INFORMATION SECURITY MANAGEMENT SYSTEM
Semester project report for Network/Internet Security, Dr. Dipankar Dasgupta instructing
Department of Computer Science, University of Memphis, Memphis, TN

Team-10:
Sujith Medisetty (U00881475)
Abhishek Gudala
Bhanu Prakash Akula

Abstract:

The Information Security Management Project (ISMP) is a crucial approach that aims to protect confidential and sensitive data from security breaches. It uses various security controls and risk management techniques to prevent data leaks and other security threats. The ISMP focuses on safeguarding sensitive data using measures like data encryption, authentication, and authorization to ensure it remains confidential, secure, and accessible only to authorized personnel. By implementing these measures, the risk of unauthorized access and data breaches is significantly reduced, making the ISMP an essential part of any organization's overall security strategy.

Problem Statement:

The problem that the Information Security Management Project (ISMP) aims to address is the increasing risk of security breaches and data leaks that threaten the confidentiality and integrity of sensitive and confidential data. With the growth of technology and the increasing reliance on digital systems to store and transmit data, the risk of unauthorized access and data breaches has also increased. This poses a significant threat to businesses, organizations, and individuals that handle sensitive information, including personal information, financial records, and trade secrets. The ISMP addresses this problem by implementing various security controls and risk management techniques to prevent data leaks and other security threats, ensuring that sensitive data remains confidential, secure, and accessible only to authorized personnel.

Literature Survey:

Behler's guide on Spring Security provides an in-depth overview of how to implement authentication and authorization in a web application using Spring Security [1]. The guide covers topics such as user management, role-based access control, and how to configure and customize Spring Security for a variety of use cases. Behler's guide also provides practical examples and step-by-step instructions for implementing Spring Security in a web application.

[1] Behler, M. (2022, May 30). Spring security: Authentication and authorization in-depth. Learn more about Java, no matter your skill level, anytime and anywhere you want. Retrieved May 2, 2023, from https://www.marcobehler.com/guides/spring-security

Project Functionalities:

The Information Security Management System (ISMS) has two dashboards that serve different purposes.

- **Admin dashboard**
- **User dashboard**

1. Admin Dashboard: Provides tools to manage and monitor the system.

The dashboard of the Information Security Management System (ISMS) provides various functions to enhance data security, such as:

- Control-Over-User-Profiles: Direct registration of users to the ISMS is not allowed. Instead, users must submit a request to activate their account, which the admin can approve or deny. The admin can also delete authorized user accounts to ensure only authorized personnel have access to the ISMS.
- Control over Resources: The admin has the authority to upload or delete any number of files and add relevant information while uploading. This helps to manage resources effectively.
- Announcement Management: The admin can create announcements for all or specific groups of users, ensuring privacy by giving the admin

control over who can access the files. A history of all announcements can be viewed by the admin.

- System health monitoring: The admin can monitor the system performance, logs, and other metrics to identify and fix any problems with the ISMS as soon as possible.
- Control over shared resources: The admin can control who has access to shared resources and can grant or revoke permission to download files, ensuring only authorized personnel have access.

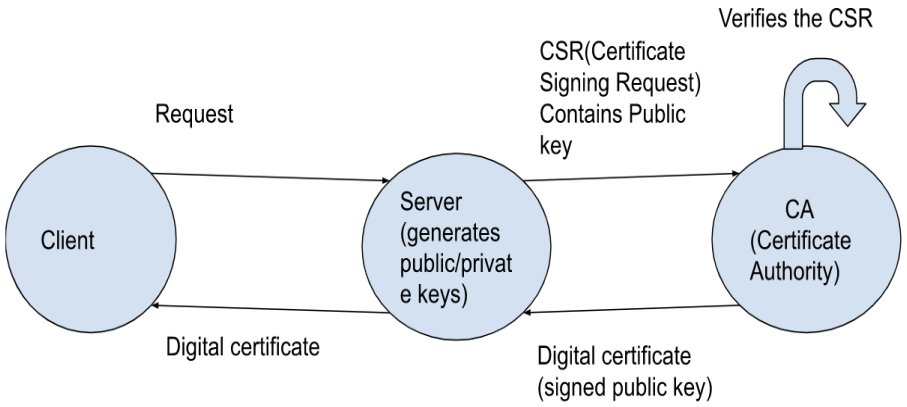
2. User Dashboard:

The user dashboard is an easy-to-use interface that provides users with access to their files and enables them to interact with the ISMS. Here are some of the key features of the user dashboard:

- File Management: Users can download files from the ISMS that have been uploaded by the admin, but only after their request to download the file has been approved by the admin. This helps to ensure that only authorized users have access to sensitive information.
- Profile Management: The user dashboard allows users to view and edit their profile information as necessary.
- Announcements: The user dashboard displays all the announcements that are made by the admin to the users, ensuring that all users are informed about important updates or changes to the system.
- People Management: The user dashboard includes a list of all the users who are authorized to use the ISMS, making it easy for users to find colleagues as needed.

