Secure Financemanagement: Enhancing JWT-supported authentication through dynamic access control system and multi-factor authentication

1st Christopher Unkart
Frankfurt University of Applied Sciences
Frankfurt, Germany
unkart@stud.fra-uas.de

2nd Marc Grunwald
Frankfurt University of Applied Sciences
Frankfurt, Germany
marc.grunwald@stud.fra-uas.de

***Abstract*—In the evolving landscape of secure finance management, robust authentication mechanisms are essential to protect sensitive financial data. This paper explores an enhanced authentication framework that integrates JSON Web Token (JWT)-supported authentication with a risk assessment component and multifactor authentication (MFA). The proposed solution addresses critical security challenges by implementing context-aware authentication policies that adapt to user behavior and environmental conditions, thus reducing unauthorized access risks. Furthermore, the incorporation of MFA strengthens the authentication process by requiring different forms of verification when necessary, significantly reducing vulnerability to credential compromise. For determining when MFA is necessary a risk score is calculated based on different factors and dependent on the resulting risk score a different layer of authentication is triggered. The proposed approach was implemented using the Spring Framework, Kotlin, JWT and PostgreSQL database.**

***Index Terms*—access control, security, authentication, jwt, mfa, finance management**

I. Introduction

The ongoing digitization of financial services has transformed the way individuals and organizations manage and transact financial resources. However, this evolution has also introduced critical security vulnerabilities, particularly in protecting sensitive financial data from unauthorized access and cyber threats, thus violating the integrity, confidentiality, and availability of the data [1]. Research indicates that the financial sector is among the most targeted sectors for cyberattacks, with incidents such as data breaches, phishing, and credential theft threatening their stability and trustworthiness [2] [3]. Traditional authentication mechanisms, particularly single-factor password-based systems, have been identified as a significant weakness, especially when users decide to use a weak password or reuse the same password across multiple systems. Additionally users can run into the issue of their password being stolen through, phishing, social engineering, man-in-the-middle, and keylogging attacks [4], making enhanced authentication systems an interesting topic. This paper will propose an enhanced system for financial transactions. The system will use JWTs and enhances the authentication process by analyzing the context of the login and behavior of the user. The result of those analyzes will be the risk score, which decides if a login is suspicious or not. In case of a suspicious login, the proposed system will be enhanced, by using Multi-factor Authentication (MFA).

A. Identification and Authentication

Users, systems and devices can provide information to identify themselves. This information can e.g be an ID or a name, but does not mean that a user truly is who he says he is. At this point Authentication becomes important, as a proof of identification through secret information [5]. Authentication methods can be categorized into three types. Those types are authentication through knowledge, authentication through possession and authentication through biometrics. Typical examples for those authentication types are passwords and Pins for knowledge-based authentication, smartcard and tokens for posession-based authentication and fingerprints and iris scan for biometric-based authentication [5].

B. Single- and Multi-Factor Authentication

In the beginning, the authentication process only took one factor into account when authenticating the subject [6]. This was known as Single-Factor Authentication and was widely considered because of its simplicity and ease of use. For example, the use of a password (or PIN) to verify a user's identity was common, despite being the weakest form of authentication [6]. Sharing a password could immediately compromise an account and unauthorized users might exploit vulnerabilities through i.e., dictionary attacks, rainbow tables, or social engineering. Consequently, systems typically enforced minimum password complexity requirements [6].
It became evident, that relying on a single-factor for authentication was insufficient in protecting from some security threats. In response to this, Two-Factor Authentication (2FA) was introduced, combining standard credentials with an additional element of personal ownership, like i.e. a smart card or mobile device [6].

Multi-Factor Authentication (MFA) was introduced to achieve a higher level of security and to ensure the protection of computing devices and other critical services from unauthorized access by employing more authentication factors. This leads to enhanced security, because users are required to provide evidence of their identity through two or more distinct factors [6]. Those factors can be i.e., a password, a token, the voice, facial features and fingerprints, but behavioral factors are also a possibility [6].

C. Authorization

Authorization defines what an identified and authenticated User, system or device is actually allowed to do. This includes which resources are accessible for the authenticated entity and which operations the entity can perform. In order to achieve this, predefined criteria have to be known, which allow granting or not granting access [5].

