Kwaghe International
Journal of Sciences and Technology

ISSN: 1595-7977

Index: Harvard, Boston, Sydney University, Dimensions, Lens, ResearchGet Scilit, Semantic, Google Scholar, Base etc

https://doi.org/10.58578/KIJST.v2i3.7228

Cryptographic System for Mobile Application (Automated Resume Builder)

Bartholomew Idoko, Sampson Agada,
Okoro Denis Isah, Chika Patricia Bossah
Federal Polytechnic, Ohodo, Enugu State, Nigeria
bartholomew.idoko@gmail.com

Article Info:

Submitted:	Revised:	Accepted:	Published:
Jul 15, 2025	Aug 3, 2025	Aug 15, 2025	Aug 20, 2025

Abstract

This paper presents a secure and user-friendly approach for developing an automated mobile application system, using a resume builder as a case study. The proposed system automates the construction of resumes by utilizing applicants' information as input, allowing users to create, edit, delete, read, and save resumes in PDF format, supported by login and signup via OTP verification. To enhance security, the study introduces a two-factor authentication (TFA) scheme that integrates a cryptographic-compatible device and a password, offering stronger protection against risks such as communication breaches, device or server vulnerabilities, and offline or online credential attacks. The TFA is implemented through shared access signature (SAS) message authentication or other PIN-based authentication methods. The system architecture incorporates an enhanced cryptographic framework adaptable to various password-based client–server authentication protocols, reducing reliance on less secure single-layer password systems. Data encryption is handled using the Advanced Encryption Standard (AES), chosen over 3DES for its superior processing efficiency, while the Message-Digest Method (MD5)

algorithm is used to hash user-defined encryption keywords. All server-side data, including encryption keys, remain encrypted, ensuring that unauthorized access yields no advantage. By enabling users to encrypt and decrypt data with AES and securing encryption keys via MD5 hashing, the system improves both privacy and security in mobile applications. The study contributes to secure software design by demonstrating how cryptographic methods can be modularly integrated into mobile systems, addressing the cybersecurity gaps of conventional job search and resume platforms.

Keywords: Encryption; Cryptography; Authentication; Security; Privacy; Hashing; Keys

Introduction

In recent years, there has been a persistent tendency among young people who want to be highly trained and talented to seek higher education. The management of knowledge and the transmission of information in education have been significantly impacted by new technologies, particularly the internet. One of the most well-liked and interesting aspect is the web portal's knowledge-management system as well as cloud computing [1]. Universities have been at the forefront of website creation, which further sparked the creation of online portals to offer more beneficial access to information sources.

Different apps or services are available on portals to address various issues. Providing access to and exchange of information through the internet is one of the goals of web portals. For instance, in a university, new students in the faculty require access to information resources in order to choose from a variety of courses and different areas and majors that are offered. The knowledge portal, which should include pertinent information about the requirements of the users and students, may be used to satisfy this need [2].

One of the severe issues facing both developed and developing nations today is the rising number of young people and graduates without jobs. Through the creation of online job portals, the Internet has altered how people hunt for jobs [3]. A job portal is a particular kind of website portal that offers a quick and easy way to look for open employment online. This study will examine several web/job portal models, but it will specifically examine job portals as a knowledge management system built on a common foundation. The data and information on open positions that the jobless or job searchers

require will be the primary emphasis of this initiative bearing in mind that the cyber security goals will be achieved [4].

This research aids in automating the creation of resumes in a secured manner for certain people. The system may be configured to produce an acceptable resume in accordance with qualifications by minimizing the requirement for thought. Often, people become confused while writing resumes, especially fresh graduates searching for work. They are not very clear about what should be on a resume and what information should be included. This research will offer a simple method of creating a resume that will appear professional. The proposed application is user friendly and requires less human efforts. People just need to complete out the form's mandatory fields, such as their educational background, areas of interest, talents, and employment history, for example. The system will save the user's information and create a well-structured resume based on it. A resume may be created by users in any format. It is mostly focused on format, requiring only the selection of a preferred template and the provision of a few basic elements, which the resume builder program will effortlessly turn into the final resume.

Cryptography can help achieve certain security objectives, such as data privacy and non-alteration. Today, cryptography is widely employed due to its many security benefits. These include; Confidentiality, Integrity, Authenticity, Non-repudiation, Access control, Service Reliability and Availability. [5, 6].