Security Features:

1. **Endpoints running on HTTPS (S1) :**



The above illustrates the Real-time implementation of HTTPS (How HTTPS works): Here's how the process works:

- The server generates a public/private key pair.
- The server creates a CSR that includes the public key, along with information about the server's identity (such as the domain name).
- The CSR is sent to a trusted Certificate Authority (CA) for verification.
- The CA verifies the identity of the server and signs the server's public key to create a digital certificate.
- The digital certificate is sent back to the server.
- When a client connects to the server over HTTPS, the server sends the digital certificate to the client as part of the initial response. The certificate includes the server's public key, which is used by the client to establish a secure connection with the server.

But in our application, We have self-signed the digital certificate as we are running our project on localhost, But the certificate is trusted by the system.

Steps that we have followed for self-signing the digital certificate:

- To secure our application, we generated a public/private key pair using OpenSSL and key-tool.
- We then created a .cnf file with all the configuration details to generate a self-signed certificate.
- Next, we trusted our self-signed certificate and our application is now

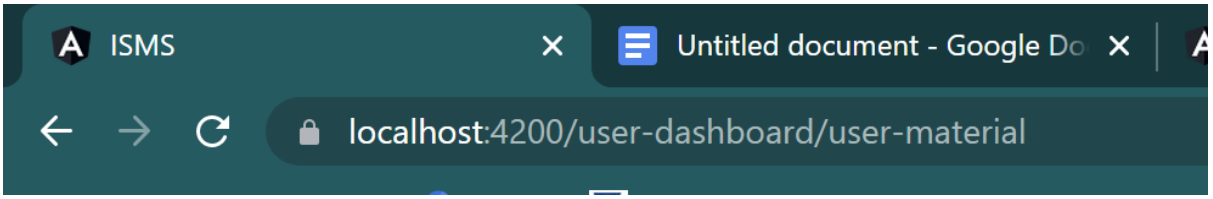
running over HTTPS.

- When the digital certificate presented by Angular is valid and trusted, the Spring App will obtain the public key from the digital certificate.
- Spring will then use the public key to encrypt a session key (randomly generated), which is then sent to Angular.
- Finally, the session key is used to encrypt and decrypt data transmitted between the client and server over the HTTPS connection.

Since we are running our application on localhost, we have self-signed our application. However, once we deploy our application, we can make it a proper HTTPS with CA authorized digital certificate.

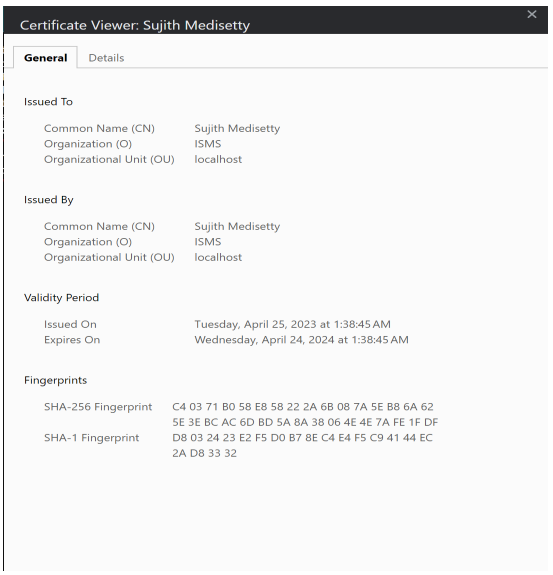
Without trusting the digital certificate:

After trusting the digital certificate



The endpoints are now secured..!

Self Signed:

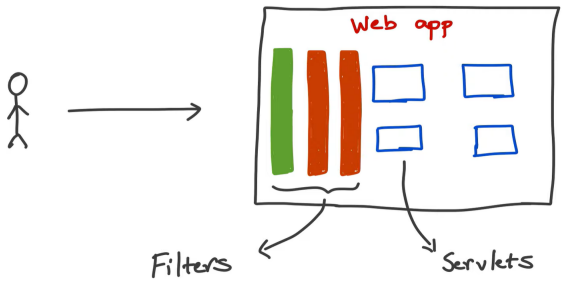


2. Spring Security Filters (S2) : (For authentication, authorization, cors)

What is a filter?