D. Risk-Based Authentication

Risk-based Authentication (RBA) is typically used alongside passwords or other user authentication methods to protect against sophisticated attackers who either know the correct credentials or can guess them with a low number of attempts [7]. Possible attacks can include credential stuffing attacks, where attackers use credentials from other compromised services, phishing attacks, or online guessing attacks [7]. During the authentication process, e.g typing in the username and password, additional features can be gathered which then can be used in deriving a risk score, based on those features. Features that can be considered when calculating the risk score can be e.g the IP address and the user agent. The impact of the features on the risk assessment can differ, because the difficulty of manipulating the considered features is different. For instance spoofing the IP address is considered more difficult than spoofing the user agent [7].

After calculating the risk score, a variety of actions can be performed, dependent on how high the risk is. If the risk is considered to be medium, additional authentication factors such as the verification of email address or phone number through a CAPTCHA could be used. If the risk is considered to be high, the access could be blocked, although this event is rare, because false classification of legitimate users as high risk can be problematic [7].

E. Density-Based Spatial Clustering of Applications with Noise

A clustering algorithm that is often used in detecting anomalies in data is the Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The algorithm forms high-density regions in the data into clusters, that are separated by lower object density regions [8]. The algorithm requires two parameters, the neighborhood distance ϵ (eps) and the minimum number of points that are required to form a high-density region (minpts) [8]. The data is categorized through the parameters into core points, border points and outlier points. A core point forms, as the name implies, the core of the cluster, which has more than minpts number of points within the ϵ (eps) distance [8]. In comparison to that, a border point has fewer than minpts number of points within eps but is still in the neighborhood of a core point [8]. Outlier points do not fit any cluster and are anomalous points. They are neither a core point nor a border point [8].

The algorithm defines density as follows:

1. ϵ-neighborhood are the objects that are within a radius ϵ from an object and can be represented by the following equation [8]:

$$N\epsilon(p) : \{q|d(p,q) \leq \epsilon$$

In the equation, p and q are data points in the space and $d(p,q)$ describes the distance between p and q [8].

2. High density is defined as an object that contains at least minpts data points in its ϵ-neighborhood [8]

F. JSON Web Token

JSON Web Token (JWT) is widely used as a standard for authentication and authorization and is based on RFC 7519 [9] [10]. JWTs are used for transmitting claims between two parties in a secure manner. A JWT is structured into three parts, the header, the payload and the signature. The header specifies the token type and signing algorithm. The payload contains the claims or information to be shared, and the signature part ensures the integrity and authenticity of the token [11]. JWTs are often used in authentication systems to represent a user's verified identity.

II. Related Work

Bucko et al. [10] address the limitations of traditional JSON Web Tokens (JWTs) by integrating user behavior analytics to enhance security in web applications. Their work proposes a method for enhancing the authentication trustworthiness by taking multiple factors into consideration. The factors they take into consideration are the number of password attempts, IP address consistency, and the user agent type. For each factor they assign a specific weight, and the cumulative score is stored and updated in the user's account after each transaction.

Haekal et al. [12] developed the SIKASIR application, which was designed to assist small business owners in managing their operations across various platforms. A critical component of this application is its authentication mechanism. The study explores the implementation of token-based authentication using JWTs within the SIKASIR RESTful web service. By adopting JWTs, the application ensures secure, stateless authentication, enhancing the overall security of the system.

Bakar and Haron [13] introduce the Unified Authentication Platform (UAP), an adaptive authentication system designed to identify suspicious login attempts by analyzing users past login histories. The system can detect anomalies and respond

accordingly, by establishing a baseline of normal user behavior. They focus on behavioral aspects by analyzing the context of a login e.g., time and location to develop a normal behavior profile for a user. The system compares the current login behavior to the normal behavior profile of the user to classify it as low risk or high risk.
Lora et al. [14] are following an approach that uses multifactor security algorithms to enhance security. Çelik [15] et al. are using the DBSCAN algorithm to detect anomalies in temperature data. They compare this approach to a statistical anomaly detection method and conclude that DBSCAN has several advantages over the statistical approach on discovering anomalies.

