Final Year Project Report

**Full Unit – Final Report**

Access Control Application

Abdallah Hamdan, Oluwatosin Fasanmi, Saajan Maharjan

**Teacher:** Majeed Jaber

Computer Security

EPITA School of Engineering and Computer Science

May 15, 2023

Table of Contents

[Abstract 3](#_Toc134795679)

[Introduction 4](#_Toc134795680)

[Literature Review 5](#_Toc134795681)

[Methodology 6](#_Toc134795682)

[Results 7](#_Toc134795683)

[Conclusion 8](#_Toc134795684)

[Future Work 9](#_Toc134795685)

[References 10](#_Toc134795686)

Abstract

This project proposes a use of credential hash authentication to specific resources within a system. This project focuses only Authentication, and this project proposes a way to offer granular authentication to privileged users as opposed to a Group Policy and Security Group in an Active Directory IAM system.

# Introduction

In Enterprise networks, there have been multiple issues of Access Control misconfiguration, most cyber-attacks are usually targeted at users with privileged access. System Administrators are known to use simple passwords to ease the administration procedures carried out in their daily job functions. Threat actors exploit this to gain access, leading to massive data theft, financial loss, and reputational damage.

Access control is one of the fundamental components of security in any enterprise network. It is the practice of restricting access to resources based on the identity of the user or device seeking access. Access control helps prevent unauthorized access to sensitive data and systems, which can lead to data breaches, financial loss, and reputational damage. In recent years, there have been multiple instances of access control misconfiguration leading to massive data breaches, resulting in a significant impact on businesses and their customers. According to the 2021 Verizon Data Breach Investigations Report (Verizon, 2021), access control misconfiguration is the second most common cause of data breaches. Out of the 1,108 data breaches reported, 43% involved web application attacks, and 25% of those attacks involved access control misconfiguration. The report also states that 61% of all breaches involve stolen or weak passwords.

These statistics highlight the need for a robust access control system to prevent unauthorized access and protect sensitive data. In this project, we hash authentication, as an additional layer of security to access control. Hash authentication is a secure method of authentication that uses a one-way mathematical function to convert a password into an unreadable format, making it difficult for attackers to decipher. Further authentication can be implemented at lower levels and organizations can ensure that only authorized personnel have access to sensitive resources, mitigating the risk of data breaches due to access control misconfiguration.

# Literature Review

Hashing, according to the dictionary, is the act of "chopping something into small pieces" to create the appearance of a "confused mess". That definition accurately describes what hashing in computing is.

A hash function is a mathematical method used in cryptography that converts data of arbitrary size into a fixed-length bit string. The input function may also be referred to as message or just input. The hash or message digest is the name for the output of the fixed-size string function. According to OWASP (OWASP, 2018), the following crucial characteristics characterize hash functions used in cryptography:

* It's easy and practical to compute the hash, but "difficult or impossible to re-generate the original input if only the hash value is known."
* It's difficult to create an initial input that would match a specific desired output.

Thus, in contrast to encryption, hashing is a one-way mechanism. The data that is hashed cannot be practically "unhashed".

Different Hashing algorithms have been used to provide this one way scrambling, these include MD5, SHA256, Bcrypt. Hash implementations use salt to further randomize the input. This way, we protect against the flaw of the hash function by having a different hashed password each time. (Jung, 2021)