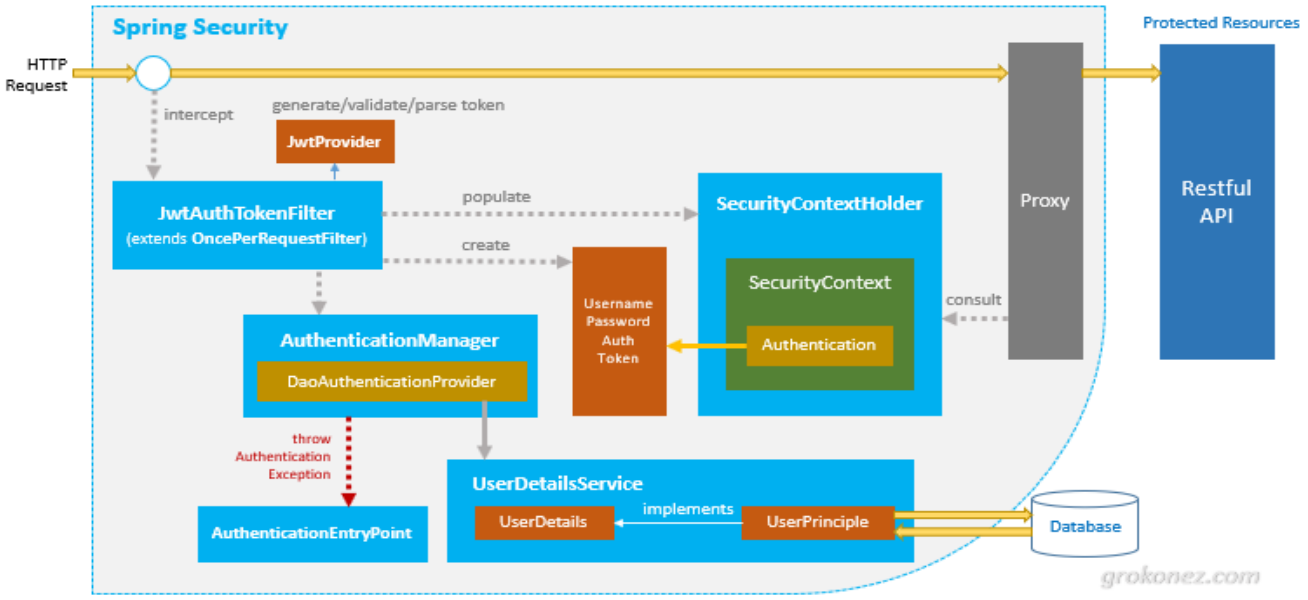
Spring Security filters intercept incoming HTTP requests and perform a variety of security-related tasks. These tasks can include:

- Authentication
- Authorization
- Cors handling



Filters used in this application:

1. **JwtAuthenticationFilter - for Authentication**



The JwtAuthenticationFilter is a crucial component in the Spring Security framework that handles authentication using JSON Web Tokens (JWTs). This filter intercepts incoming HTTP requests to the web application and checks for the presence of a JWT.

If a JWT is found, the filter proceeds to verify its signature to ensure that it has not been tampered with during transit. Once the signature is verified, the user's identity is extracted from the claims embedded within the token.

If the JWT is determined to be valid and the user's identity is successfully extracted, the filter passes the control flow to the UsernamePasswordAuthenticationFilter to handle additional authentication tasks.

This process helps to provide a robust and secure authentication mechanism for the web application, thereby enhancing its overall security posture.

The code in the project provided is the same as above described architecture. The picture illustrates the JWT validation and the application also does the validation in the same way. HTTP requests will pass through the JWTAuth filter and if the token is present then it will validate the signature and extract the user identity from claims and validate with the data present in the database with the

help of the UserDetailsService interface if the user details are valid then the filter will check the validity like expiry of the token ..etc and if everything looks fine then the request is sent to UserPasswordAuthenticationFilter and later on saves the current user details in the SpringSecurityContext.

2. corsFilter - To allow the specific origin requests

CORS, which stands for Cross-Origin Resource Sharing, is a security feature implemented by web browsers that limit web pages from making requests to a different domain than the one that initially served the web page.