III. Methodology

For the development of the Finance management application, the Spring Framework was used. Kotlin was used as the programming language. REST technology and PostgreSQL as database engine were used to implement the API.
The application was kept simple for demonstration purposes and allows users to transfer money between each other or change the settings. When the application is called, a sign in page is shown as in Fig. 1. Before the application can be used, the user has to authenticate himself at the sign in page or register himself, if he does not have an account yet. For the sign in process the **"api/auth/signin"** endpoint is used. The user has to enter his credentials, that consists out of his E-mail address and his password. The entered password is hashed by using Bcrypt as algorithm and is compared to the hashed value that is stored in the database. After verifying that the user entered the correct credentials, a risk score is calculated based on different factors. Dependent on the resulting risk score, the system decides if the Single Factor Authentication is sufficient or if Multi Factor Authentication (MFA) is necessary. The system implements multiple MFA methods, which are explained in more detail in III-B .When the user signs in successfully, a JSON Web Token (JWT) is generated and send to the client as a response as seen in Fig. 2 and stored in the local storage of the used browser. This JWT is used for authentication in subsequent requests by being added to the authorization header of the request as shown in Fig. 3. After a user is signed in successfully and receiving his JWT he is redirected to the dashboard as seen in Fig. 4. At the dashboard the user can change his settings or can manage money transfers between him and other users by specifying the recipient of the transfer and the amount of money that should be transferred.
As mentioned before, a risk score is calculated for a login attempt, before a user is getting his JWT. This risk score takes multiple factors into consideration. Those factors are the IP address, the failed login attempts in the last 30 minutes, the usual login times of the user and the browser consistency. The details of how the risk score is being calculated are explained in III-A

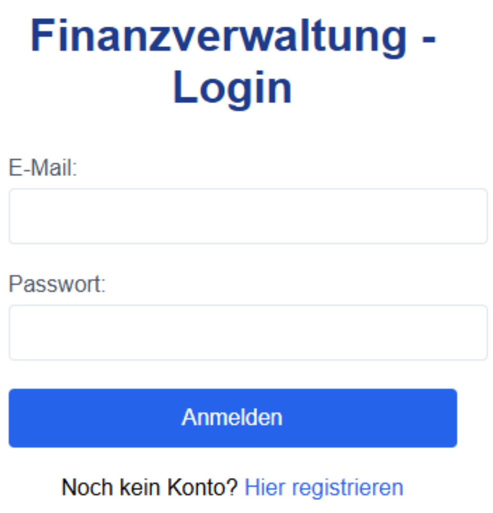
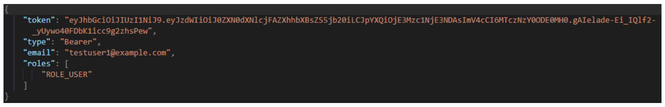
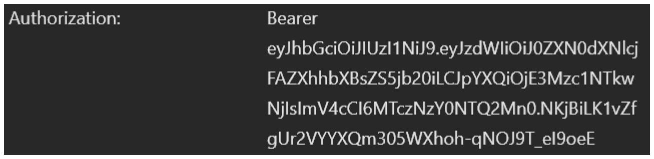


Fig. 1. Signin page of the finance management application



```
{
    "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ0ZXN0dXNlcjFAZXhhbXBsZS5jb20iLCJpYXQiOjE3Mzc1NjE3NDAsImV4cCI6MTczNzY0ODE0MH0.gAIelade-Ei_IQlf2-
      _yUywo40FDbK1icc9g2zhsPew",
    "type": "Bearer",
    "email": "testuser1@example.com",
    "roles": [
        "ROLE_USER"
    ]
}
```

Fig. 2. JWT response after successful Sign in



```
Authorization:        Bearer
                      eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ0ZXN0dXNlcj
                      FAZXhhbXBsZS5jb20iLCJpYXQiOjE3Mzc1NTkw
                      NjIsImV4cCI6MTczNzY0NTQ2Mn0.NKjBiLK1vZf
                      gUr2VYYXQm305WXhoh-qNOJ9T_el9oeE
```

Fig. 3. Authorization header of a transfer request that contains the JWT as bearer token

