## **Abstract**

In der heutigen digitalen Welt, in der Cyberangriffe und Datenverletzungen zunehmend an der Tagesordnung sind, ist die Sicherheit von Authentifizierungsprozessen von entscheidender Bedeutung. Das Projekt zielt darauf ab, ein sicheres Authentifizierungssystem zu entwickeln, das JSON Web Tokens (JWT) nutzt und eine dynamische Zugriffskontrolle implementiert. Dieses System wird in Webanwendungen eingesetzt, um die Integrität und Vertraulichkeit von Benutzerdaten zu gewährleisten und gleichzeitig eine benutzerfreundliche Erfahrung zu bieten. Die Zielsetzung des Projekts umfasst die Entwicklung einer robusten Authentifizierungsarchitektur, die nicht nur die Standardanmeldung über JWT ermöglicht, sondern auch zusätzliche Sicherheitsmaßnahmen integriert. Diese Maßnahmen beinhalten die dynamische Anpassung der Zugriffsrechte basierend auf Kontextfaktoren wie Benutzerverhalten, Standort und Anmeldehistorie. Die messbaren Ziele sind die Implementierung eines funktionalen Prototyps, der eine sichere Benutzerregistrierung und -anmeldung ermöglicht sowie eine adaptive Zugriffskontrolle bietet. Das Vorgehen umfasst eine systematische Analyse bestehender Authentifizierungsmethoden und Sicherheitsanforderungen, gefolgt von der Entwicklung des Systems in mehreren Phasen. Zunächst wird eine grundlegende JWT-Implementierung erstellt, gefolgt von der Integration des Moduls zur dynamischen Zugriffskontrolle. Hierbei werden relevante Kontextdaten erfasst und analysiert, um die Sicherheitsrichtlinien anzupassen. Abschließend erfolgt eine umfassende Testphase zur Validierung der Sicherheit und Benutzerfreundlichkeit des Systems. Die erwarteten Ergebnisse des Projekts sind sowohl theoretischer als auch praktischer Natur. Theoretisch wird das Projekt neue Erkenntnisse über die Implementierung dynamischer Zugriffskontrollen in Authentifizierungssystemen liefern. Praktisch wird ein funktionierendes Prototyp-System bereitgestellt, das die Sicherheit von Webanwendungen durch adaptive Authentifizierungsmechanismen verbessert und gleichzeitig den Benutzern eine intuitive Interaktion ermöglicht. Dieses Abstract bietet einen klaren Überblick über Ihr Projekt und erfüllt die Anforderungen an Kontext, Zielsetzung, Vorgehen und erwartete Ergebnisse.

## **Projektplan**

## **1. Projektvorbereitung**

* **Inhalte:**
  + Definition der Projektziele und -anforderungen.
  + Erstellung eines Lastenhefts.
  + Recherche zu bestehenden Authentifizierungsmethoden und Sicherheitsanforderungen.
* **Dauer:** Woche 1

## **2. Systemarchitektur und Design**

* **Inhalte:**
  + Entwurf der Systemarchitektur.
  + Auswahl der Technologien (z.B. Node.js, Express, MongoDB).
  + Erstellung von Diagrammen (z.B. UML-Diagramme) zur Veranschaulichung der Architektur.
* **Dauer:** Woche 1-2

## **3. Implementierung der Benutzerregistrierung und -anmeldung**

* **Inhalte:**
  + Entwicklung der API-Routen für die Registrierung und Anmeldung.
  + Implementierung der JWT-Generierung und -Verifizierung.
  + Dokumentation des Codes.
* **Dauer:** Woche 2-3

## **4. Entwicklung des Moduls zur dynamischen Zugriffskontrolle**

* **Inhalte:**
  + Programmierung des Moduls zur Erfassung von Kontextdaten (z.B. IP-Adresse, Standort).
  + Implementierung der Logik zur Anpassung der Zugriffsrechte basierend auf Kontextfaktoren.
  + Dokumentation des Moduls.
* **Dauer:** Woche 3-4

## **5. Integration von Sicherheitsrichtlinien**

* **Inhalte:**
  + Entwicklung von adaptiven Sicherheitsrichtlinien für verschiedene Risikostufen.
  + Implementierung zusätzlicher Sicherheitsmaßnahmen (z.B. MFA bei verdächtigen Anmeldungen).
  + Dokumentation der Sicherheitsrichtlinien.
* **Dauer:** Woche 4-5

## **6. Testen und Validieren des Systems**

* **Inhalte:**
  + Durchführung von Funktionstests und Sicherheitstests.
  + Behebung von Fehlern und Optimierung des Systems basierend auf Testergebnissen.
  + Dokumentation der Testergebnisse.
* **Dauer:** Woche 5-6

## **7. Erstellung der Ausarbeitung**

* **Inhalte:**
  + Zusammenstellung aller Projektdokumentationen.
  + Erstellung einer abschließenden Projektdokumentation mit theoretischen Grundlagen, Implementierungsdetails und Testergebnissen.
* **Dauer:** Woche 6

