Sichere Finanzverwaltung: Verbesserung von JWT gestützter Authentifizierung durch dynamisches Zugriffskontrollsystem und Multi-Faktor-Authentifizierung

## **Abstract**

Die Sicherheit von Finanzdaten ist in der digitalen Welt entscheidend. Immer mehr Menschen nutzen Apps, um ihre Finanzen zu verwalten und ihre Ausgaben im Blick zu behalten. In diesem Projekt wird eine Finanzmanagement-App entwickelt, die für die Authentifizierung JSON Web Tokens (JWT) und JWT durch ein dynamisches Zugriffkontrollsystem mit Multi-Faktor-Authentifizierung (MFA) kombiniert. Ziel ist es, eine sichere Anwendung zu schaffen, die Benutzer vor unbefugtem Zugriff schützt.

Es wird ein dynamisches Zugriffskontrollsystem entwickelt, das auf Basis von Analysen des Benutzerverhaltens entscheidet, ob die Verwendung von MFA erforderlich ist oder nicht. Die Daten, die zur Analyse des Benutzerverhaltens verwendet werden, beinhalten unter anderem die Anmeldehistorie, Geräteinformationen und Standortdaten. In Abhängigkeit davon, wie das Zugriffskontrollsystem entscheidet, kommen verschiedene MFA-Methoden zum Einsatz. Es können als zusätzliche Authentifizierungsmaßnahmen Passwörter, Einmalpasswörter (OTP), biometrische Daten wie Fingerabdrücke und adaptive Sicherheitsfragen eingesetzt werden. Diese Ansätze erhöhen die Sicherheit und schützen sensible Daten.

Das Projekt gliedert sich in mehrere Schritte: Zunächst erfolgt eine Analyse der Sicherheitsanforderungen und Benutzerbedürfnisse. Danach wird das System entworfen, wobei sowohl die Backend-Architektur als auch die Benutzeroberfläche berücksichtigt werden. In der Entwicklungsphase wird die App programmiert. Es wird die JWT-basierte Authentifizierung, Multi-Faktor-Authentifizierung und das dynamische Zugriffskontrollsystem implementiert. Nach der Implementierung folgt eine umfassende Testphase, um die Sicherheitsfunktionen zu validieren.

Die erwarteten Ergebnisse sind eine funktionierende Finanzmanagement-App, die sicher und benutzerfreundlich ist. Die App soll den Benutzern helfen, ihre Finanzen effizient zu verwalten und das Risiko unbefugter Zugriffe soll durch das dynamische Zugriffskontrollsystem minimiert werden. Zudem sollen neue Erkenntnisse über die Implementierung von MFA in JWT-basierten Systemen gewonnen werden, die für zukünftige Entwicklungen wichtig sein könnten.

**Projektplan**

## **1. Anforderungsanalyse**

* Identifikation der Sicherheitsanforderungen für die Anwendung.
* Analyse der Benutzerbedürfnisse hinsichtlich Funktionalität und Benutzerfreundlichkeit.
* Untersuchung potenzieller Risiken und Schwachstellen in der Anwendung.
* Erstellung eines Anforderungsdokuments, dass alle gesammelten Informationen zusammenfasst.

## **2. Systemdesign**

* Entwurf der Architektur der Anwendung, einschließlich Backend- und Frontend-Komponenten.
* Definition der JWT-Strategie für die Authentifizierung, einschließlich Token-Generierung und -Verwaltung.
* Planung des dynamischen Zugriffskontrollsystems
* Planung der Multi-Faktor-Authentifizierung (MFA) mit Auswahl geeigneter Methoden (SMS-OTP, biometrische Authentifizierung).
* Erstellung eines UML-Diagramms, um die Systemarchitektur zu visualisieren.

## **3. Implementierung**

* Programmierung der Backend-Logik zur Verwaltung von Benutzerdaten und Authentifizierung.
* Implementierung der JWT-Generierung und -Validierung im Backend.
* Entwicklung des Frontends, dass eine benutzerfreundliche Oberfläche für die Interaktion mit der Anwendung bietet.
* Integration des dynamischen Zugriffkontrollsystems und der MFA-Methoden in den Anmeldeprozess.
* Sicherstellung, dass alle sicherheitsrelevanten Daten verschlüsselt gespeichert werden.

## **4. Dokumentation**

* Erstellung einer technischen Dokumentation, die alle Aspekte des Systems beschreibt, einschließlich Architektur, API-Spezifikationen und Datenbankdesign.
* Entwicklung eines Benutzerhandbuchs, das erklärt, wie die App verwendet wird, einschließlich Anweisungen zur Anmeldung und Nutzung der MFA.
* Dokumentation von Code und Implementierungsdetails zur Unterstützung zukünftiger Wartung und Weiterentwicklung.