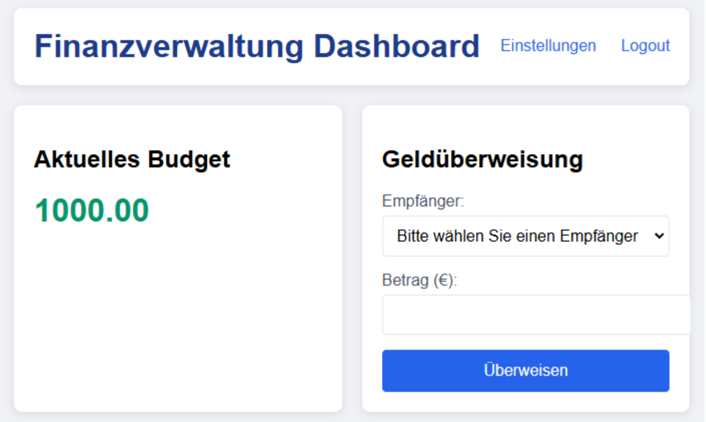


Fig. 4. Redirect to the dashboard page of the finance management application after successful sign in

A. Risk Score Factors and Calculation

For calculating the risk score, data has to be gathered. This data is gathered by implementing listeners for successful and unsuccessful login events. When a user tries to sign in, the data associated with this login attempt is stored in the database, and a status is added that shows whether the login event was successful or not. When calculating the risk score, the data that is stored in the login event repository is used. The stored data includes the IP address, the failed login attempts in the last 30 minutes, the usual login times of the user and the browser consistency and were, except for the login time, also considered by Bucko et. al [10] in their approach. The IP address data is extracted from the request sent to the server. The failed login attempts are calculated by taking the time of the current login and selecting the login events that happened between the current time and 30 minutes before the current login time. After that the login events in the database with the status unsuccessful are counted to get the number of failed login attempts in the last 30 minutes. The browser consistency data is extracted from the user agent header of the request, which contains the browser, the browser version and the operating system. For the creation of the usual time windows and the detection of anomalies in those windows, the DBSCAN algorithm is used. For the DBSCAN algorithm we are taking the data of the login times of the user that are stored as LocalDateTime in our database and turn this data into the day and time. The day is represented by a number between one and seven, with one indicating the day being Monday and seven indicating the day being Sunday. The time is turned into a decimal value between 0 and 24. The data for the day and for the time is then being normalized through the following min-max feature scaling equation [16]:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

With x_{min} being 1.0 and x_{max} being 7.0 for the day. And x_{min} being 0.0 and x_{max} being 24.0 for the time. The rescaling is mainly done to prevent the time feature from dominating the day feature. For the parameters minpts and ϵ the values 3 for minpts and 0.1 for ϵ are chosen. Because the data was normalized a ϵ value between 0 and 1 made sense and 0.1 showed good results for the clustering. The login event database grows for a user the longer the application is used. The application did not run for a long time and the database was wiped regularly, which led to the database being rather small so 3 was chosen for the minpts parameter, meaning that a cluster is formed when at least 3 points were in its ϵ-neighborhood. This parameter might need adjustment when the databases size increases. More details will be discussed in section IV.

The risk score calculation based on the previously mentioned factors follows a similar approach as [10] Bucko et. al proposed, but instead of calculating a score in terms of the trustworthiness a percentage based score is calculated in terms of the risk of a login. The following shows how the percentage based risk score is calculated based on the previously mentioned factors:

- IP of login:
  - The ip address is knwon from a successful login: 0%
  - The ip address is unknown: 20%
- failed login attempts
  - 0 retries: 0%
  - 1 retry: 10%
  - 2 retries: 20%
  - 3 and more retries: 40%
- usual login times
  - Login time is within usual login times: 0%
  - Login time is outside usual login times: 25%
- browser consistency
  - Combination of Browser, Browser Version and operating system are known from previous successful login: 0%
  - Combination of Browser, Browser Version and operating system are unknown: 15%

Dependent on the resulting risk score, an additional factor might be neccessary for authentication, which is explained in more detail in III-B.