## **8. Vorbereitung der Präsentation**

* **Inhalte:**
  + Erstellung einer Präsentation zur Vorstellung des Projekts.
  + Übung der Präsentation im Team.
* **Dauer:** Woche 6

## **Zeitplan**

| **Arbeitspaket** | **Dauer** | **Verantwortlich** |
| --- | --- | --- |
| 1. Projektvorbereitung | Woche 1 | Projektleiter |
| 2. Systemarchitektur und Design | Woche 1-2 | Technischer Architekt |
| 3. Implementierung Benutzerregistrierung | Woche 2-3 | Entwickler |
| 4. Entwicklung Modul dynamische Zugriffskontrolle | Woche 3-4 | Entwickler |
| 5. Integration Sicherheitsrichtlinien | Woche 4-5 | Sicherheitsspezialist |
| 6. Testen und Validieren | Woche 5-6 | QA-Ingenieur |
| 7. Erstellung der Ausarbeitung | Woche 6 | Projektleiter |
| 8. Vorbereitung der Präsentation | Woche 6 | Projektleiter |

## **Meilensteine**

1. Abschluss von Konzept und Anforderungsanalyse (Ende Woche 1)
2. Abschluss des Systemdesigns (Ende Woche 2)
3. Fertigstellung der Benutzerregistrierungs- und Anmeldemodule (Ende Woche 3)
4. Abschluss des Moduls zur dynamischen Zugriffskontrolle (Ende Woche 4)
5. Integration der Sicherheitsrichtlinien (Ende Woche 5)
6. Abschluss aller Tests und Validierungen (Ende Woche 6)
7. Fertigstellung der Projektdokumentation (Ende Woche 6)
8. Durchführung der Präsentation (Ende Woche 6)

## **1. Welche Fragestellung soll im Rahmen des Projekts bearbeitet werden?**

Die zentrale wissenschaftliche Fragestellung lautet: **"Wie kann ein sicheres Authentifizierungssystem, das JSON Web Tokens (JWT) verwendet, durch die Implementierung einer dynamischen Zugriffskontrolle die Sicherheit und Benutzerfreundlichkeit in modernen Webanwendungen verbessern?"** Diese Frage wird durch die Analyse der Sicherheitsmechanismen von JWT und der Effektivität dynamischer Zugriffskontrollmethoden untersucht.

## **2. Welche Grundlagen werden beschrieben und ausgearbeitet?**

Im Rahmen des Projekts werden folgende Grundlagen beschrieben und ausgearbeitet:

* **JSON Web Tokens (JWT):** Die Struktur, Funktionsweise und Sicherheitsaspekte von JWTs, einschließlich der Herausforderungen und Schwachstellen (z.B. Verwendung des "none"-Algorithmus).
* **Dynamische Zugriffskontrolle:** Definition und Merkmale der dynamischen Zugriffskontrolle, einschließlich der Anpassungsfähigkeit an Benutzerverhalten und Kontextinformationen.
* **Sicherheitsmechanismen:** Eine Analyse der Sicherheitsmechanismen, die in Authentifizierungssystemen implementiert werden können, um unbefugten Zugriff zu verhindern.
* **Best Practices:** Empfehlungen für die sichere Implementierung von JWTs und dynamischer Zugriffskontrolle in Webanwendungen.

## **3. Welche (weiterführenden) aktuellen (Forschungs-)Arbeiten gibt es dazu?**

Es gibt mehrere aktuelle Forschungsarbeiten und Artikel, die sich mit den Themen JWT und dynamische Zugriffskontrolle befassen:

* **Penetrationstest von JSON Web Tokens:** Diese Bachelorarbeit untersucht die Sicherheit von JWTs in der IoT-Plattform Thingsboard und analysiert, wie JWTs als Authentifizierungsmethode manipuliert werden können. Diese Arbeit zeigt auf, welche Schwachstellen existieren und wie sie ausgenutzt werden könnten
* **Dynamische Zugriffskontrolle:** Verschiedene Artikel beschreiben die Vorteile der dynamischen Zugriffskontrolle im Vergleich zu traditionellen Methoden. Sie betonen, dass dynamische Systeme flexibler sind und eine bessere Anpassung an sich ändernde Bedingungen ermöglichen
* **Sicherheitsrisiken von JWTs:** Studien haben gezeigt, dass JWTs häufige Angriffsvektoren für fehlerhafte Nutzerauthentifizierung darstellen. Diese Arbeiten analysieren spezifische Risiken und geben Empfehlungen zur Verbesserung der Sicherheit
* **Best Practices für JWT-Sicherheit:** Es gibt zahlreiche Ressourcen, die bewährte Methoden für den Umgang mit JWTs beschreiben, einschließlich Token-Ablaufzeiten, sichere Übertragung und Validierung

## **1. Was soll die fertige Software an Funktionalitäten bieten?**

Die fertige Software soll folgende Funktionalitäten bieten:

* **Benutzerregistrierung:** Ermöglicht neuen Benutzern die Erstellung eines Kontos.
* **Benutzerauthentifizierung:** Validierung der Anmeldedaten und Rückgabe eines JWT bei erfolgreicher Authentifizierung.
* **Dynamische Zugriffskontrolle:** Anpassung der Zugriffsrechte basierend auf Kontextfaktoren wie Standort, Gerät und Benutzerverhalten.
* **Multi-Faktor-Authentifizierung (MFA):** Zusätzliche Sicherheitsüberprüfungen bei verdächtigen Anmeldungen oder unüblichen Zugriffen.
* **Token-Management:** Verwaltung von JWTs, einschließlich Ablaufzeiten und Widerruf von Tokens.
* **API-Schutz:** Sicherer Zugriff auf geschützte Routen und Ressourcen durch Überprüfung der JWTs in den Autorisierungsheadern.

## **2. Welches (Anwender-/Anwendungs-)Problem soll die Software lösen?**

Die Software soll das Problem der unsicheren Authentifizierungsprozesse in modernen Webanwendungen lösen. Insbesondere zielt sie darauf ab,:

* Die Risiken von unbefugtem Zugriff durch die Implementierung dynamischer Zugriffskontrollen zu minimieren.
* Die Benutzerfreundlichkeit zu verbessern, indem eine flexible Authentifizierungsmethode bereitgestellt wird, die sich an das Benutzerverhalten anpasst.
* Die Sicherheit durch zusätzliche Maßnahmen wie MFA zu erhöhen, um Benutzerkonten besser zu schützen.

## **3. Welche (Nutzer-/System-)Schnittstellen gibt es? Wie sehen diese aus?**

## **Schnittstellen**

* **Benutzerschnittstelle (UI):** Eine Webanwendung mit Formularen für die Registrierung und Anmeldung sowie einer Benutzeroberfläche zur Anzeige von Benutzerprofilinformationen und Einstellungen.
* **API-Schnittstelle:**
  + **POST /register:** Registrierung eines neuen Benutzers (Eingabe: E-Mail, Passwort).
  + **POST /login:** Authentifizierung des Benutzers (Eingabe: E-Mail, Passwort; Ausgabe: JWT).
  + **GET /protected:** Zugriff auf geschützte Ressourcen (Überprüfung des JWT im Header).
  + **POST /token/refresh:** Erneuerung eines abgelaufenen Tokens.

Grobarchitektur:

+-------------------+ +-------------------+

| Benutzer- | | Authentifizierungs|

| oberfläche | <------> | API |

| (Frontend) | | (Backend) |

+-------------------+ +-------------------+

|

|

v

+---------------------------+

| Datenbank |

| (Benutzerdaten) |

+---------------------------+

## **4. Welche Technologien sollen verwendet werden?**

## **Frameworks und Bibliotheken**

* **Backend:**
  + **Kotlin**
  + **SQL Datenbank für Benutzerspeicherung**
  + **jsonwebtoken** für die Erstellung und Verifizierung von JWTs.
* **Frontend:**
  + **HTML und CSS** für die Entwicklung der Benutzeroberfläche.

## **Entwicklungsumgebung**

* Verwendung von Visual Studio Code oder einer ähnlichen IDE.
* Postman für API-Tests.

## **5. Eigenanteil: Was ist vorhanden? Was muss selbst programmiert werden?**

## **Vorhandene Komponenten**

* Es können Open-Source-Bibliotheken wie jsonwebtoken und Express.js verwendet werden, um grundlegende Authentifizierungsfunktionen zu implementieren.

## **Selbst programmierte Anteile**

1. **Dynamische Zugriffskontrolle:**
   * Programmierung des Moduls zur Erfassung und Analyse von Kontextdaten zur Anpassung der Zugriffsrechte.
2. **Multi-Faktor-Authentifizierung:**
   * Implementierung zusätzlicher Sicherheitsmaßnahmen zur Validierung von Anmeldungen.
3. **Token-Management-System:**
   * Entwicklung eines Systems zur Verwaltung von Token-Ablaufzeiten und Widerrufmechanismen.

## **6. Beschreibung der selbst programmierten Anteile, inkl. deren Funktionalität und Bedeutung für das Gesamtprojekt**

Die selbst programmierten Anteile sind entscheidend für die Verbesserung der Sicherheit und Benutzerfreundlichkeit des Authentifizierungssystems:

* **Dynamische Zugriffskontrolle:** Dieses Modul ermöglicht es dem System, sich an unterschiedliche Nutzungsszenarien anzupassen und erhöht somit die Sicherheit durch kontextbasierte Entscheidungen über den Zugriff auf Ressourcen.
* **Multi-Faktor-Authentifizierung:** Diese Funktion bietet eine zusätzliche Sicherheitsebene, insbesondere bei verdächtigen Anmeldeversuchen, was das Risiko von unbefugtem Zugriff erheblich verringert.
* **Token-Management-System:** Durch effektives Management von Tokens wird sichergestellt, dass nur autorisierte Benutzer Zugang zu geschützten Ressourcen haben. Dies trägt zur Gesamtintegrität des Systems bei.