Unsichere API

Softwaretechnik-Projekt IT-Sicherheit
Wintersemester 2024/25

Pflichtenheft

Hochschule Mittweida

Betreuer: M.Sc Engler, Philipp

Name	Email
Michael P.	mprimke@hs-mittweida.de
Jonas G.	jgermann@hs-mittweida.de
Noah H.	nhiller@hs-mittweida.de
Franz S.	fsander@hs-mittweida.de
Jakob H.	jhindemi@hs-mittweida.de
Hannes L.	hlange3@hs-mittweida.de
Jon R.	jroemmli@hs-mittweida.de

Mittweida 21.01.2025

Inhaltsverzeichnis

1. Ziel / Produktzweck

Das Softwareprodukt „Unsichere API“ dient als Demonstrationsplattform, um typische Sicherheitslücken in APIs aufzuzeigen. Es soll Studierenden, Entwicklern und Sicherheitsexperten ermöglichen, potenzielle Schwachstellen zu verstehen und sichere Entwicklungspraktiken zu fördern.

1.1. Musskriterien:

- Das Produkt muss typische API-Funktionen eines Online-Shops simulieren, wie Registrierung, Login, Produktkatalog und Warenkorb.
- Es müssen absichtlich implementierte Sicherheitslücken enthalten sein, die häufig in der Praxis vorkommen (z. B. XSS, schwache Authentifizierung).
- Eine begleitende Dokumentation der Sicherheitslücken muss enthalten sein, um deren Entdeckung und Analyse zu erleichtern.

1.2. Wunschkriterien:

- Das Produkt sollte visuell ansprechende Berichte oder Dashboards enthalten, um die Auswirkungen der Sicherheitslücken zu verdeutlichen.
- Es sollte mit verschiedenen Schwierigkeitsgraden konfigurierbar sein, um unterschiedliche Zielgruppen anzusprechen.
- Die API sollte eine einfache Möglichkeit bieten, weitere Sicherheitslücken für Demonstrationszwecke hinzuzufügen.

1.3. Abgrenzungskriterien:

- Es werden keine realen Zahlungssysteme oder vollständige Shop-Funktionalitäten implementiert.
- Das Produkt wird ausschließlich für Schulungs- und Demonstrationszwecke entwickelt und ist nicht für den produktiven Einsatz vorgesehen.

2. Produkteinsatz

2.1. Anwendungsbereich:

Das Softwareprodukt wird im Rahmen von Vorlesungen, Workshops und Schulungen eingesetzt, um typische Sicherheitsprobleme in der API-Entwicklung aufzuzeigen. Es dient als Lern- und Übungsplattform für Studierende und Entwickler.