In our web application, the Spring web server runs on port 8080 while Angular runs on port 4200. To enable requests from port 4200, we utilize a CORS filter which overrides the browser's default restrictions and allows the web pages to access the resources they require from the server. This helps to ensure that the web application runs smoothly and securely, providing users with a seamless browsing experience.

3. UsernamePasswordAuthenticationFilter

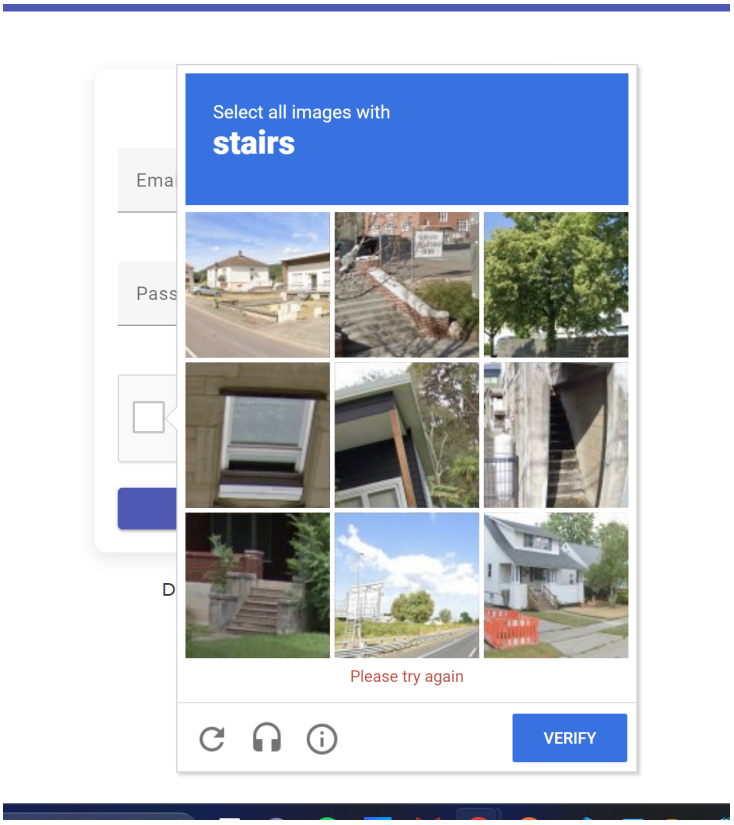
After the JwtAuthenticationFilter has processed the request, the UsernamePasswordAuthenticationFilter takes over. This filter's primary function is to set the user's authentication object in the request's security context if the credentials are valid. Once this is done, the web application can proceed to grant the user access to protected resources.

4. Authorization Filter

The AuthorizationFilter is a crucial component of Spring Security used for web application authorization. This filter examines HTTP requests and verifies whether the user has the necessary permissions to access the requested resource. The filter can be used together with other authentication filters, such as JwtAuthenticationFilter and UsernamePasswordAuthenticationFilter, to enhance the security of the application. In our case, we have configured the AuthorizationFilter to use two roles - ROLE_ADMIN and ROLE_USER - for authorization purposes.

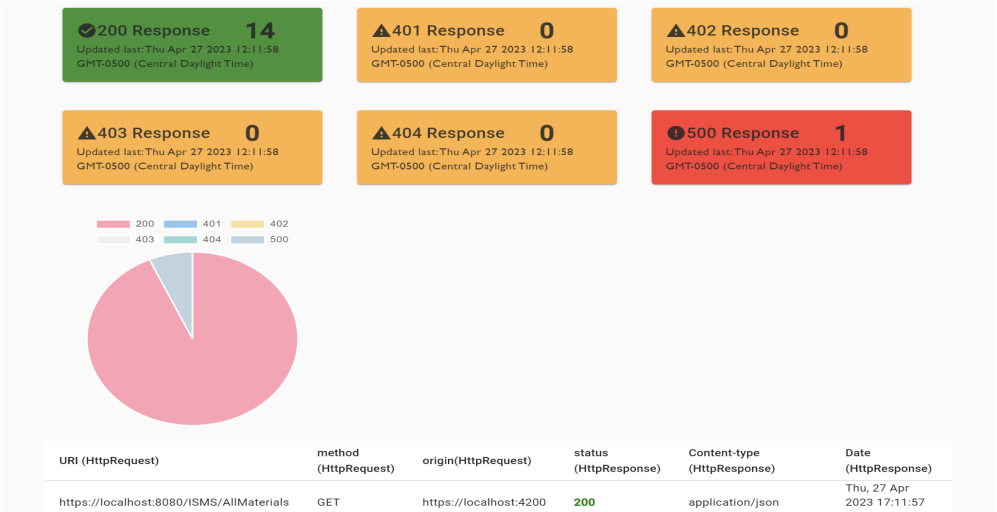
3. CAPTCHA (S3)

CAPTCHA can help prevent DoS attacks by adding an extra layer of protection The below picture is from the project at the time of login.