Security Issues in Automated Mobile Applications

The internet and the cloud is susceptible to security issues related to automated mobile applications for the majority of systems and technology. One network that needs to be secured is the one that links the systems in a cloud. Users run the risk of their data being accessed by hackers or other uninvited parties when they upload it to cloud servers. Encrypting data before it reaches cloud servers is the only reliable solution to address the issue of cloud computing or else, providers will never be guaranteed of data protection. Some of the pertinent security issues from automated mobile applications include:

i. Password Attack

People attack database in different ways, to gain access, the user name and password must be compromised. To prevent this type of attack and others, we use

Mongoose sanitizer package as middleware to reject inputs that are queries in the database, for example if someone try to input {email: $gte: “”} which is a valid query for the database because a correct password will take any user that meets this condition and grant them permission, however, mongoose sanitizer would remove the $ sign from the query to make it invalid. Mongoose-sanitizer will automatically go over each field and sanitize them before storing them in the database for document created using MyMongooseSchema [7].

ii. XSS

XSS (cross-site scripting) attack insert malicious code into otherwise reliable and benign websites. XSS attacks occur when a hacker uses an online application to deliver malicious code, usually in the form of a browser side script, to a different end user. Anywhere a web program accepts user input without validating or encoding it into the output it generates, these attacks can be successfully carried out.

A malicious script can be delivered by an attacker using XSS to an unsuspecting user [8]. The script should not be trusted, yet the end user's browser will nonetheless run it. This is because it thinks the script is from a trustworthy source, the malicious script can access any cookies, session tokens, or other sensitive data stored by the browser and used with that website. These apps have the power to alter the content of HTML pages entirely. JavaScript is a common example of dangerous material that is delivered to a web browser, but it may also be HTML, Flash, or any other sort of code that the browser is capable of executing. Attacks based on XSS can take a wide range, although they frequently involve sending sensitive information to the attacker, such as cookies or other session data, rerouting the victim to an attacker-controlled website, or damaging the user's computer while impersonating the vulnerable site.

iii. HPP (HTTP Parameter Pollution)

An attacker can create an HTTP request using the Web attack evasion method known as HTTP Parameter Pollution (HPP) in order to modify or extract concealed information [9]. The foundation of this evasion method is the division of an attack vector across several occurrences of a parameter bearing the same name. Each web application delivery platform may handle HTTP parameter manipulation differently since none of the pertinent HTTP RFCs describe its semantics. Some environments, in particular, handle these requests by concatenating the values obtained from each occurrence of a parameter name inside the request. The attacker takes use of this characteristic to get beyond pattern-

based security measures. In req.query and/or req.body, HPP ignores array arguments and only chooses the final parameter value. After adding the middleware, a successful invasion must definitely take place. The whitelist option enables you to define parameters that HPP must not modify. Typically, certain route parameters are utilized as arrays on purpose.

iv. Privacy

Protecting user privacy is one of the most important cloud computing security goals. Risks are difficult to avoid in cloud computing because it's a shared environment that relies on a shared infrastructure. As a result, information will be susceptible to unauthorized access. Stated differently, there is a great deal of difficulty in sharing cloud computing resources while preserving client privacy. The data needs to be encrypted before it is sent to cloud servers. This is an essential phase in resolving the issue [10].

v. Access control

Access control implies that the value of the data we save locally or online must be shielded from alteration or access, thus the best way to address this problem is to always maintain the data encryption techniques to protect easy access. [11].

Related works

Many vulnerabilities exist in most software systems, making it possible for hackers to obtain data from cloud users. Concerns like security, privacy, and access control are central to cloud computing in the computer world [12, 13]. Numerous research have been undertaken in an effort to strengthen user data security and privacy on cloud servers. The security and privacy concerns, and a novel approach to safeguard user data has been suggested, involving the utilization of AES and RSA encryption algorithms for encryption and decryption [14]. The proposed approach is based on data partitioning, whereby the data is first divided and subsequently encrypted using RSA and AES before being delivered to the server. Following the encryption process, many servers will receive the divided data. Users can download the partitioned data and apply the same method to decrypt it again if they want their data back.

Another novel approach was suggested for cloud storage data security [15]. The RSA encryption algorithm was used in the study to encrypt the data. The RSA technique was used to improve cloud security and privacy while also safeguarding data. The RSA

technique was utilized to encrypt the original data. During the encryption process, RSA produces the private keys that are needed for both encryption and decryption. The original encrypted files were uploaded together with those keys. When private keys were uploaded to servers, MD5 was used to confirm their contents. The one-time password for authentication, the MD5 hashing method for hashing one-time passwords, the RSA algorithm for secure communication, and the AES algorithm for secure file encryption were all suggested in the work [15]. The original file was encrypted in the study using the AES algorithm before the data is uploaded to the cloud. The name of the user account and the key were stored in the system server's database table. The user account name was additionally hashed using MD5 hashing prior to being inserted. This ensures that an unauthorized individual cannot access the system server's database table and obtain the key needed to decrypt a file on behalf of a user. Consequently, a file's key becomes secure and hidden.