2.2. Zielgruppen:

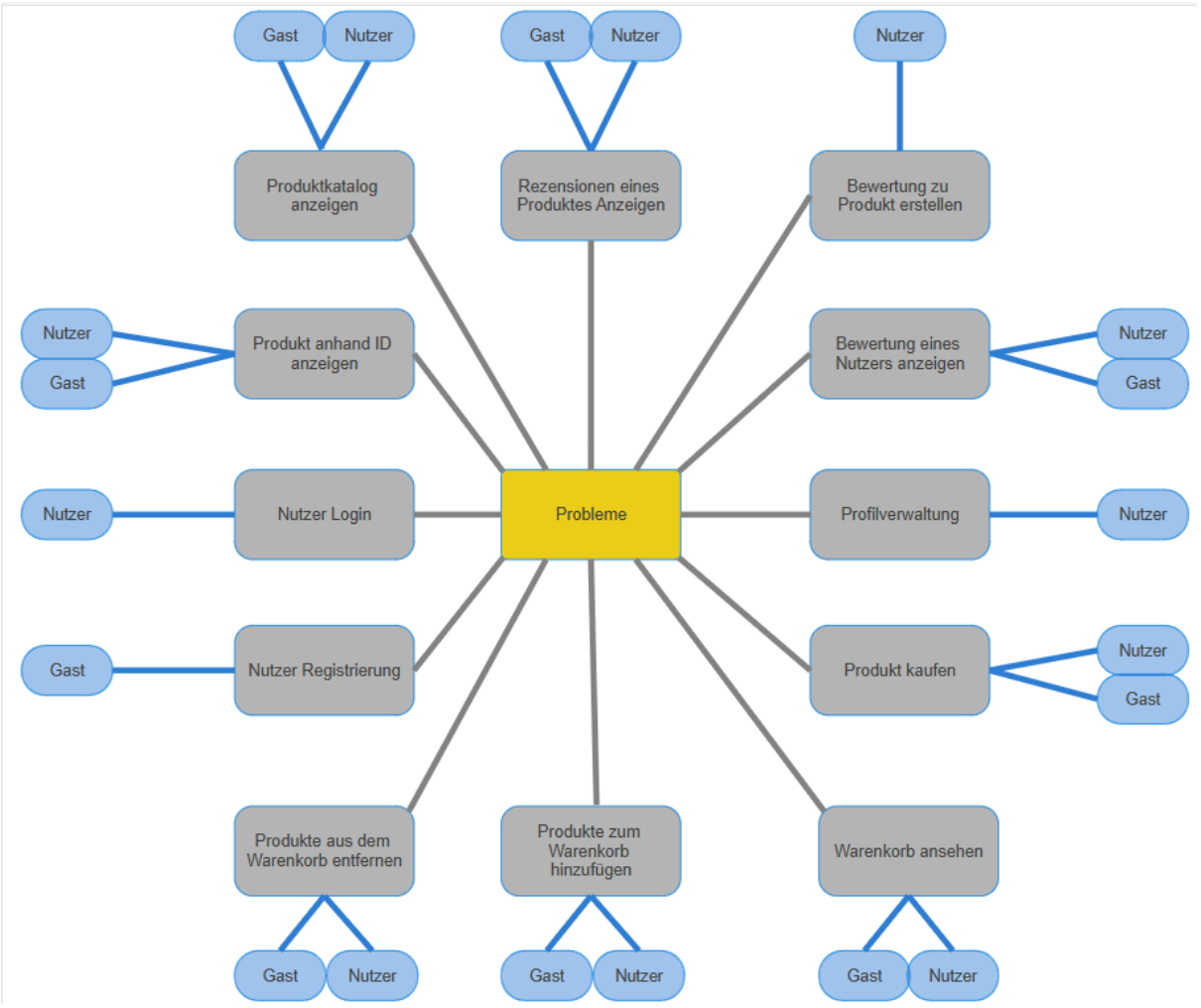
- Studierende der Informatik und verwandter Fachrichtungen
- Entwickler, die sich im Bereich API-Sicherheit weiterbilden möchten
- IT-Sicherheitsexperten, die Schulungen oder Workshops durchführen

2.3. Betriebsbedingungen:

- Das Produkt wird in einer isolierten Umgebung oder auf Schulungs Servern ausgeführt, um reale Sicherheitsrisiken auszuschließen.
- Es wird auf Standard-Webtechnologien basieren (z. B. Node.js, REST-API), die auf üblichen Betriebssystemen und in Cloud-Umgebungen lauffähig sind.

3. Produktübersicht

Abbildung 1: Interaktion des Produkts



4. Produktfunktionen

/PF10/
Funktion: Nutzer-Login
Akteur: Gast
Beschreibung: Ein Gast loggt sich mit den persönlichen Daten ein.

/PF20/
Funktion: Nutzer-Registrierung
Akteur: Gast
Beschreibung: Der Gast legt sich ein Konto an, indem er einen Namen und ein Passwort anlegt.

/PF30/
Funktion: Produktkatalog anzeigen
Akteur: Gast, Nutzer
Beschreibung: Der Nutzer kann die Inhalte des Kataloges jederzeit ansehen.

/PF40/
Funktion: Produkt anhand ID anzeigen
Akteur: Gast, Nutzer
Beschreibung: Produkte können mit ihrer ID gesucht und angezeigt werden.

/PF45/
Funktion: Produkt Rezensionen Anzeigen
Akteur: Gast, Nutzer
Beschreibung: Gast/Nutzer kann die Rezensionen zu einem Produkt anzeigen lassen.

/PF50/
Funktion: Produkt kaufen
Akteur: Gast, Nutzer
Beschreibung: Benutzer können Produkte kaufen.