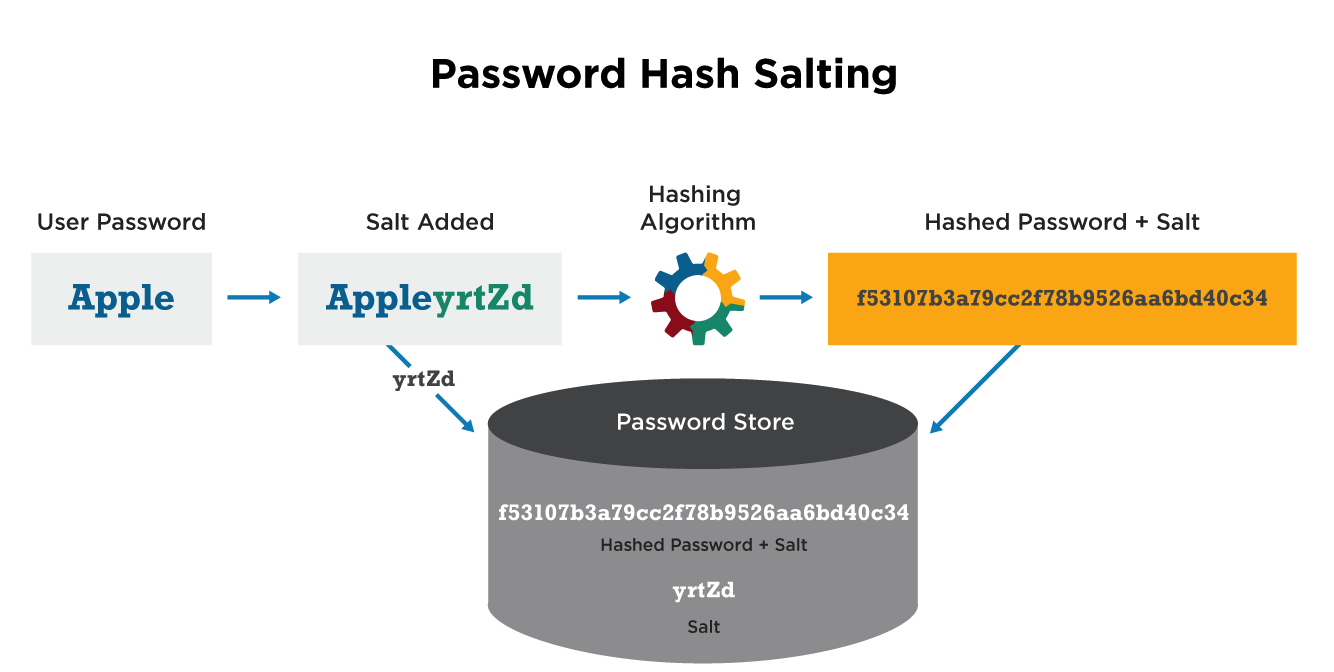


Figure 1

*Source: https://cyberhoot.com/cybrary/password-salting/*

# Methodology

For this project we use bcrypt. The bcrypt algorithm involves the following steps: The password is padded with null bytes until it reaches a length of 72 bytes. A random salt value is generated. The password and salt are combined and hashed 16 times using the Blowfish encryption algorithm. The resulting hash value is then encoded in a way that includes the salt value and a prefix indicating the number of iterations used in the hashing process. The use of a random salt value and multiple iterations of the hash function makes bcrypt resistant to attacks such as brute force and dictionary attacks.



Figure 2

A root account was pre-configured with the credentials stored in an XML file in a directory that the application can access. When the credentials are provided, the application generates a hash and compares the generated hash to the pre-set hash in our XML file, if the hash matches, then the root user is redirected to the second page to access the specified directories.

It is a security best practice not to use root account for routine procedures, hence we added a “create user” button, this new “admin” user can then be used to for routine administration. During additional account creation, the application prompts for the usernames and a password, computes the hash of the password and stores the username and hashed password in the XML file.

On successful authentication, the application redirects to the directory listing page, at the time of this report, the application only has read access in the specified directory.

# Results

The results that observed when implementing credential hash authentication:

* Increased security: Hashing makes it difficult for attackers to decipher passwords even if they gain access to the hashed values, reducing the risk of data breaches and unauthorized access.
* Improved access control: Credential hash authentication allows for granular authentication, enabling privileged users to access specific resources within a system. This provides better access control than a Group Policy and Security Group in an Active Directory IAM system, reducing the risk of access control misconfiguration.
* Resistant to brute force and dictionary attacks: Bcrypt, the hashing algorithm used in this project, is resistant to attacks such as brute force and dictionary attacks due to the use of a random salt value and multiple iterations of the hash function. This increases the security of the access control system.