The examination of the performance metrics of various encryption algorithms for data transmission in a study that compared their effectiveness was performed [16]. The analysis took into consideration variables such as computation time, memory usage, and output byte count. The simulation's findings compared three encryption methods; AES, DES, and RSA using the same text file for five experiments. AES and DES produced the same amount of bytes regardless of the text file size. When compared to the AES and DES algorithms, it was shown that the RSA algorithm produces far fewer bytes. When compared to AES and DES, RSA takes longer to process data. It was determined that the AES and DES algorithms have very modest encryption times and the DES algorithm takes the least amount of time to encrypt data, based on the text files that were used and the simulation results that were achieved. When it comes to encryption times and memory use, RSA requires the longest and produces the fewest bytes in output. Furthermore, a thorough analysis of symmetric key encryption techniques including DES, 3DES, AES, and Blowfish was provided [17]. It was determined that private key encryption requires less memory than public key encryption algorithms and operates more quickly than public key encryption methods like RSA, etc. Additionally, it was shown that private key encryption has a higher level of security than public key encryption. In a similar vein, [18] concentrated on Blowfish and AES, two popular private key encryption algorithms. The effectiveness of these approaches was evaluated and contrasted. The investigations' findings were presented to determine how well these techniques worked. Furthermore, a thorough performance

comparison of the four most used encryption algorithms; DES, 3DES, AES, and Blowfish was presented [19]. To evaluate the encryption and decryption speed of the algorithm, the contrasts were performed by processing varying sizes of information blocks using several distinct settings. The programming language used for the simulation setup was C#. The results of this study demonstrated that blowfish outperforms the other encryption techniques in terms of performance.

Another author states that the majority of symmetric algorithms have a high encryption ratio [20]. That it is preferable to use A5 versions when creating Stream Cipher because it seems that AES is superior in symmetric algorithms, while RSA is superior in asymmetric algorithms. The researcher also observed that whilst Twofish and Threefish were discovered to be less secure than AES, Blowfish and RC versions were shown to be more sensitive than others.

The two most popular symmetric encryption methods, data encryption standard (DES) and advanced encryption standard (AES), were examined [21]. The avalanche effect which occurs when a single bit of plaintext variation maintains the key unchanged, the memory requirements for implementation, and the simulation time required for encryption were the factors that were examined. The term "avalanche effect" refers to the characteristic of any encryption technique wherein a slight modification in either the plaintext or the key should result in a significant modification in the cipher text. While DES requires more memory and takes longer to simulate than AES, AES has a higher avalanche effect than DES, indicating its superiority over DES. AES was regarded as the best encryption method for communications sent between objects via chat channels and is useful for objects that include financial transactions. AES encryption was also termed the quickest encryption technique that is both scalable and adaptable [5]. Additionally, AES is simpler to use. AES method requires less memory than other symmetric algorithms like DES, 3DES, and blowfish [10]. AES has an extremely high level of security because its algorithm employs a 128, 192, or 256-bit key [22]. AES exhibits resilience to a range of assaults as a result, it is seem as a highly secure encryption technique [5]. AES has been developed to operate quickly and effectively in both software and hardware, as well as to accommodate small devices like smartphones [23]. Because AES employs a longer key and a larger block size, it will eventually provide more security. In contrast to other symmetric encryption techniques, the AES encryption algorithm has no flaws or restrictions and boasts excellent performance and low storage requirements [10].

Proposed Model Design and Implementation Processes

In this paper, we proposed a model that could enhanced the security of an automated Resume builder. This secured automated system is recommended for its effectiveness, when compared with the conventional resume system. The model utilizes frameworks and tools such as; HTML, CSS, JAVASCRIPT, POSTMAN, MONGO DB /MONGO DB COMPASS, VS CODE & NODE JS. We used JSON Web Token (JWT for short) as our proposed Internet standard to allow for the creation of data with optional encryption and signatures, whose payload contains JSON and to make a variety of claims. Either a public/private key pair or a private secret is used to sign the tokens. Furthermore, we deploy PUG as the active server that is linked directly to the database so as to define dynamic components depending on the data while reusing static web page elements. HTML is used at the client side. [24, 25]

Explored Algorithms for the System Hosting

Our choice of Heroku is because it is a polyglot platform as a result of its capabilities that enable developers to create, deploy, and grow applications in a consistent way across the majority of languages. Hence the Resume builder application can operate a domain that is used to direct HTTP requests to the appropriate application container or dyno. [26, 27]

A "dyno grid" made up of many servers serves as the distribution point for each of the dynos. Data from authorized users' application repository are handled by Heroku's Git server as shown in fig.1.