The percentage-base calculation is represented in the following Equation:

$$\begin{aligned} &riskscore = \\ &\quad \{[(knownip \rightarrow 0\%)||(unknownip \rightarrow 20\%)]_{ip} + \\ &\quad [(0 \rightarrow 0\%)||(1 \rightarrow 10\%)||(2 \rightarrow 20\%)||(3 \rightarrow 40\%)]_{retries} + \\ &\quad [(usualtime \rightarrow 0\%)||(unusualtime \rightarrow 25\%)]_{loginTime} + \\ &\quad [(conistent \rightarrow 0\%)||(inconsistent \rightarrow 15\%)]_{userAgent}\} \end{aligned}$$

B. Multi-Factor Authentication factors

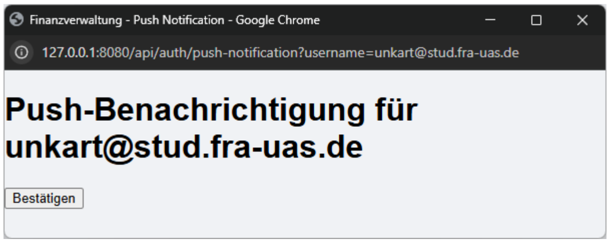
The proposed system implements four different additional layers that can be used for authentication, when the risk score implies the necessity of an additional authentication layer. Dependent on the risk score the users might get a push notification, have to answer a security question or have to enter a One Time Password (OTP) via mail or SMS. The following table I shows when which additional authentication method is required.

Riskscore	Additional Authentication Layer
0%-20%	Nothing
20%-30%	Push notification
30%-40%	Security question
40%-50%	OTP via mail
50%-100%	OTP via SMS

TABLE I
TABLE THAT SHOWS WHICH ADDTITIONAL AUTHENTICATION LAYER IS USED DEPENDENT ON THE RISK SCORE

C. Integration of the risk score and Multi-Factor authentication into the frontend

When the risk score is between 20% and 30% a push notification is simulated and sent to the user as seen in Fig. 5 When the user confirms the notification he signs in and a JWT is generated and used in further requests. When the risk score is between 30% and 40% a hidden field is shown at the sign in



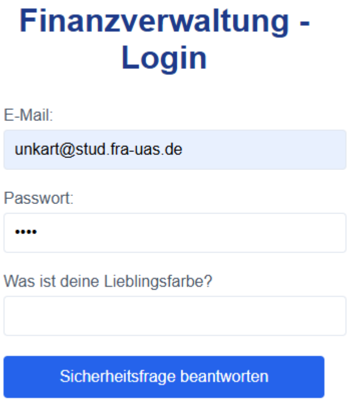
Finanzverwaltung - Push Notification - Google Chrome

127.0.0.1:8080/api/auth/push-notification?username=unkart@stud.fra-uas.de

Push-Benachrichtigung für unkart@stud.fra-uas.de

Fig. 5. Push notification of the finance management application that is send when the risk score is between 20% and 30%

page as seen in Fig. 6, that asks the user a secret question that was saved in the database previously. When the user answers the question correctly he is granted access and the JWT is generated. When the risk score is between 40% and 50% an OTP is being send to the users mail. At the sign in page, the user is shown a message that requests him to enter an OTP to proof that he has access to the mail and a hidden field to enter the OTP is made visible. The sign in page that is shown with a risk score between 40% and 50% can be seen in Fig. 7. If the user enters the OTP the JWT is generated. Finally, when the risk score is between 50% and 100% an OTP is also used, but the OTP is send to the user's phone number by SMS. This is only simulated by showing a message field on the sign in page, where the user is informed about an OTP being send to his phone number as seen in Fig. 8.



Finanzverwaltung - Login

E-Mail:

unkart@stud.fra-uas.de

Passwort:

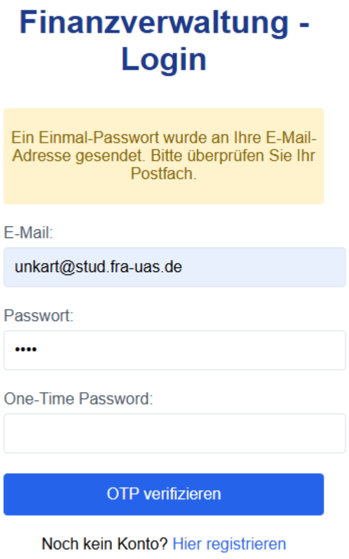
••••

Was ist deine Lieblingsfarbe?