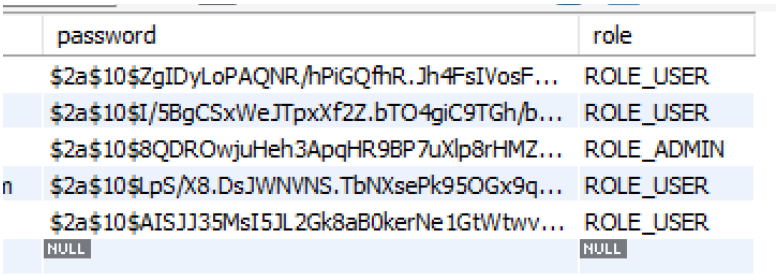
4. Health Dashboard (S4):

Health Dashboard provides detailed information about the HTTP requests and responses made by the application, including status codes and backend server information. Additionally, graphs are used to enhance the user experience. Monitoring and tracking the application is made easier with this feature. Below is a screenshot of the health monitoring dashboard:



5. Encryption of resources (S5):

To ensure the security of sensitive resources, we have stored them in an encrypted format in a secure database using the AES algorithm. Furthermore, all passwords and encryption keys have been encoded using the PasswordEncoder class in Java, which ensures that they can only be encoded and not decoded. This ensures that the passwords and encryption keys are securely stored and cannot be easily accessed by unauthorized entities.



	password	role
	$2a$10$ZgIDyLoPAQNR/hPiGQfhR.Jh4FsIVosF...	ROLE_USER
	$2a$10$I/5BgCSxWeJTpxXf2Z.bTO4giC9TGh/b...	ROLE_USER
	$2a$10$8QDROwjuHeh3ApqHR9BP7uXlp8rHMZ...	ROLE_ADMIN
n	$2a$10$LpS/X8.DsJWNVNS.TbNXsePk95OGx9q...	ROLE_USER
	$2a$10$AISJJ35MsI5JL2Gk8aB0kerNe1GtWtwv...	ROLE_USER
	NULL	NULL

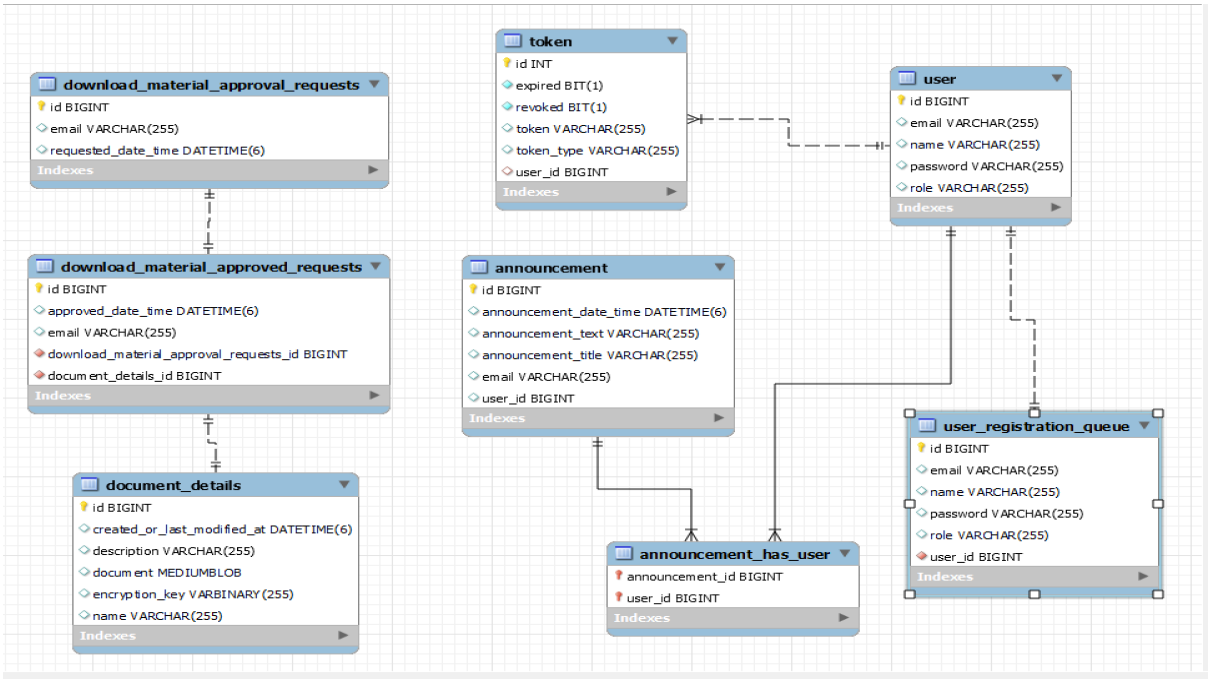
6. Control Over Profiles (S6)