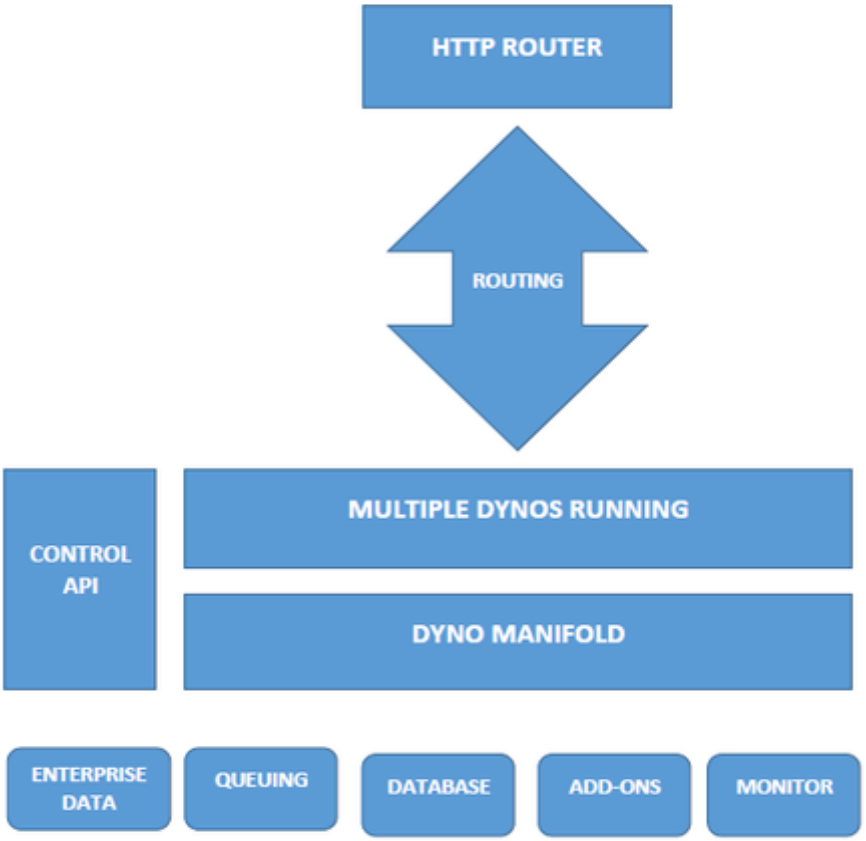

Fig. 1 Heroku Hosting process.

System Design

During the design phase of the automated resume builder app development, Figma was utilized as the main design tool. Figma is a web-based platform that is commonly used for creating UI/UX prototypes and animations before implementing the designs using code or no-code builders. In order to bring the designs to life, simple CSS syntax was utilized to style the HTML blocks [25]. A white color was chosen for the background, while a darker shade of red was used for error boxes and blue/green for buttons. A green color was selected for the "mark as successful". Overall, the design of the application has a cohesive and visually appealing look in order to create a positive user experience and increases the chances of conversion as shown in fig. 2 and 3.