Fig. 6. Security question field of the finance management application that is shown when the risk score is between 30% and 40%

IV. Results and Discussion

The proposed approach shows, that JWT-supported authentication, or authentication in general can be enhanced through the usage of a dynamic access control system and multi-factor authentication. When a new user is created, the database entries for this newly created user are empty. This leads to the login attempts of a new user being classified as suspicious looking login events. This happens, because at the first login attempt the factors that are considered when calculating the risk score are unknown, which leads to the risk score being rather high with a score of 60% for an unknown IP address, unknown browser consistency data and unknown usual login times. The only missing factor that stops the risk score from being the highest possible score of 100% are the failed login attempts recently. This leads to the system being enhanced with an additional authentication layer through confirmation of an OTP through SMS. When a user logs in for the second time from the same system the IP address and the browser consistency data will already be known. This knowledge leads to the risk score being lower than previously, because now



Finanzverwaltung - Login

Ein Einmal-Passwort wurde an Ihre E-Mail-Adresse gesendet. Bitte überprüfen Sie Ihr Postfach.

E-Mail:

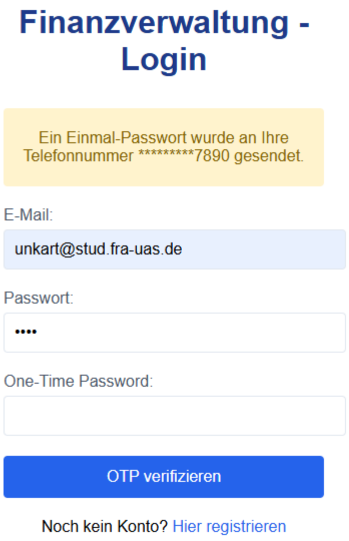
unkart@stud.fra-uas.de

Passwort:

••••

One-Time Password:

Fig. 7. Sign in page of the finance management application that is shown when the risk score is between 40% and 50% and shows the message that requests the user to enter the OTP and the field where the user has to enter the OTP



Finanzverwaltung - Login

Ein Einmal-Passwort wurde an Ihre Telefonnummer *********7890 gesendet.

E-Mail:

unkart@stud.fra-uas.de

Passwort:

••••

One-Time Password:

Fig. 8. Sign in page of the finance management application that is shown when the risk score is between 50% and 100% and shows the message that requests the user to enter the OTP send by SMS and the field where the user has to enter the OTP

the unknown factors remaining are the usual login times. Assuming there were no failed login attempts recently, the risk score will be lower, with a score of 25%. This means the system will still enhance the security with an additional layer of authentication through confirming a push notification. A user might find this additional layer inconvenient but will only get rid of the inconvenience, when a behavior pattern is established for the usual login times. The issue here is, that the DBSCAN algorithm is implemented with a minpts value of three, which leads to the user having to log in at least three times before a cluster will be formed at this weekday around the time. This means that the usual login time window will become more consistent with the number of login events stored in the database. Another issue might be, that the DBSCAN parameters are not adjusted dynamically. Since the requirement of forming a cluster is defined as three through the minpts parameter, a cluster might be formed and treated as the usual login time of the user, although it is outside of his usual login times. The risk of forming such a cluster will probably rise with an increasing number of login events stored in the database. If a user has for instance 100000 login events stored in the database, a minpts parameter of three will probably not be a good value for the parameter because in comparison to the large number of login events stored in the database, the three data points that form a small cluster will be more likely to be outliers than a true usual login times window. When taking this into consideration, a dynamically adjusted minpts parameter might make sense for a future implementation, that scales during the runtime of the application, dependent on the size of the database. A high number of failed login attempts recently indicates that suspicious activity is going on for the account of the user. If there are at least two failed login attempts the risk score will be 20% for two failed login attempts or 40% for three and more failed login attempts and an additional layer of authentication will be necessary. A user might fail to enter his credentials a few times. Failing one time will increase the risk score by only 10%, which alone will not trigger an additional authentication layer.

