics for the unqualified or $LATEST resource, choose **Filter by**. To view metrics for a specific function version or alias, choose **Aliases** or **Versions**, select the alias or version, and

Learn how to implement common use cases in AWS Lambda.

Create a simple web app

In this tutorial you will learn how to:

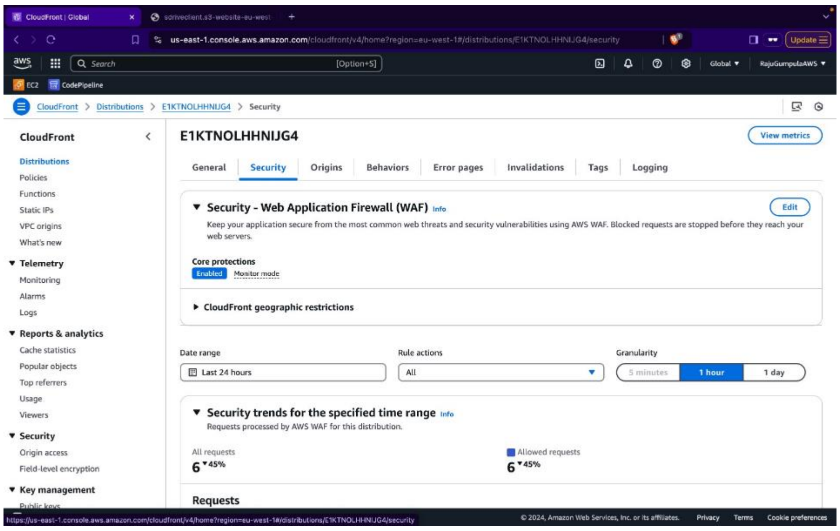
- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

5.4 Database Management

MongoDB was used as the database from our EC2 server. It contains metadata like file names, timestamps of file uploads, user information and security status. This helps users find files quicker with minimal issues. The database is designed to support both rapid query and query, integrity.

5.5 Security Measures

AWS Cognito is used for user verification to control access to the system. It enables multi-factor authentication (MFA) and role-based access control (RBAC), ensuring data security with limited access to resources. AWS WAF also helps protect the application from common internet facing threats like SQL injection, Cross-scripting and many more.


E1KTNOLHHNIJG4

General | Security | Origins | Behaviors | Error pages | Invalidations | Tags | Logging

Security - Web Application Firewall (WAF)
Keep your application secure from the most common web threats and security vulnerabilities using AWS WAF. Blocked requests are stopped before they reach your web servers.

Core protections: Enabled | Monitor mode

CloudFront geographic restrictions

Date range: Last 24 hours | Rule actions: All | Granularity: 5 minutes | 1 hour | 1 day

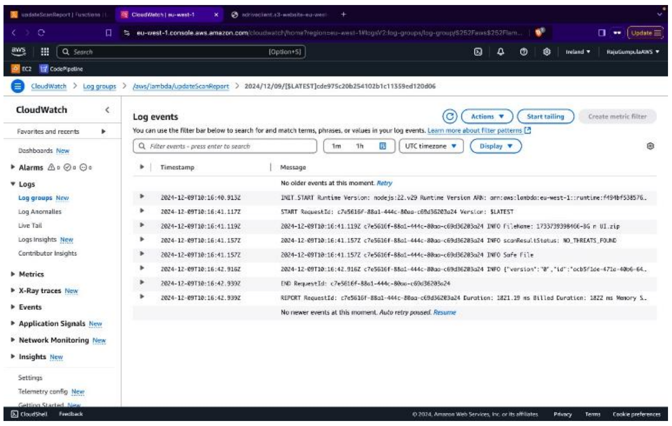
Security trends for the specified time range
Requests processed by AWS WAF for this distribution.

All requests: 6 ▾45% | Allowed requests: 6 ▾45%

Requests

5.6 Testing and Deployment

Both on development and production, check if the application is operational. The project deploys using templates stored in CloudFormation which makes it very convenient to push infrastructure for scaling purposes. These extensive implementation strategies ensure that the system is robust, secure, and user-friendly, providing a cloud-based file sharing and storage solution.