/PF60/
Funktion: Produktbewertung erstellen
Akteur: Nutzer
Beschreibung: Nutzer können Produktbewertungen erstellen.

/PF70/

Funktion: Produkte zum Warenkorb hinzufügen
Akteur: Gast, Nutzer
Beschreibung: Gast/Nutzer kann Produkte zum Warenkorb hinzufügen

/PF80/

Funktion: Warenkorb ansehen
Akteur: Gast, Nutzer
Beschreibung: Gast/Nutzer kann den Warenkorb und dessen Inhalt sehen.

/PF85/

Funktion: Produkte aus Warenkorb entfernen
Akteur: Gast, Nutzer
Beschreibung: Gast/Nutzer kann Produkte aus dem Warenkorb entfernen.

/PF90/

Funktion: Nutzer Rezensionen anzeigen
Akteur: Nutzer
Beschreibung: Nutzer können die Rezensionen von anderen Nutzern anzeigen lassen.

5. Produktdaten

/PD10/ Nutzerkontodaten
Maximale Anzahl: ca. 1.000 Nutzerkonten
Beschreibung: Enthält die persönlichen Daten von Nutzern, wie Benutzername, Passwort (nicht gehasht) und E-Mail-Adresse.

/PD20/ Produktkatalogdaten
Maximale Anzahl: 1.000 Produkte
Beschreibung: Enthält Informationen zu Produkten wie Produkt-ID, Name, Beschreibung, Preis, Verfügbarkeit, Rabatt und ein Thumbnail.

/PD30/ Warenkorbdaten
Maximale Anzahl: ca. 10.000 aktive Warenkörbe
Beschreibung: Temporäre Speicherung von Produkt-IDs, Mengen und Zwischensummen für jeden Nutzer, der Produkte in den Warenkorb gelegt hat.

/PD40/ Rezensionsdaten
Maximale Anzahl: ca. 1.000 Rezensionen pro Produkt
Beschreibung: Speichert Bewertungen von Nutzern mit Informationen wie Bewertungspunktzahl, Rezensionstext, Datum der Erstellung und der Nutzer-ID des Verfassers.

/PD50/ Bestelldaten
Maximale Anzahl: ca. 1.000.000 Bestellungen
Beschreibung: Enthält die Bestell-ID, User-ID, den Zeitstempel, Produktdetails (Produkt-ID und Menge).

/PD60/ Sitzungsdaten
Maximale Anzahl: ca. 100.000 aktive Sitzungen
Beschreibung: Temporäre Speicherung von Sitzungs-IDs, Nutzerdaten und Ablaufzeiten für eingeloggte Nutzer..

/PD70/ Authentifizierungsdaten
Maximale Anzahl: ca. 1.000 aktive Tokens
Beschreibung: Speichert temporäre JWT-Token, die zur Authentifizierung der Benutzer verwendet werden. Es existiert kein Ablaufdatum für die Tokens und es ist ein schwaches Secret zur Signierung der Tokens festgelegt.

6. Produktleistungen

/PL10/ Die Ladezeit des Produktkatalogs
Die Ladezeit des gesamten Produktkatalogs darf bei einem Zugriff auf den Server maximal 2 Sekunden betragen, bei einer durchschnittlichen Netzverbindung (100 Mbit/s).

/PL20/ Die Aufrufzeit eines spezifischen Produkts
Die Anzeige der Details eines Produkts anhand seiner ID darf maximal 1 Sekunde in Anspruch nehmen.

/PL30/ Die Antwortzeit des Warenkorbs
Das Hinzufügen eines Produkts zum Warenkorb muss innerhalb von 1,5 Sekunden abgeschlossen sein.

/PL40/ Die Antwortzeit beim Nutzer-Login
Die Authentifizierung eines Nutzers darf maximal 1 Sekunde dauern, einschließlich Token-Generierung.

/PL50/ Die Ladezeit der Nutzerrezensionen
Die Anzeige der Rezensionen eines Produkts darf bei maximal 100 Rezensionen nicht länger als 1,5 Sekunden dauern.