**5. Testphase**

* Durchführung von Funktionstests zur Überprüfung aller App-Funktionen auf korrekte Arbeitsweise.
* Durchführung von Sicherheitstests, um sicherzustellen, dass die JWT-Implementierung und MFA effektiv sind.
* Dokumentation aller Testergebnisse sowie identifizierter Fehler und deren Behebung.

## **6. Ausarbeitung**

* Erstellung einer Projektpräsentation, die den gesamten Entwicklungsprozess zusammenfasst.
* Verfassen eines Abschlussberichts, der die Ergebnisse des Projekts dokumentiert, einschließlich Herausforderungen und Lösungen während des Entwicklungsprozesses.
* Zusammenfassung von Erkenntnissen über die Implementierung von MFA in JWT-basierten Systemen.

**Zeitplan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Arbeitspaket | Dauer (in Wochen) | Startdatum | Enddatum | Meilenstein |
| Anforderungsanalyse | 1 | 13.12.24 | 20.12.24 | Abschluss Anforderungs-  analyse |
| Systemdesign | 2 | 13.12.24 | 10.01.25 | Abschluss Systemdesign |
| Implementierung | 3 | 20.12.24 | 24.01.25 | Abschluss Implemen-  tierung |
| Dokumentation | 2 | 24.01.25 | 05.02.25 | Abschluss Dokumentation |
| Testphase | 1 | 24.01.25 | 31.01.25 | Abschluss Testphase |
| Ausarbeitung | 1 | 31.01.25 | 05.02.25 | Abschluss Ausarbeitung |

## **Meilensteine**

1. **Abschluss Anforderungsanalyse** (20.12.2024)
   * Alle Anforderungen sind dokumentiert und genehmigt.
2. **Abschluss Systemdesign** (10.01.2025)
   * Das Systemdesign ist fertiggestellt und alle Diagramme sind erstellt.
3. **Abschluss Implementierung** (24.01.2025)
   * Die App ist vollständig entwickelt und alle Funktionen sind implementiert.
4. **Abschluss Dokumentation** (05.02.2025)
   * Alle technischen Dokumentationen und Benutzerhandbücher sind erstellt.
5. **Abschluss Testphase** (31.01.2025)
   * Alle Tests sind abgeschlossen und alle identifizierten Fehler wurden behoben.
6. **Abschluss Ausarbeitung** (05.02.2025)
   * Der Abschlussbericht ist fertiggestellt und das Projekt wird präsentiert.

**Wissenschaftliche Fragestellung:**

Wie kann die Sicherheit einer auf JSON Web Tokens basierten Authentifizierung in mobilen Anwendungen durch die Implementierung einer dynamischen Zugriffskontrolle und Multi-Faktor-Authentifizierung (MFA) verbessert werden?

## **Grundlagen**

1. **Tokenbasierte Authentifizierung:**
   * Erklärung der Funktionsweise von JWTs, einschließlich ihrer Struktur (Header, Payload, Signature) und der Sicherheitsmechanismen, die sie bieten.
   * Analyse der Vorteile von tokenbasierter Authentifizierung im Vergleich zu traditionellen Methoden, wie z.B. Sessions.
2. **Multi-Faktor-Authentifizierung (MFA):**
   * Definition und Bedeutung von MFA zur Verbesserung der Sicherheit in Anwendungen.
   * Übersicht über verschiedene MFA-Methoden, wie SMS-OTP, biometrische Authentifizierung und adaptive Sicherheitsfragen.
3. **Sicherheitsrisiken:**
   * Identifikation und Analyse von Risiken im Zusammenhang mit der Nutzung von JWTs, einschließlich potenzieller Angriffe wie Token-Manipulation oder Replay-Angriffe.
   * Diskussion von Best Practices zur Minimierung dieser Risiken.
4. **Benutzerverhalten:**
   * Analyse von Methoden zur Erfassung und Auswertung von Benutzerverhaltensdaten.
5. **Dynamische Zugriffskontrolle**

* Definition von dynamischer Zugriffskontrolle
* Untersuchung der Frage, unter welchen Umständen ein dynamisches Zugriffskontrollsystem auf Basis des Benutzerverhaltens MFA verwendet.

## **Aktuelle (Forschungs-)Arbeiten**

1. **M. A. Alzahrani et al. (2022)** - *Enhancing Security in Web Applications Using Multi-Factor Authentication and JWT*

* Diese Arbeit untersucht, wie die Kombination von MFA und JWT die Sicherheit von Webanwendungen verbessern kann. Es werden verschiedene Implementierungsansätze und deren Effektivität analysiert.

1. **R. K. Gupta et al. (2021)** - *A Study on Multi-Factor Authentication Techniques for Secure Access Control*

* Diese Studie befasst sich mit verschiedenen MFA-Techniken und deren Integration in bestehende Authentifizierungssysteme, einschließlich der Verwendung von JWTs zur Verbesserung der Sicherheit.