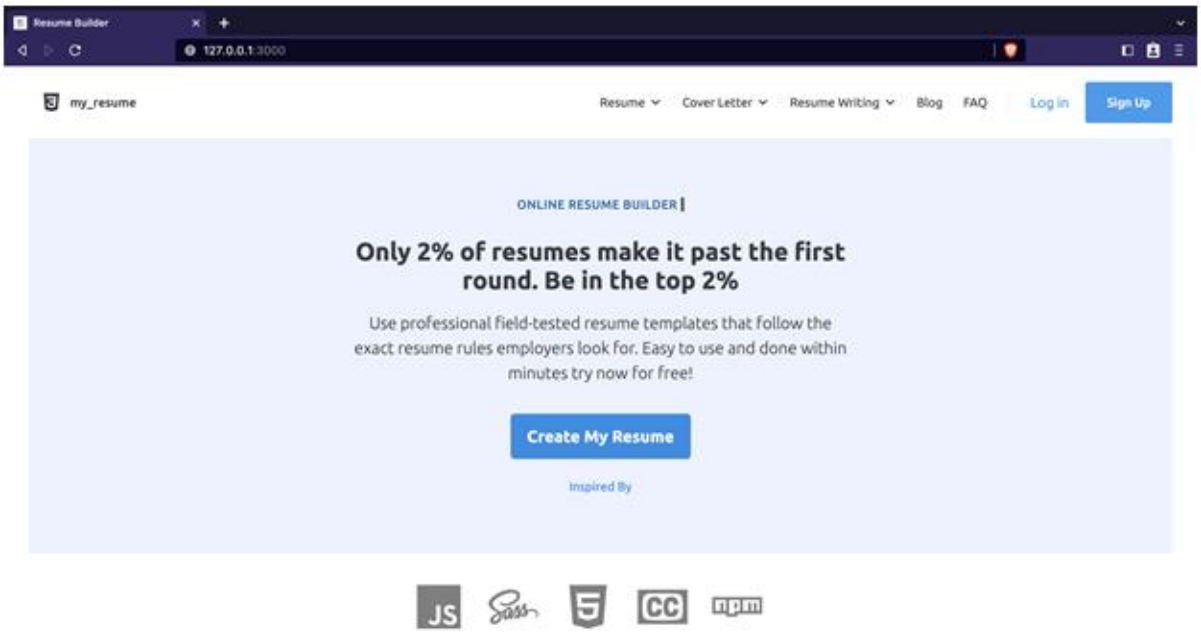

Fig. 2. Automated Resume App Builder

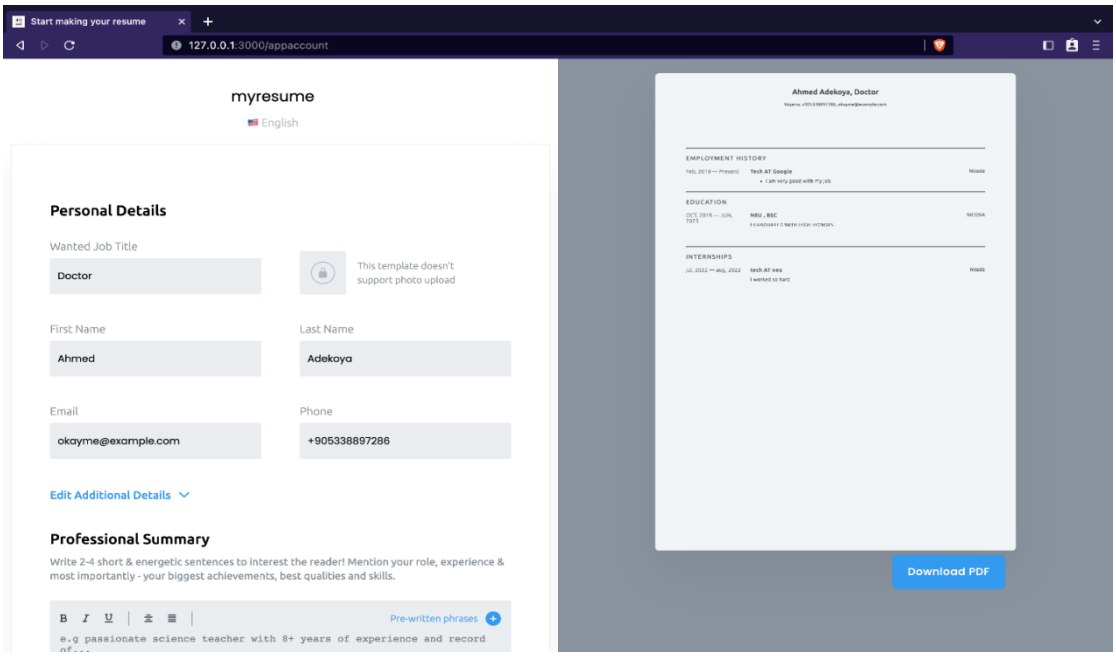

Fig. 3. Auto update Resume builder template with enhanced security features.

System Flow Chat and Function

The proposed system is developed to operate at both the frontend and backend as represented in fig. 4.

The front-end structure of the application development process was implemented using PUG, CSS, and JavaScript. This structure includes:

- A single PUG file containing a root div element that is used to render all components within the system.
- A single CSS file that holds all the styles for the PUG elements returned by a component.
- A component folder that contains all the different components of the application, such as the navigation bar, art pieces, etc. These components are all accessible to each other and return HTML elements with defined classes.
- A routes folder that includes all the different interfaces, including a default page for routes that are not defined or do not exist (404 page) and 500 for internal server error. Each route can call a component and pass in data to make the component dynamic.
- By using this structure, the application is able to maintain a clear separation of concerns and make it easier to manage and update the different components and pages of the website.

However, the backend structure of the application utilizes JavaScript's node js as the primary programming language. This enables effective communication between the client and server sides of the application, allowing for the execution of algorithms and computations, modification of PUG component values and attributes, and much more. Data is transmitted between the input PUG components and the server, and is processed and handled differently depending on the specified routes. To facilitate this, the Express library is utilized to create different routes on the localhost port, enabling the use of JavaScript callback functions to efficiently handle errors, responses, and requests for both the programmer and the application [28]. For example, on the home route of the localhost, the server establishes a connection to the database and outputs all of the data in JSON format. When a user accesses the home page of the application, a GET request is made to the home route of the localhost, and the JSON data is retrieved and processed at the front end to create seller cards. This system allows for the efficient handling and processing of data transmitted between the client and server, enabling the smooth functioning of the application [29].