CloudWatch > Log groups > /aws/lambda/updateScanReport > 2024/12/09/[$LATEST]cde975c20b254102b1c11359ed120d06

Log events

```
Timestamp                  Message
                           No older events at this moment. Retry
2024-12-09T10:16:40.913Z   INIT_START Runtime Version: nodejs:22.v29 Runtime Version ARN: arn:aws:lambda:eu-west-1::runtime:f494bf538576..
2024-12-09T10:16:41.117Z   START RequestId: c7e5616f-88a1-444c-80aa-c69d36203a24 Version: $LATEST
2024-12-09T10:16:41.119Z   2024-12-09T10:16:41.119Z c7e5616f-88a1-444c-80aa-c69d36203a24 INFO fileName: 1733739398466-BG n UI.zip
2024-12-09T10:16:41.157Z   2024-12-09T10:16:41.157Z c7e5616f-88a1-444c-80aa-c69d36203a24 INFO scanResultStatus: NO_THREATS_FOUND
2024-12-09T10:16:41.157Z   2024-12-09T10:16:41.157Z c7e5616f-88a1-444c-80aa-c69d36203a24 INFO Safe File
2024-12-09T10:16:42.916Z   2024-12-09T10:16:42.916Z c7e5616f-88a1-444c-80aa-c69d36203a24 INFO {"version":"0","id":"acb5f1de-471a-40b6-64..
2024-12-09T10:16:42.939Z   END RequestId: c7e5616f-88a1-444c-80aa-c69d36203a24
2024-12-09T10:16:42.939Z   REPORT RequestId: c7e5616f-88a1-444c-80aa-c69d36203a24 Duration: 1821.19 ms Billed Duration: 1822 ms Memory S..
                           No newer events at this moment. Auto retry paused. Resume
```

6 Evaluation

About the cloud-based file sharing and storage application, the evaluation of its functionality, performance, security and the user experience is made. In addition to providing a secure, scalable, and user-friendly platform for file management, it also provides a comprehensive evaluation to ensure such application meets its objectives. A rigorous testing and validation were performed on each component of the system to identify areas to improve and assure optimal performance.

A 'proof of principle' of functionality was evaluated based on the ability to complete core tasks, including file uploads, metadata management, sharing files, and monitoring security. This seamless interaction between the frontend, backend and AWS services was tested. They could upload files, look at the metadata and share links without a hitch. The project confirmed GuardDuty's integration by uploading test files with malicious content, GuardDuty successfully flagged these malicious files, and quarantined them, while notifying the user.

The project evaluated the performance of the application as regards its speed, scalability and resource efficiency as a metric. Even during peak usage, the system showed consistently response time for file uploads and metadata retrieval. Consequently, content delivery was minimized through the use of AWS S3 and Cloud Front, and the EC2 instance hosting the backend scaled up nicely to handle the concurrent requests. And Docker containerization further enhanced consistency from environment to environment, helping make deployment a no fuss affair.

We conducted our security evaluation to ensure safety of user data and application resilience against cyber threats. Multiple layers of protection were offered by integrating AWS GuardDuty, WAF (Web Application Firewall) and Cognito authentication. Cognito was used to do user authentication and provide role-based access control, since only authorized users can do sensitive actions. The implemented security measures in the application secured user's data and system's integrity really well.

Feedback from test users about the application's interface, ease of use and responsiveness were used to assess the resulting user experience. Feedback on the React based frontend was positive given such features as drag and drop file uploads, and real time security status updates. Finally, users found file management via the dashboard design, with file metadata and security statuses presented, to be user friendly and effective. The evaluation did identify substantial strength and also opportunities for improvement. For example, the system being tied to AWS services is dependent and could potentially cost you a lot depending on how long you actually use the system. Furthermore, the system could also be improved by adding more user notifications like file uplinks and scan completions writes with real time.

6.1 Discussion

Design and implementation of a cloud-based file sharing and storage application utilizing AWS services is a step towards meeting the challenges of contemporary times in regard to secure and scalable content management. In this chapter, discussion will be made on the overall importance of the project, the difficulties experienced while implementation of the study, the measures taken to overcome the perceived difficulties and future developments. Regarding the theoretical framework and research objectives, the discussion also considers how design decisions and implementation strategies have been made.