1. **A. M. Alhassan et al. (2020)** - *Token-Based Authentication: A Review of JSON Web Tokens and Its Security Implications*

* In dieser Arbeit wird die Sicherheit von JWTs untersucht, einschließlich der Herausforderungen bei der Implementierung von MFA in tokenbasierten Authentifizierungssystemen.

1. **Zhang, Y., & Zhao, H. (2019)** - *Research on the Application of Multi-Factor Authentication in Cloud Computing*

* Diese Forschung analysiert die Anwendung von MFA in Cloud-Computing-Umgebungen und diskutiert, wie JWTs als Teil dieser Sicherheitsarchitektur verwendet, werden können, um den Zugriff zu sichern.

1. **Bucko, A., Vishi, K., Krasniqi, B., & Rexha, B. (2023)** - Enhancing JWT Authentication and Authorization in Web Applications Based on User Behavior History

* Diese Arbeit stellt eine Lösung zur Verbesserung der Authentifizierung mittels JWT durch Berücksichtigung des Benutzerverhaltens vor.

**Lastenheft**

**Funktionalitäten der Software**

* **Benutzerregistrierung und -anmeldung:** Benutzer können sich registrieren und anmelden, wobei JWT für die Authentifizierung verwendet wird.
* **Multi-Faktor-Authentifizierung:** Implementierung von MFA zur Sicherstellung eines höheren Sicherheitsniveaus bei der Anmeldung. Methoden umfassen SMS-OTP, E-Mail-Bestätigung und biometrische Authentifizierung.
* **Dynamisches Zugriffskontrollsystem:** Implementierung eines Systems, das auf Basis des Benutzerverhaltens entscheidet, ob MFA verwendet werden soll und wenn ja welche Art von MFA verwendet werden soll.
* **Finanzverwaltung:** Benutzer können ihre Ausgaben verfolgen, Budgets erstellen und Berichte über ihre finanziellen Aktivitäten generieren.
* **Datenvisualisierung:** Grafische Darstellung der Ausgaben und Budgets zur besseren Übersichtlichkeit.
* **Benachrichtigungen:** Benutzer erhalten Benachrichtigungen über wichtige Ereignisse, wie z.B. das Erreichen von Budgetgrenzen oder ungewöhnliche Ausgaben.

## **Anwender-/Anwendungsproblem**

Die Software soll das Problem unzureichender Sicherheit bei der Verwaltung sensibler Finanzdaten lösen. Viele Benutzer verwenden einfache Passwörter oder keine zusätzlichen Sicherheitsmaßnahmen, was sie anfällig für Cyberangriffe macht. Durch die Implementierung von JWT, eines dynamischen Zugriffkontrollsystems und MFA wird ein höheres Maß an Sicherheit erreicht, das den Zugriff auf persönliche Finanzdaten schützt.

## **Nutzer-/System-Schnittstellen**

* **Benutzerschnittstelle (UI):** Eine intuitive Webanwendung, die es Benutzern ermöglicht, sich zu registrieren, anzumelden und ihre Finanzen zu verwalten. Mockups können einfache Layouts für Anmelde-, Registrierungs- und Dashboard-Seiten zeigen.
* **API-Schnittstelle:** Eine RESTful API, die es ermöglicht, Daten zwischen dem Frontend und Backend auszutauschen. Diese API wird Endpunkte für die Authentifizierung (JWT), Finanzdatenverwaltung und MFA-Anforderungen bereitstellen. **Beispiel für API-Endpunkte:**
  + POST /api/auth/register - Registrierung eines neuen Benutzers
  + POST /api/auth/login - Benutzeranmeldung und JWT-Generierung
  + POST /api/auth/mfa - MFA-Verifizierung

## **Technologien**

* **Programmiersprache:** Kotlin für das Backend
* **Frontend-Technologien:** Plain HTML und CSS für die Benutzeroberfläche.
* **Frameworks:**
  + **Backend:** ktor für die Erstellung der RESTful API.
  + **Datenbank:** MySQL für die Speicherung von Benutzerdaten und Finanzinformationen.
* **JWT-Bibliothek:** jsonwebtoken zur Erstellung und Verifizierung von JWTs.
* **MFA-Bibliotheken:** Verwendung von Bibliotheken wie speakeasy (für TOTP) oder Twilio (für SMS-Bestätigungen).