This work did not examine how an attacker could bypass the risk assessment part. An attacker could manipulate the request sent to the server so that his login attempt appears as not suspicious. Although an attacker might get through the system, the system makes it more difficult to get through. An attacker would have to be in possession of the credentials and has to know the factors, the risk assessment considers. If he knows those factors, he also needs to know the values of the factors for the specific user. In addition to the credentials, he also needs to know the IP address, browser, browser version, operating system and usual login time of the user while not failing the login attempts more than once. An attacker could also try to break the second authentication layer, for instance, by getting possession of the user's mail.

V. Conclusion

The system will detect suspicious login attempts and add another layer of authentication when deemed necessary, which leads to enhanced security of the system. An attacker might still get through the system, but the system makes it more difficult to do so. The approach might need further investigation especially for the minpts parameter of the DBSCAN algorithm when a large number of login events is stored in the database. The current implementation also only checks whether the IP address is known or not, which might be inconvenient when the IP address often changes. Using a geolocation service to analyze the IP address could be an interesting approach.

References

[1] B. T. Familoni and P. O. Shoetan, "Cybersecurity in the financial sector: a comparative analysis of the usa and nigeria," *Computer Science & IT Research Journal*, vol. 5, no. 4, pp. 850–877, 2024.

[2] P. V. Shevchenko, J. Jang, M. Malavasi, G. W. Peters, G. Sofronov, and S. Trück, "The nature of losses from cyber-related events: risk categories and business sectors," *Journal of Cybersecurity*, vol. 9, no. 1, p. tyac016, 2023.

[3] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki *et al.*, "Data breaches, phishing, or malware? understanding the risks of stolen credentials," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 1421–1434.

[4] C. Herley, P. C. Van Oorschot, and A. S. Patrick, "Passwords: If we're so smart, why are we still using them?" in *Financial Cryptography and Data Security: 13th International Conference, FC 2009, Accra Beach, Barbados, February 23-26, 2009. Revised Selected Papers 13*. Springer, 2009, pp. 230–237.

[5] M. Zviran and Z. Erlich, "Identification and authentication: technology and implementation issues," *Communications of the Association for Information Systems*, vol. 17, no. 1, p. 4, 2006.

[6] A. Ometov, S. Bezzateev, N. Mäkitalo, S. Andreev, T. Mikkonen, and Y. Koucheryavy, "Multi-factor authentication: A survey," *Cryptography*, vol. 2, no. 1, p. 1, 2018.

[7] S. Wiefling, L. Lo Iacono, and M. Dürmuth, "Is this really you? an empirical study on risk-based authentication applied in the wild," in *ICT Systems Security and Privacy Protection: 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings 34*. Springer, 2019, pp. 134–148.

[8] P. K. Jain, M. S. Bajpai, and R. Pamula, "A modified dbscan algorithm for anomaly detection in time-series data with seasonality." *Int. Arab J. Inf. Technol.*, vol. 19, no. 1, pp. 23–28, 2022.

[9] M. Jones, J. Bradley, and N. Sakimura, "Rfc 7519: Json web token (jwt)," 2015.

[10] A. Bucko, K. Vishi, B. Krasniqi, and B. Rexha, "Enhancing jwt authentication and authorization in web applications based on user behavior history," *Computers*, vol. 12, no. 4, p. 78, 2023.

[11] K. Shingala, "Json web token (jwt) based client authentication in message queuing telemetry transport (mqtt)," *arXiv preprint arXiv:1903.02895*, 2019.

[12] M. Haekal *et al.*, "Token-based authentication using json web token on sikasir restful web service," in *2016 International Conference on Informatics and Computing (ICIC)*. IEEE, 2016, pp. 175–179.

[13] K. A. A. Bakar and G. R. Haron, "Adaptive authentication based on analysis of user behavior," in *2014 Science and Information Conference*. IEEE, 2014, pp. 601–606.

[14] C. P. Lora, S. Ballal, and J. S. Dhanjal, "Enhancing security through the utilization of multifactor security algorithms," in *2024 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (AIMLA)*. IEEE, 2024, pp. 1–6.

[15] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz, "Anomaly detection in temperature data using dbscan algorithm," in *2011 international symposium on innovations in intelligent systems and applications*. IEEE, 2011, pp. 91–95.

[16] Wikipedia, "Feature scaling — Wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Feature%20scalingoldid=1241939671, 2025, [Online; accessed 03-February-2025].