Concerning the important features in the project, the safety and security of the users' data mattered a lot. About security issues mentioned in the preceding literature review, integration of other AWS services such as, GuardDuty, WAF, an Cognito authentication was made. The use of; GuardDuty to identify ingestion of malicious files and to alert for anything suspicious in real time offered significant protection to the application. This feature supports the findings in the literature that recommend early threats identification in the cloud system. Also, to customer due care, Cognito, offered a role-based access control option that limited the operations that certain users could perform. Nonetheless, the focus on AWS security brings a new dependency with potential costs, mainly in the case of small businesses. This trade-off is a common theme in the literature focusing on the trade-off between utilizing increased feature set for operational cost.

Other determinants of success included scalability and performance optimization of the project. The low latency and high reliability of the application were observed in this experiment by deploying the AWS S3 for file storage and CloudFront for content delivery. Such decisions were aimed at developing the capacity to increase client traffic without a decline in productivity, thus corresponding with the literature concerning geometrically scalable cloud structures. The management of backend with an EC2 instance helped the application to scale while using Docker for developing and running an application on different environments. These design decisions are revealing a clear appreciation of scale as a critical issue in today's cloud applications. However, evaluation showed that the system would better be stress tested with larger datasets and higher user numbers which further evaluation illustrated.

Another area of concern in the system development was in aspect of usability. A frontend based on the React environment allowed uploading files, viewing metadata information, and sharing links without efforts. In a survey of test users, there was a very high level of satisfaction with the look and feel of the dashboard that presented file information and security status well. The functionality that was by far considered to be an improvement included a drag and drop upload and real time security notifications. These results reinforce the literature's focus on user-oriented design of cloud applications. Nonetheless, some of the users pointed out that they would benefit from more frequent notifications, especially about scan completion, and more options for organizing files by folders, for example. These ideas explain why user feedback should be sharp and consistent in enhancing the cloud applications.

There were however several challenges which were experienced while implementing the above project objectives. Using AWS services in the stack meant that deeper learning of how they operated and their APIs was necessary and this was somewhat challenging. For instance, to enable GuardDuty to perform the task of identifying mischievous files, it required creation of event buses as well as Lambda triggers for notifications and status updates. These were basic steps for the functional system but necessary coordination was crucial as misconfigurations could easily corrupt the integrity of the system. Also relevant to other work is the focus on specific AWS features might lead to vendor lock-in compared to standard, open specifications, a disadvantage identified previously. Such a scenario could limit the application portability to other cloud systems in the future which may be a future consideration to be made depending on the specific cloud systems in place.

The choice of MongoDB as the database to store metadata is also a good example of weighing between functionality at the expense of cost. Flexible capabilities of MongoDB for handling unstructured data enabled this software to be used for storing the metadata of files as well as the results of the scans performed. However, managing a separate database service of its own can be costly in terms of expenses and needs to be linked with AWS resources. Further development of the system can look for ways to save cost or even link the content of AWS preload with other databases like DynamoDB.

Another major topic of conversation is whether or not the system properly meets functional and non-functional needs. From the functional standpoint, all objectives of the secure file uploading, continuous threat identification, and metadata regulation were achieved. Other sorts of requirements, which cover performance, reliability and scalability among others, can also be stated to be met to a large extent. However, areas such as notification enhancements and folders are capabilities still in their potential for improvement. These gaps show that software development is a process of repetition with the collection and analysis of feed-back for improvements.

Using AWS services also pose ethical and regulatory issues particularly with the project in consideration. Even though AWS has proper security in place, data protection with whom is up to the application developers. Adhering to privacy laws, like GDPR, is important to avoid damaging fines and loss of user trust especially when dealing with, personal user information. Nowadays data leaks are a severe issue and confronted with the present configuration of the system, files are encrypted in transmission and storage; nevertheless, other measures are available to increase compliance, for example, anonymization of files and better access restrictions.

One more general level, the work adds to the existing literature of cloud-based application research. Explaining the usage of the sophisticated AWS services, the system showcases how the business can achieve the security and scale of an enterprise-level company. However, the results also point to the importance of understanding efficient, yet inexpensive approaches that would serve as functional and inexpensive. This balance is quite crucial where small and start-up firms are involved in their business since they might not afford to adopt all the premium offered services by AWS.

7 Conclusion and Future Work

Contemporary needs in data management have been met with the creation of the cloud-based file sharing and storage application that employs AWS scaling and security as a solution and fundamental features of the application have highlighted possible areas for future improvement. In this chapter, the success of the project is summarized and its drawbacks are discussed, along with possible directions for its evolution.