## **Eigenanteil**

**Vorhandene Komponenten:**

* Open-Source-Bibliothek “jsonwebtoken” für Kotlin
* Analyse des Benutzerverhaltens
  + Google Analytics
  + Hotjar
* MFA Bibliotheken und Frameworks für Kotlin
  + auth0: Push-MFA, One Time Password (OTP)
  + Spring Security: OTP
  + Android Biometric API: Fingerabdruck-, Gesichts- und Iris-Authentifizierung

**Eigenentwicklung:**

* Entwicklung des gesamten Frontend mit HTML und CSS, um eine benutzerfreundliche Oberfläche zu schaffen.
* Implementierung der Logik zur Verwaltung von Finanzen, einschließlich Budgetierung und Ausgabenverfolgung in Kotlin.
* Implementierung eines dynamischen Zugriffkontrollsystems, dass entscheidet wann und wie MFA angewendet wird. Dies beinhaltet das Entwickeln von Algorithmen, die auf Basis von Analysen des Benutzerverhaltens Entscheidungen bezüglich der Verwendung von MFA treffen.

**Beschreibung der selbst programmierten Anteile:**

1. **Frontend-Anwendung:**
   * Entwicklung einer responsiven Benutzeroberfläche mit HTML und CSS, die eine einfache Navigation ermöglicht.
   * Implementierung von Formularen zur Registrierung, Anmeldung und Budgetverwaltung.
2. **Backend-Anwendung:**
   * Erstellung einer RESTful API mit Ktor zur Handhabung aller Anfragen vom Frontend.
   * Implementierung der JWT-Generierung und -Validierung sowie der Logik zur MFA.
3. **Dynamisches Zugriffskontrollsystem und MFA-Integration:**
   * Entwicklung eines Moduls zur Verwaltung von MFA-Anforderungen basierend auf dem Benutzerverhalten.
   * Implementierung der Logik zur Versendung von OTPs per SMS oder E-Mail.
   * Implementierung der Logik zur Authentifizierung mittels Biometrie (Fingerabdruck oder Gesichtserkennung).

## **Szenarien für die Nutzung von MFA**

1. **Szenario 1: Anmelden von einem neuen Gerät**
   * **Verhalten:** Ein Benutzer meldet sich zum ersten Mal von einem neuen Gerät an.
   * **MFA-Methode:** **SMS- oder E-Mail-Verifizierung**
     + Der Benutzer erhält einen einmaligen Code (OTP) per SMS oder E-Mail, den er eingeben muss, um den Anmeldevorgang abzuschließen.
   * Diese Methode stellt sicher, dass nur der rechtmäßige Benutzer Zugriff auf das Konto hat, auch wenn jemand anderes das Passwort kennt.
2. **Szenario 2: Hohe Transaktion**
   * **Verhalten:** Der Benutzer versucht, eine große Geldüberweisung durchzuführen.
   * **MFA-Methode:** **Biometrische Authentifizierung (Fingerabdruck oder Gesichtserkennung)**
     + Der Benutzer muss seine biometrischen Daten zur Bestätigung der Transaktion verwenden.
   * Bei großen finanziellen Transaktionen ist es wichtig, sicherzustellen, dass der Benutzer tatsächlich der Kontoinhaber ist. Biometrie bietet eine zusätzliche Sicherheitsebene.
3. **Szenario 3: Verdächtige Anmeldeaktivität**
   * **Verhalten:** Ein Benutzer meldet sich von einem unbekannten Standort oder einer verdächtigen IP-Adresse an.
   * **MFA-Methode:** **Adaptive MFA mit zusätzlichen Sicherheitsfragen**
     + Der Benutzer wird aufgefordert, zusätzliche Sicherheitsfragen zu beantworten (z.B. „Was ist der Name Ihres ersten Haustiers?“).
   * Diese Methode hilft dabei, potenziell unbefugte Zugriffe zu verhindern, indem zusätzliche Informationen angefordert werden.
4. **Szenario 4: Regelmäßige Anmeldungen**
   * **Verhalten:** Der Benutzer meldet sich regelmäßig von seinem vertrauten Gerät an.
   * **MFA-Methode:** **Push-Benachrichtigung über eine Authentifizierungs-App**
     + Der Benutzer erhält eine Push-Benachrichtigung auf sein Mobilgerät und muss diese bestätigen.
   * Diese Methode ist benutzerfreundlich und schnell für regelmäßige Anmeldungen, während sie dennoch eine zusätzliche Sicherheitsebene bietet.
5. **Szenario 5: Kontoänderungen**
   * **Verhalten:** Der Benutzer versucht, seine Kontoeinstellungen zu ändern (z.B. Passwort ändern oder E-Mail-Adresse aktualisieren).
   * **MFA-Methode:** **Zwei-Faktor-Authentifizierung (2FA) mit Hardware-Token**
     + Der Benutzer muss einen physischen Hardware-Token verwenden, um die Änderungen zu bestätigen.
   * Hardware-Token bieten eine sehr hohe Sicherheit und sind besonders wichtig für kritische Änderungen im Konto.