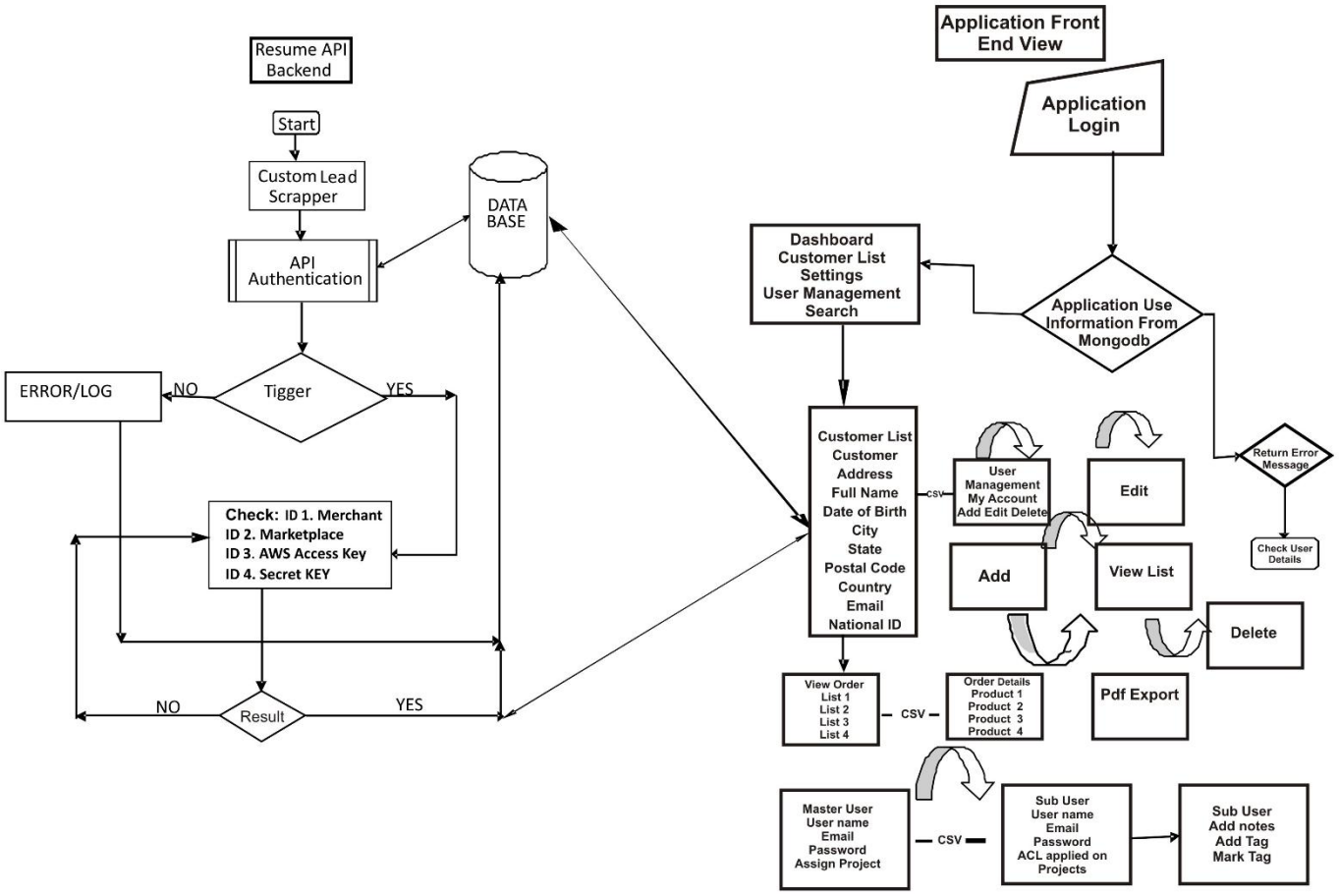

Fig. 4. Flowchart diagram showing the automated App. Functions.

Results from Findings

In this system, we used MongoDB Atlas. The MongoDB database was used to store non-image data, such as user names, username, and email, personal details. The MongoDB database was structured using a decentralized system, with one collection for user's data. To create a user, a user must register on the signup page. To create a resume page, a user must register as a role of user and use the dashboard page. To download a resume pdf, a user must click the "download pdf" button.

This structure allows for easy association of data items from different collections through the use of a reference ID, which is the user's unique ID. When a user is created, their unique ID is passed down to the user dashboard. This enables sorting of resume and data to their respective user as shown in fig. 5