The project was initiated to meet the need for better security and performance of the platforms as interactions become dominated by the use of data. The system proved that with adoption of AWS services including S3, Guard Duty, Lambda, Cognito, and Cloud Front the potential of cloud computing is realizable in terms of an efficient, secure, and easily scalable application. Under the support of React-based interface, users could upload files, open files of others for viewing, downloading and editing, as well as monitor the real-time status related to the security of the files. The application was built based on modern features such as real-time virus detection and metadata handling to meet the goals at the start of the project and lay solid ground for further improvements of cloud applications.

Perhaps one of the greatest strategic successes here is the overall sound security architecture of the system. In cases of detecting anomalies, the application provided a measure through GuardDuty while WAF was implemented to filter out any malicious traffic. The first important advantage was an identification of malicious files and the ability to mark them and second, utilization of the role-based access control by Cognito reduced major security concerns highlighted in literature. These measures can be said to be in harmony with the common best practices in the protection of data as well as safety of the users- hereby setting the system above the rest with an assurance of safety for storage of files and sharing. However, the use of AWS services as a primary dependence brings a risk to vendor lock-in that should be discussed further.

This focus on scalability was just as successful as the rest of the project. Here, Amazon S3 file storage uses and CloudFront for content delivery were implemented to keep blazing, high demanding users. The backend that run as an EC2 instance and that is Dockerized was flexible and efficient.

Nevertheless, the spare stress testing in extraordinary circumstances does not allow for verifying the system's potential to the maximum extent. Possible work for later would be to make a lot of experiments with more and different data as well as higher traffic loads to see how it performs.

In the course of the evaluation phase of the project, the system was proven to be successful in the functional and non-functional requirement objectives, but some constraints were noted as well. For instance, there were no other features like folder system management and multiple file operations at once had constrained the user accessibility. Also, real-time malware scanning was a good start, but users claimed they require even better notifications like an email or push notification about the scans, the results, and the usage of files. These gaps are potential grounds for improvement for the subsequent modifications of the system from the usability and functionality perspectives.

On the one hand, the use of AWS services turned out to as a major advantage. As it facilitated the execution of those features smoothly, it also brought and additional level of challenges in term of set up and administration. For instance, configuring GuardDuty took elaborate efforts in order to connect with event buses and Lambda functions flawlessly. Probably the most profound technical hurdles mean that proper documentation and highly trained people are required to understand how the cloud-based systems work and how best to operate them. Furthermore, the nature of the work is heavy on AWS-specific services, which opens the question of portability. Perhaps future updates can target extension of support for multiple cloud scenarios so as to eliminate or at least minimize dependency on a particular cloud vendor as well as service.

In terms of cost and expense sensitivity, the project revealed an important balance that may be achieved while using modern cloud services. Despite the powerful functionality of provided services, using AWS may turn into an expensive experience for businesses especially for start-ups. Introducing new less-costly models like hybrid clouds or different open-source software could also extend the presence of the system among the society. This corresponds well with the larger vision of extending the use of safe and highly available file sharing solutions to everyone.

Nevertheless, it was seen that there are some disadvantages in current concept and gives the following list of shortcomings: the lack of features like creation of folders, using and availability of real time alert. The inclusion of these characteristics in subsequent versions would enhance the overall customer fulfilment level drastically. The standpoints of this project are not simply the constraints it overcomes but implications that are well beyond this notion of bounded problem solving. As a contribution to current developments in the use of cloud-based applications, the system helps to address some inherent challenges of file sharing and storage. This will show how small business and enterprises can deploy cloud technologies to enjoy enterprise level performance, security, and sustainability. Nevertheless, the study also establishes that there is a need to keep developing new ideas to meet new challenges and user needs. For example, the system will expand its security features to better respond to incoming threat models in the future, including application of artificial intelligence and fully automated responses.

The next studies on this project could be aimed at several perspectives. First, as it was pointed out during the evaluation phase, improving the UI and the overall user experience further by incorporating more file management functionality as well as features like organizing the folders, batch operations, and real time notifications would also resolve the problem. Second, possibility of going further and enhancing the system with addition capabilities: usage patterns and threats patterns might be useful to the user and system administrators. Third, developing multi-cloud support would improve the system's portability and would eliminate the issues of vendor lock-in as a drawback. Fourth, incorporating into the system examining the cost-saving methods, for instance, that using AWS free tier or incorporating free open-source tools can expand the facilities available to enterprises and especially small business and start-ups.