Users can't create an account on the ISMS directly. Instead, they need to submit a request to activate their account. An admin will then review and either approve or reject the request. If approved, the user can access the system. If an admin wants to revoke access, they can delete the user's account. This ensures that only authorized users can access the ISMS.

7. File Access (S7)

Only authorized users can download files that have been uploaded to the system by an admin. To download a file, a user needs to first request access to it. An admin will then review the request and approve or reject it. If approved, the user can download the file. This ensures that sensitive information remains protected and that only authorized users can access it.

ER-Diagram of the Project:



Tables:

User table - stores the data of all approved users
User_registration_queue - stores the data of the registration requests raised by users.
Token table - Stores the JWT tokens of the users in an encoded fashion
Announcement table - stores the details of announcements made by the admin
Download_material_approval_requests table - all the download material requests raised by users to admin are stored in this table.
Download_material_approved_requests table - all the approved download request details are stored in this table
Document_details table - all the documents with their details are stored in this table in an encrypted format.

Relationships between the tables:

- The one-to-one relationship between the user table and the user_registration_queue table
- The One-to-one relationship between Download_material_approval_requests and Download_material_approved_requests tables
- The One-to-one relationship between Download_material_approved_requests and Document_details tables
- One-to-many relationship between the user table and the tokens table
- The many-to-many relationship between the user and announcements table.

Experiments and results:

We used Postman, a tool to test our application's endpoints and ensure they are working correctly. This allowed us to validate the authentication and authorization features of the application, which restrict endpoint access based on user roles. Therefore, only authorized individuals can access specific endpoints.
Coming to automation tools we have a bit of selenium to test the behavior of our application. (especially backend application)

All of the endpoints in the application are secure and functioning properly. The various security features, such as authentication, authorization, captcha, encryption, and control over user profiles and file access, are all working as expected. Additionally, the application's front end was developed using Angular, which is a modern front-end development framework.

The Postman experimental results will be uploaded along with this document.

Conclusion:

In conclusion, the Information Security Management Project (ISMP) is a crucial approach to protecting sensitive and confidential data from security breaches. The ISMP uses various security controls and risk management techniques to prevent data leaks and other security threats, ensuring that sensitive data remains confidential, secure, and accessible only to authorized personnel.

The application implements various security features, including endpoint security using HTTPS, Spring Security Filters for authentication and

authorization, CAPTCHA for preventing DoS attacks, encryption and encoding of resources, control over user profiles and file access by the admin, and a health dashboard for monitoring and tracking the application's performance.

Furthermore, the application is developed with the modern frontend framework Angular, resulting in a fluid and user-friendly interface. Overall, the application's security features and modern frontend framework make it a reliable and secure tool for managing sensitive and confidential data.

References:

- Authorization architecture. Authorization Architecture:: Spring Security. (n.d.). Retrieved May 2, 2023, from https://docs.spring.io/spring-security/reference/servlet/authorization/architecture.html
- *Spring Security - Reference Documentation*. (n.d.). Retrieved May 2, 2023, from https://docs.spring.io/spring-security/site/docs/3.0.x/reference/springsecurity.pdf
- *Get started with angular, An Advanced Frontend Framework*. Angular. (n.d.). Retrieved May 2, 2023, from https://angular.io/guide/component-overview
- Agnihotri, A. (n.d.). *Angular HTTP (web API integration)*. C# Corner. Retrieved May 2, 2023, from https://www.c-sharpcorner.com/article/angular-http-web-api-integration/
- Wagde, W. by: S. (2022, November 17). *Using JWT with Spring Security Oauth*. Baeldung. Retrieved May 2, 2023, from https://www.baeldung.com/spring-security-oauth-jwt