/PL60/ Die Belastbarkeit der API
Die API muss mindestens 500 gleichzeitige Anfragen verarbeiten können, ohne die oben definierten Antwortzeiten zu überschreiten

7. Qualitätsanforderungen

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität		**X**		
Angemessenheit		**X**		
Richtigkeit				**X**
Interoperabilität				**X**
Ordnungsmäßigkeit			**X**	
Sicherheit				**X**
Zuverlässigkeit			**X**	
Reife				**X**
Fehlertoleranz				**X**
Wiederherstellbarkeit				**X**
Benutzbarkeit		**X**		
Verständlichkeit	**X**			
Erlernbarkeit			**X**	
Bedienbarkeit		**X**		
Effizienz			**X**	
Zeitverhalten				**X**
Verbrauchsverhalten				**X**
Änderbarkeit	**X**			
Analysierbarkeit		**X**		

Modifizierbarkeit		**X**		
Stabilität				**X**
Prüfbarkeit			**X**	
Übertragbarkeit			**X**	
Anpassbarkeit		**X**		

8. Benutzeroberfläche (GUI)

Die Benutzeroberfläche der Anwendung wurde so gestaltet, dass sie eine einfache und intuitive Bedienung ermöglicht. Das Layout ist klar strukturiert, mit einer Navigationsleiste für den schnellen Zugriff auf alle wichtigen Funktionen wie Produktübersicht, Warenkorb und Benutzerkonto. Die Produktseite zeigt eine Liste der verfügbaren Produkte mit Details wie Name, Preis und Verfügbarkeit, die durch Eingabe von Suchbegriffen oder Filteroptionen angepasst werden kann. Benutzer können sich registrieren, anmelden und ihre Bestellungen einsehen. Die Registrierung und Anmeldung erfolgt über Formulare, bei denen Benutzerdaten wie Benutzername und Passwort eingegeben werden.

Das Frontend nutzt Bootstrap für die Gestaltung und sorgt für ein responsives Design, das sich an verschiedene Bildschirmgrößen anpasst. Die Kommunikation mit dem Backend erfolgt über eine RESTful API und JSON-Daten. Hierbei wird jedoch keine Verschlüsselung der Daten zwischen Frontend und Backend durchgeführt. Die Anwendung enthält zudem auch keine weiteren Validierungen auf der Clientseite, sodass Angriffe wie XSS ermöglicht werden.

In Bezug auf die Benutzerrollen gibt es derzeit nur eine Standardrolle (der Benutzer), die Zugriff auf alle Shop-Funktionen hat. Weitere Benutzerrollen wie Administratoren sind derzeit nicht implementiert, könnten jedoch in Zukunft hinzugefügt werden. Dies könnte auch die Erweiterung um ein Admin-Panel mit sich bringen, welches einen weiteren umfangreichen Sicherheitsaspekt mit sich bringt.

9. Nichtfunktionale Anforderungen

/NF10/ Sicherheitsanforderungen
Beschreibung: Die API soll bewusst eingebaute Sicherheitsrisiken enthalten, dazu gehören:

- fehlende/mangelnde Eingabevalidierung
- fehlende Rate Limiting
- unsichere Speicherung und Verwaltung von Passwörtern
- fehlendes Logging
- nicht eingeschränkter Zugriff auf sensible Daten

/NF20/ Plattformunabhängigkeit
Die Anwendung soll durch Docker-Container auf Windows, Linux und macOS gleichermaßen lauffähig sein, um flexible Entwicklung und Bereitstellung zu gewährleisten.

/NF30/ Anwenderfreundlich
Das Produkt muss anwenderfreundlich sein (intuitive Bedienbarkeit für Benutzer ohne EDV-Vorkenntnisse, umfangreiche Hilfefunktion)

/NF40/ Erweiterbarkeit und Wartungsfreundlichkeit
Das Produkt muss mit geringem Aufwand weiterentwickelbar und wartbar sein

10. Produkt-Schnittstellen

10.1. Softwareschnittstellen

Betriebssystem:

- Windows, Linux, MacOS (Plattformunabhängig)

Laufzeitumgebung:

- Node.js (Serverseitige Anwendung)

Datenbank:

- PostgreSQL (relationale Datenbank für persistente Speicherung)

Frameworks und Bibliotheken:

- Express.js (API-Entwicklung)
- JWT (JSON Web Token-Handling)

Entwicklungstools:

- Docker (Containerisierung der Anwendung und Datenbank)
- GitHub (Code-Versionierung und Kollaboration)