References

Abdullah, P.Y., Zeebaree, S.R., Jacksi, K. and Zeabri, R.R., 2020. An hrm system for small and medium enterprises (sme) s based on cloud computing technology. International Journal of Research-GRANTHAALAYAH, 8(8), pp.56-64. https://www.academia.edu/download/81575215/873.pdf

Alam, T., 2021. Cloud-based IoT applications and their roles in smart cities. Smart Cities, 4(3), pp.1196-1219. https://www.mdpi.com/2624-6511/4/3/64/pdf

Alfadel, M., Costa, D.E., Mokhallalati, M., Shihab, E. and Adams, B., 2020. On the threat of npm vulnerable dependencies in node. js applications. arXiv preprint arXiv:2009.09019.

Basu, S., Bardhan, A., Gupta, K., Saha, P., Pal, M., Bose, M., Basu, K., Chaudhury, S. and Sarkar, P., 2018, January. Cloud computing security challenges & solutions-A survey. In 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 347-356). IEEE.

Biehl, M., 2018. Serverless GraphQL APIs with Amazon's AWS AppSync (Vol. 8). API-University Press.

Borra, P., 2024. Comprehensive survey of amazon web services (AWS): techniques, tools, and best practices for cloud solutions. International Research Journal of Advanced Engineering and Science, 9(3), pp.24-29.

Botonakis, S., Tzavaras, A. and Petrakis, E.G., 2020. iSWoT: service oriented architecture in the cloud for the semantic web of things. In Advanced Information Networking and Applications: Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA-2020) (pp. 1201-1214). Springer International Publishing. https://www.researchgate.net/profile/Euripides-Petrakis/publication/340238535_iSWoT_Service_Oriented_Architecture_in_the_Cloud_for_the_Semantic_Web_of_Things/links/60b4eba945851557bab32023/iSWoT-Service-Oriented-Architecture-in-the-Cloud-for-the-Semantic-Web-of-Things.pdf

Djonov, M. and Galabov, M., 2020, June. Real-time data integration AWS Infrastructure for Digital Twin. In Proceedings of the 21st International Conference on Computer Systems and Technologies (pp. 223-228).

Ghodke, G.M. and Chavan, T., 2024. An Overview of Git. International Journal of Scientific Research in Modern Science and Technology, 3(6), pp.17-23.

Giménez-Alventosa, V., Moltó, G. and Caballer, M., 2019. A framework and a performance assessment for serverless MapReduce on AWS Lambda. Future Generation Computer Systems, 97, pp.259-274.

Gupta, B., Mittal, P. and Mufti, T., 2021, March. A review on amazon web service (aws), microsoft azure & google cloud platform (gcp) services. In Proceedings of the 2nd International Conference on ICT for Digital, Smart, and Sustainable Development, ICIDSSD 2020, 27-28 February 2020, Jamia Hamdard, New Delhi, India.

Hashmi, S.A., Ali, C.F. and Zafar, S., 2021. Internet of things and cloud computing-based energy management system for demand side management in smart grid. International Journal of Energy Research, 45(1), pp.1007-1022. https://www.researchgate.net/profile/Saima-Zafar-4/publication/346482467_Internet_of_things_and_cloud_computing-based_energy_management_system_for_demand_side_management_in_smart_grid/links/65cdbd311bed776ae3610720/Internet-of-things-and-cloud-computing-based-energy-management-system-for-demand-side-management-in-smart-grid.pdf

Kadaskar, H.R. and Kamthe, V.R., 2024. An overview of AWS. International Journal of Scientific Research in Modern Science and Technology, 3(7), pp.22-30.

Kewate, N., Raut, A., Dubekar, M., Raut, Y. and Patil, A., 2022. A review on AWS-cloud computing technology. International Journal for Research in Applied Science and Engineering Technology, 10(1), pp.258-263.

Kotha, S.K., Rani, M.S., Subedi, B., Chunduru, A., Karrothu, A., Neupane, B. and Sathishkumar, V.E., 2022. A comprehensive review on secure data sharing in cloud environment. Wireless Personal Communications, 127(3), pp.2161-2188.