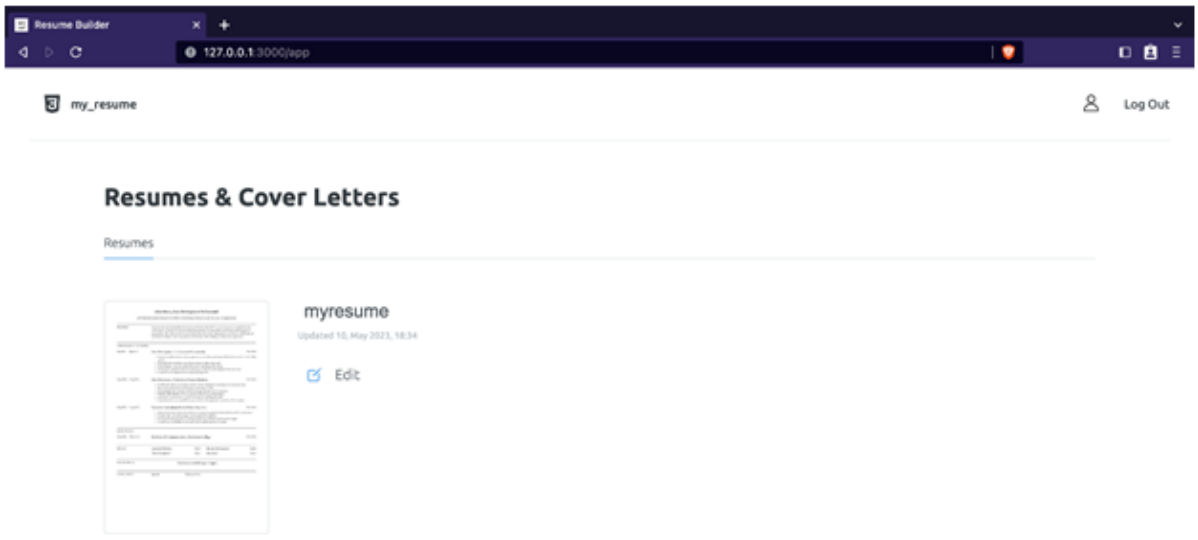

Fig. 5. Resume App. Interface.

The result of the analysis of the resume encryption and decryption time taken by both 3DES and AES algorithms is presented in table 1 and table 2. The results of the findings has proven that AES takes less time for both encryption and decryption on larger file sizes when compared to 3DES.

Table 1: Triple Data Encryption Standard (3DES) Encryption and Decryption Time

File Size	Time-Start (ms)	Time- End (ms)	Time Diff. (ms)
10 mb	445	898	453
5 mb	662	886	224
1 mb	563	714	151
50 kb	223	334	111

Table 2. Advanced Encryption Standard (AES) Encryption and Decryption Time

File Size	Time-Start (ms)	Time- End (ms)	Time Diff. (ms)
10 mb	464	610	146
5 mb	523	686	163
1 mb	492	673	181
50 kb	368	560	192

Encrypting data before uploading it to other servers is the only way to ensure that the data on the primary server is safe. The original data is encrypted using the AES encryption technique in the proposed system architecture, and the private keys which is hashed using MD5 was utilized to encrypt and decrypt the data [30, 31]. Since every piece of data on the server, including the private keys, is encrypted, the purpose of hashing the

encryption keys is to secure them while they are being held on the server. As a result, any unauthorized access to the server will not be advantageous. The suggested encryption technique can be used to guarantee that the data is safer while it is stored on the servers. Only the owners who possess the decryption technique will be able to access the data, as they are the ones who encrypted it. To sum up, the user will benefit from the anticipated recommended design execution in the following ways:

i. Improved privacy and security features for mobile applications
ii. Safeguarding user data through the use of the AES and MD5 algorithms for encryption and decryption;
iii. Novel approaches for server-side key storage for encryption and decryption based on the MD5 hashing technique;
iv. Improved data access control

Conclusion

This paper outlined the value of a great enhanced cryptographic resume builder and provided instructions on how to write one that is both official and attractive. The suggested program is highly beneficial and simplifies the resume creation process, allowing an individual to easily and without any problems obtain their resume in a formal style by utilizing this application builder tools. It is mostly focused on format, requiring only the selection of the preferred template and the provision of a few basic elements, which the application will quickly turn into the final resume.

The research also suggests a Two-Factor Authentication system that can work in tandem with an end-to-end encryption system based on the SHA256 algorithm. The proposed model and criteria are meant to act as a standard for comparing two-factor authentication systems that are currently in use and those that will be developed in the future, as well as for creating a novel system that meets the needs of practicability, simplicity, and strong security concepts while still being reasonably priced. The solution's chosen algorithm (SHA256) has undergone extensive testing and is widely recognized as the industry standard internationally. This encryption's crack ability has been discussed several times without success. The SHA256 cryptosystem, which provides extraordinarily high levels of security, enables this paradigm. It is also advisable to make use of AES for larger files as it takes lesser time when compare with 3DES [32].

One may use the secured Resume Builder Application to generate resume, whether as a seasoned professionals or young graduates. Users of the Resume Builder Application can build resumes in a common format. Any version of Windows may operate a resume builder application. It may be used to construct a resume by either experienced or brand-new users.

References

[1] Sailaja, K. & Usharani, M. (2015). Cloud computing security issues, challenges and its solutions in financial sectors. International Jour-nal of Advanced Scientific Technologies, Engineering and Man-agement Sciences, 3(1), 190-196.

[2] Taylor, C. (2017). Cloud Computing and Service Level Agreements (SLAs), http://www.datamation.com/cloud-computing/service-level-agreements.html

[3] https://jobportal.osunstate.gov.ng. Retrieved on the 30th of January, 2024

[4] Yao, G., Li, Y., Lei, L., Wang, H., & Lin, C. (2016). An efficient dynamic provable data possession scheme in cloud storage. In X. Huang, Y. Xiang, & K. C. Li (Eds.), Green, Pervasive, and Cloud Computing. Cham: Springer, pp. 63-81.