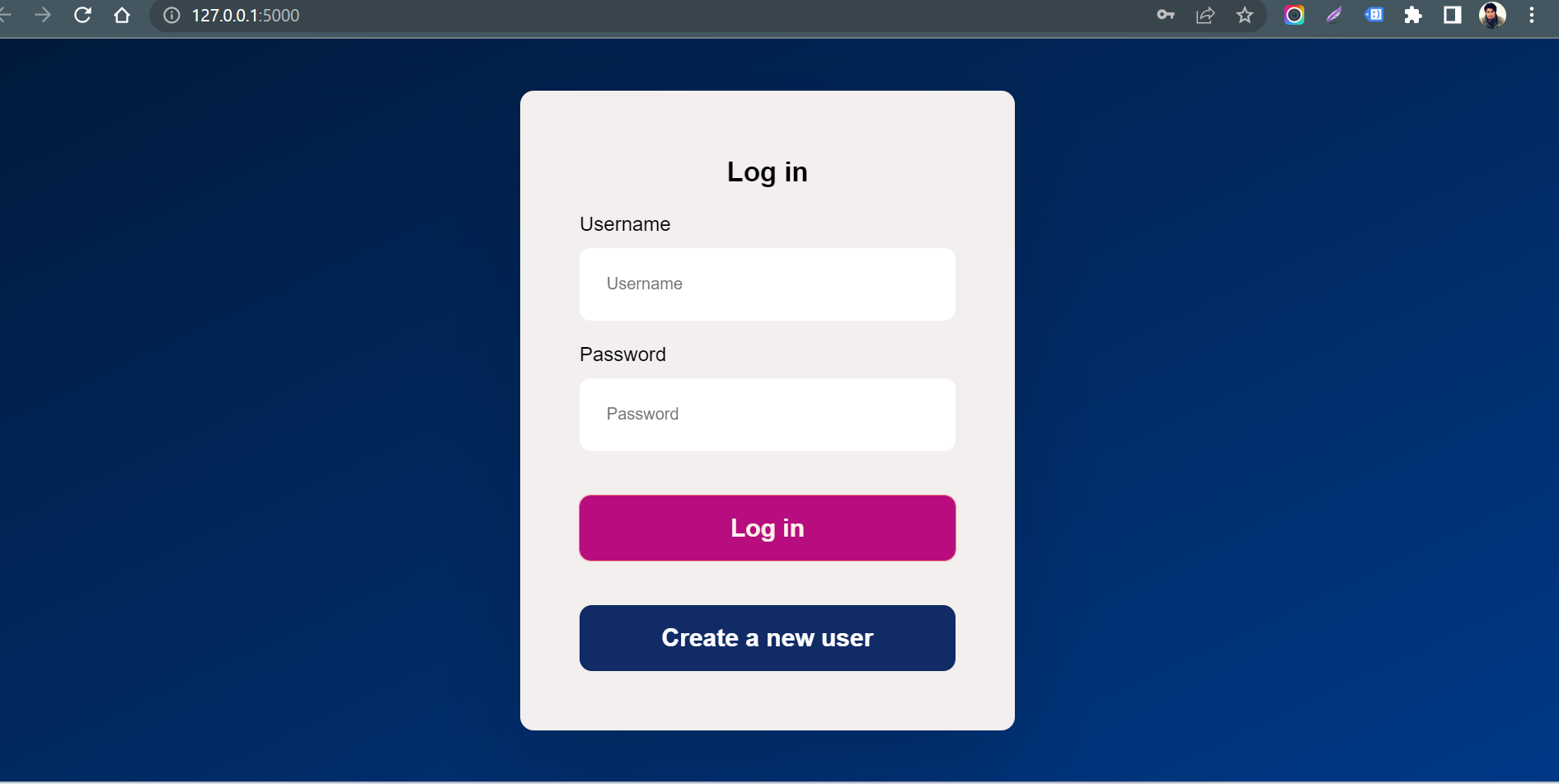


Figure 3: Login page

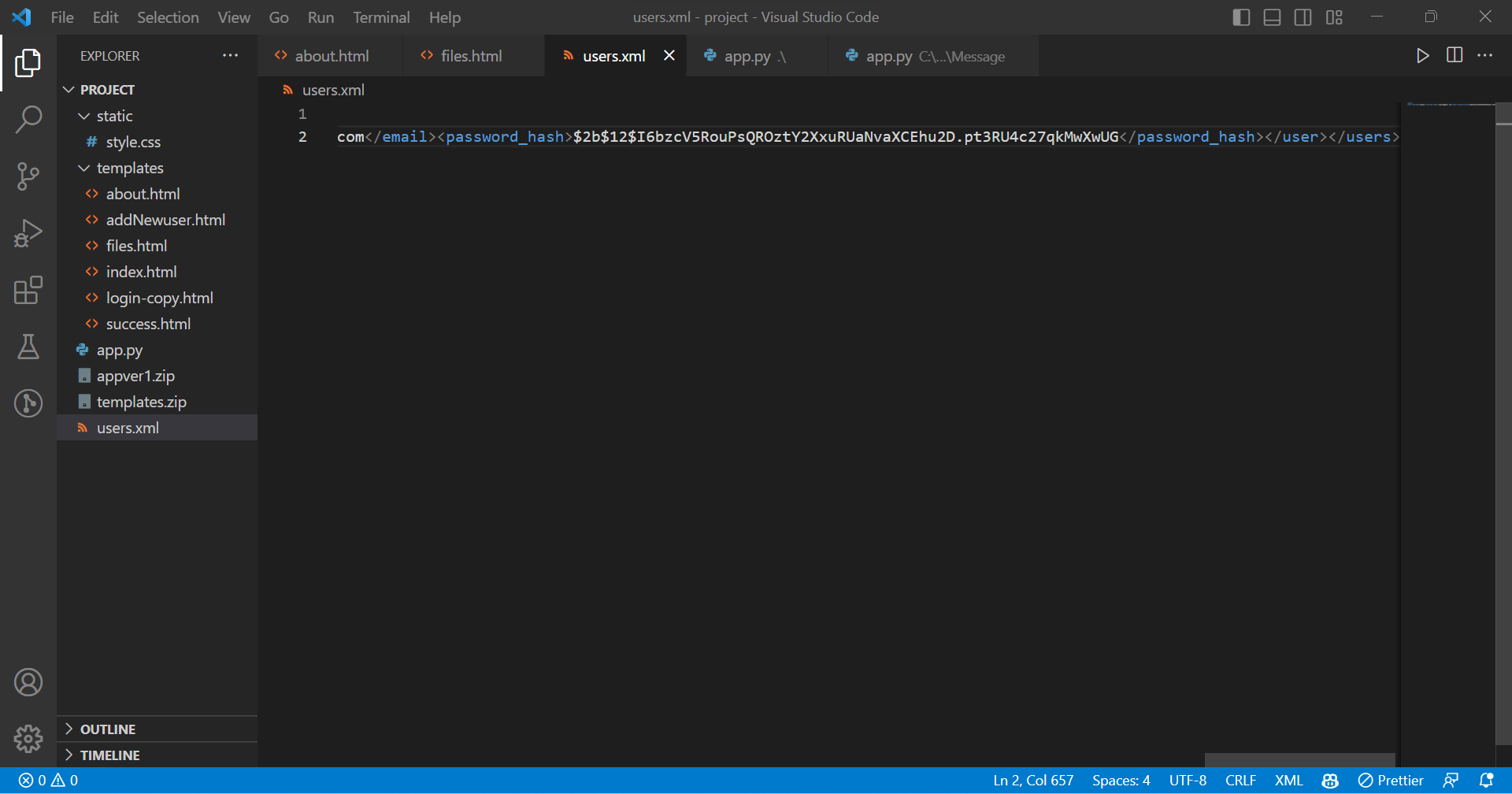


Figure 4: XML content

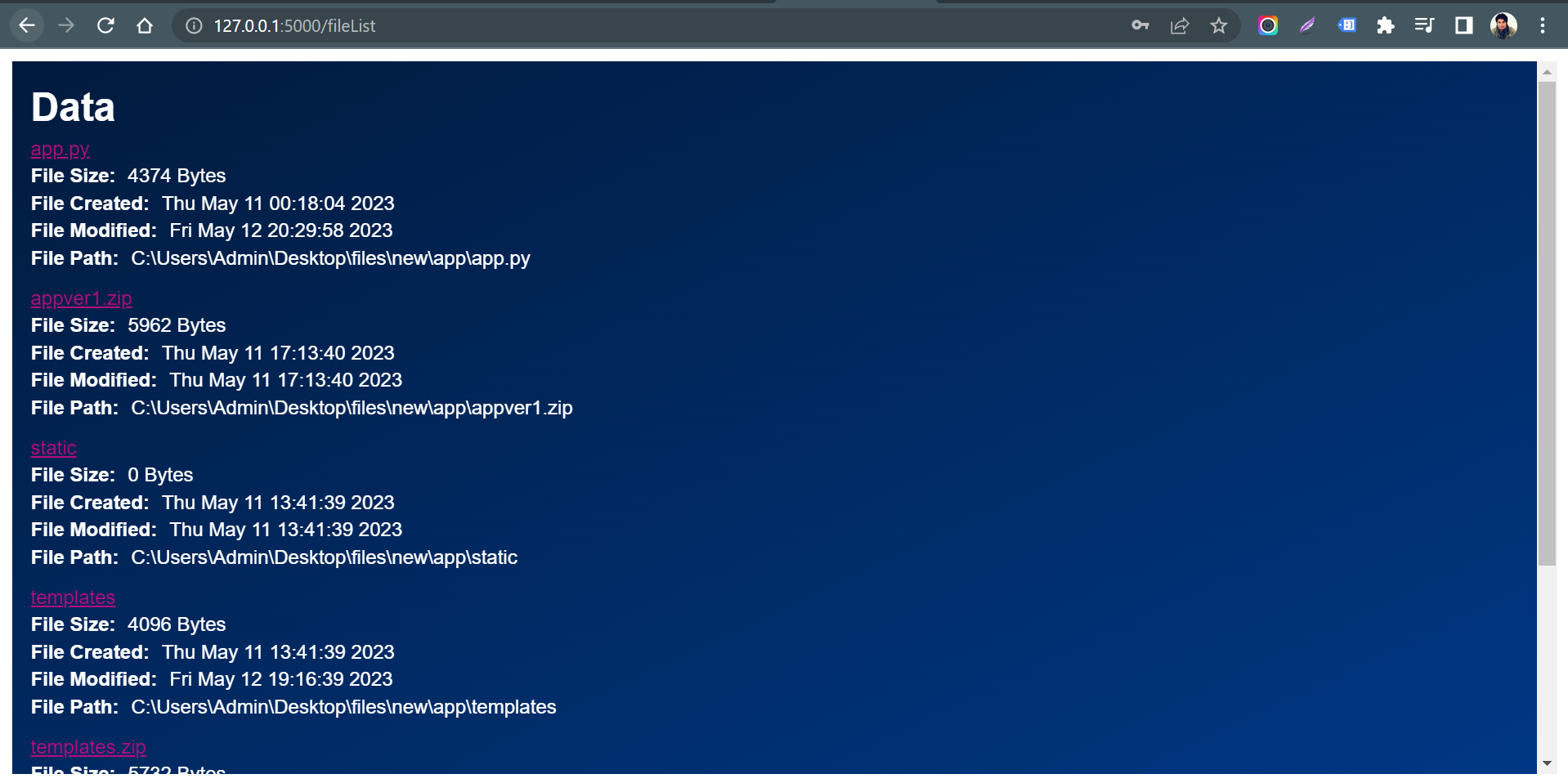


Figure 5: Directory List Page

# Conclusion

An application can be used to authenticate users based by hash comparison in systems, it provides an additional layer of security to access control, making it more difficult for attackers to gain unauthorized access.

Our application can be implemented independently of the underlying operating system or file system. This means that you can use it to control access to resources on any platform or device that supports web access, including mobile devices and cloud services.

# Future Work

The main problem this application tackles is the lack of a password authentication to access directories in. The application permissions can be extended from reading to writing on not only a local directory but also remote directory. It can also support MFA depending on requirements.

The use cases can also be extended to support a Windows Security Group which does not have encryption in the traditional sense but an access control list (ACL) References

# References

Jung, J. (2021, 05). *What are Salted Passwords and Password Hashing?* Retrieved from www.octa.com: https://www.okta.com/blog/2019/03/what-are-salted-passwords-and-password-hashing/

OWASP. (2018, June). *Guide to Cryptography*. Retrieved from https://wiki.owasp.org: https://wiki.owasp.org/index.php/Guide\_to\_Cryptography#Hashes

Verizon. (2021). *2021 DBIR Master's Guide*. Retrieved from Verizon: https://www.verizon.com/business/resources/reports/dbir/2021/masters-guide/