Mansouri, Y. and Buyya, R., 2020. Data Access Management System in Azure Blob Storage and AWS S3 Multi-Cloud Storage Environments. Handbook of Research on Intrusion Detection Systems, pp.130-147.

Marinescu, D.C., 2022. Cloud computing: theory and practice. Morgan Kaufmann. https://www.dcs.bbk.ac.uk/~Dell/teaching/cc/book/cctp/cctp_ch2a.pdf

Mykhaylova, O., Korol, M. and Kyrychok, R., 2024. Research and analysis of issues and challenges in ensuring cyber security in cloud computing. Cybersecurity Providing in Information and Telecommunication Systems II 2024, 3826, pp.30-39.

Park, S.J., Lee, Y.J. and Park, W.H., 2022. Configuration method Of AWS security architecture that is applicable to the cloud lifecycle for sustainable social network. Security and Communication Networks, 2022(1), p.3686423.

Poornalinga, K.S. and Rajkumar, P., 2016. Continuous integration, deployment and delivery automation in AWS cloud infrastructure. Int. Res. J. Eng. Technol.

Prajapati, P. and Shah, P., 2022. A review on secure data deduplication: Cloud storage security issue. Journal of King Saud University-Computer and Information Sciences, 34(7), pp.3996-4007.

Rai, B., 2024. A Review of Advancing Sustainable Practices in International Business Through "AWS and Other Cloud Services.

Saini, R. and Behl, R., 2020, January. An Introduction to AWS-EC2 (Elastic Compute Cloud). In ICRMAT (pp. 99-102).

Sajid, R., Mumtaz, G., Zaidi, H.Z. and Mubeen, Z., 2024. Securing DynamoDB: In-Depth Exploration of Approaches, Overcoming Challenges, and Implementing Best Practices for Robust Data Protection. Journal of Computing & Biomedical Informatics, 7(02).

Sanne, S.H.V., Overcoming Challenges in Migrating Legacy Applications to AWS Cloud. J Artif Intell Mach Learn & Data Sci 2022, 1(1), pp.620-625.

Saura, J.R., 2021. Using data sciences in digital marketing: Framework, methods, and performance metrics. Journal of Innovation & Knowledge, 6(2), pp.92-102. https://www.sciencedirect.com/science/article/pii/S2444569X20300329

Sharma, S. and Chaturvedi, R., 2021. Optimizing Scalability and Performance in Cloud Services: Strategies and Solutions. ESP Journal of Engineering & Technology Advancements (ESP JETA), 1(2), pp.116-133.

Singh, A. and Aggarwal, A., 2024. Artificial Intelligence Self-Healing Capability Assessment in Microservices Applications deployed in AWS using Cloud watch and Hystrix. Australian Journal of Machine Learning Research & Applications, 4(1), pp.84-97.

Singh, A.P., Rai, P. and Dixit, A.K., 2021, December. A Review On Low Latency Web Data Packet Distribution Using CloudFront. In 2021 First International Conference on Advances in Computing and Future Communication Technologies (ICACFCT) (pp. 258-262). IEEE.

Vadlamani, S., Kankanampati, P.K., Agarwal, R., Jain, S. and Jain, A., 2024. Integrating cloud-based data architectures for scalable enterprise solutions. International Journal of Electrical and Electronics Engineering, 13(1), pp.21-48. https://www.academia.edu/download/118865458/Cloud_Based_Data_Architectures_for_Scalable_Enterprise_Solutions_1_.pdf

Viikari, V., 2022. Integrating a Monolithic Web Service Into a Microservice-Based Cloud Platform. https://trepo.tuni.fi/bitstream/handle/10024/139229/ViikariValtteri.pdf?sequence=2

Yanamala, A.K.Y., 2024. Optimizing data storage in cloud computing: techniques and best practices. International Journal of Advanced Engineering Technologies and Innovations, 1(3), pp.476-513. https://ijaeti.com/index.php/Journal/article/download/389/401

Yang, H., Pan, H. and Ma, L., 2023. A review on software defined content delivery network: a novel combination of CDN and SDN. IEEE Access, 11, pp.43822-43843.

Zhang, J. Y., and Zhang, Y. (2024). Quantitative DevSecOps Metrics for Cloud-Based Web Microservices. IEEE Access. https://ieeexplore.ieee.org/iel8/6287639/6514899/10735195.pdf