[5] J. Thakur and N. Kumar, "DES, AES and Blowfish: Symmetric key cryptography algorithms simulation based performance analysis," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 1, no. 2, pp. 6–12, 2011.

[6] Adedeji Kazeem B. & Ponnle Akinlolu.A, "A New Hybrid Data Encryption and Decryption Technique to Enhance Data Security in Communication Networks: Algorithm Development., *Int. J. Sci. Eng. Res.*, vol. 5, 2014.

[7] Mongoose https://www.npmjs.com/package/mongoose, https://github.com /Automattic /mongoose, https://mongoosejs.com/

[8] J. Kaur, S. Sharma, and M. Tech, "Secure image sharing on cloud using cryptographic algorithms: survey," *Int J Future Revolut Comput Sci Commun Eng*, vol. 4, no. 2, pp. 319–325, 2018.

[9] https://www.impervacom/learn/application-security/http-parameter-polution/ Retrieved on 12th February, 2024

[10] Pancholi, V. R., & Patel, B. P. (2016). Enhancement of cloud com-puting security with secure data storage using AES. International Journal for Innovative Research in Science and Technology, 2(9), 18-21.

[11] Sen, M., & Choudhury, S. S (2017). Security and privacy issues for cloud computing and its challenges. Review of Computer Engineer-ing Studies, 4(2), 62–66.

[12] Wagh, G., Guran, S., Vira, S, Dung, M., & Chaugule, A. (2017). Securing user's data on cloud using AES. International Journal on Recent and Innovation Trends in Computing and Communication, 5(2), 17–20.

[13] Pitchay, S. A., Alhiagem, W. A. A., Ridzuan, F., & Saudi, M. M. (2015). A proposed system concept on enhancing the encryption and decryption method for cloud

computing. Proceedings of the IEEE 17th UKSim-AMSS International Conference on Modelling and Simulation, pp. 201-205.

[14] Chauhan, A., & Gupta, J. (2016). Review on encrypt the text by MD5 and RSA in client cloud approach. International Journal of Advance Research, Ideas and Innovations in Technology, 2, 1-7.

[15] Jaglan, V. (2015). Proposing efficient approach to improve integrity checking in cloud data security.

[16] S. M. Seth and R. Mishra, "Comparative analysis of encryption algorithms for data communication 1," 2011.

[17] M. Agrawal and P. Mishra, "A comparative survey on symmetric key encryption techniques," *Int. J. Comput. Sci. Eng.*, vol. 4, no. 5, p. 877, 2012.

[18] M. A. Kumar and S. Karthikeyan, "Investigating the efficiency of Blowfish and Rejindael (AES) algorithms," *Int. J. Comput. Netw. Inf. Secur.*, vol. 4, no. 2, p. 22, 2012.

[19] A. A. Tamimi, *Performance Analysis of Data Encryption Algorithms. Retrieved October 1, 2008 Fromhttp*. 2008.

[20] N. Advani, C. Rathod, and A. M. Gonsai, "Comparative Study of Various Cryptographic Algorithms Used for Text, Image, and Video," in *Emerging Trends in Expert Applications and Security*, Springer, 2019, pp. 393–399.

[21] G. Singh, "A study of encryption algorithms (RSA, DES, 3DES and AES) for information security," *Int. J. Comput. Appl.*, vol. 67, no. 19, 2013.

[22] M. N. A. Wahid, A. Ali, B. Esparham, and M. Marwan, "A Comparison of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish for Guessing Attacks Prevention," *J. Comput. Sci. Appl. Inf. Technol.*, vol. 3, pp. 1–7, 2018.

[23] N. Aleisa, "A Comparison of the 3DES and AES Encryption Standards," *Int. J. Secur. Its Appl.*, vol. 9, no. 7, pp. 241–246, 2015.

[24] https://en.wikipedia.org/wiki/HTML, Retrieved; 20th January, 2024

[25] https://en.wikipedia.org/wiki/CSS, Retrieved; 28th December, 2023

[26] https://www.npmjs.com/package/bcrypt, Retrieved, 5th February, 2024

[27] https://www.npmjs.com/package/crypto-js, Retrieved, 3rd January, 2024

[28] https://en.wikipedia.org/wiki/JavaScript, retrieved, 2nd February, 2024

[29] https://jwt.io/, Retrieved, 5th February, 2024

[30] Pandey T. (2022). A secure data transmission over the cloud computing: using salted MD5. International Journal of Innovative Re-search in Computer and Communication Engineering, 4(2), 1-6.

[31] Nicholas, K., Wilson, C., Kibe, A. (2017). Enhancing trust in cloud computing using MD5 hashing algorithm and RSA encryption standard. International Journal of Scientific and Engineering Re-search, 8(3), 550-566.

[32] Gul, F, Amin, A, & Ahsraf, S. (2017). Enhancement of cloud computing security with secure data storage using AES. International Journal of Computer Science and Mobile Computing, 6(7), 27-32.