10.2. Hardwareschnittstellen

Minimale Systemanforderungen

- CPU: Dual Core Prozessor
- RAM: 4GB
- Festplattenspeicher: 500 MB

Peripheriegeräte

- Netzwerkzugriff für API-Aufrufe und Datenbankverbindungen
- Bildschirm und Tastatur für lokale Administration

10.3. Schnittstellen zur Produktfamilie

Containerisierung:

- Docker

Code-Kollaboration und Fernwartung

- GitHub als zentrales Repository

11. Spezielle Anforderungen an Entwicklungsumgebung

- Node.js
- Express.js
- Postgres.js
- PostgreSQL
- Docker
- MySweb
- Swagger

12. Gliederung in Teilprodukte

Step 1: Nutzerverwaltung und Authentifizierung

- Registrierung, Login und Profilverwaltung
- Implementierung der Sicherheitsmechanismen (z. B. Passwort-Hashing, JWT für Authentifizierung)

Step 2: Produktkatalog und Suchfunktion

- Anzeige des Produktkatalogs
- Detailansicht einzelner Produkte (inkl. Rezensionen)

Step 3: Warenkorb und Checkout

- Hinzufügen und Entfernen von Produkten im Warenkorb
- Bestellung abschließen

Step 4: Rezensionen und Bewertungen

- Nutzer können Produkte bewerten und Rezensionen hinterlassen

Step 5: Sicherheitslücken demonstrieren

- Einbau absichtlich unsicherer Funktionen (z. B. XSS) für Lehrzwecke
- Dokumentation der Sicherheitslücken und deren Auswirkungen

13. Ergänzungen

Installationsbedingungen

- Docker muss auf dem Zielsystem installiert sein
- Docker Compose muss installiert sein
- *docker compose up* um die API zu starten (*--build* hinzufügen falls änderungen vorgenommen wurden)

14. Glossar

Gast:
ist ein neuer Interessent, der an Produkten interessiert ist. Die Registrierung ist noch nicht oder bewusst nicht erfolgt.

Nutzer:
ein Gast, der entschieden hat, sich zu registrieren.

Nutzerrezension:
eine Bewertung oder Meinung, die ein Nutzer zu einem Produkt äußert. Diese besteht aus einer Sterne Bewertung von 1-5 und optional ein Kommentar und eventuell Bilder und Videos.

Onlineshop:
eine Website oder Plattform, auf der Produkte
über das Internet angeboten und verkauft werden. Kunden können dort stöbern, bestellen und bezahlen, ohne ein physisches Geschäft zu besuchen.

API (Application Programming Interface):
Eine Schnittstelle, die es ermöglicht, Softwarekomponenten miteinander zu verbinden und Daten oder Funktionen auszutauschen.

JWT (JSON Web Token):
Ein kompakter, URL-sicherer Standard zur Übertragung von Informationen zwischen zwei Parteien als JSON-Objekt. Wird oft zur Authentifizierung verwendet.

XSS (Cross-Site Scripting):
Eine Sicherheitslücke, bei der Angreifer schädliches JavaScript einschleusen können, das bei anderen Nutzern im Browser ausgeführt wird.

Rate Limiting:
Eine Technik, um die Anzahl der Anfragen an eine API oder einen Server innerhalb eines bestimmten Zeitraums zu begrenzen, um Missbrauch oder Überlastung zu verhindern.

Hashing:
Ein kryptographischer Prozess, bei dem eine Eingabe (z. B. ein Passwort) in eine feste Länge umgewandelt wird, um Daten sicher zu speichern. Beispiele: bcrypt, SHA-256.

HTTPS (HyperText Transfer Protocol Secure):
Eine Erweiterung von HTTP, die eine verschlüsselte Kommunikation zwischen dem Nutzer und der Website sicherstellt.

SQL (Structured Query Language):
Eine Sprache zur Verwaltung und Abfrage von Daten in relationalen Datenbanken.

Token:
Ein digitaler Schlüssel, der oft für die Authentifizierung oder Autorisierung verwendet wird, z. B. in API-Interaktionen.

OWASP (Open Web Application Security Project):
Eine gemeinnützige Organisation, die sich auf die Verbesserung der Sicherheit von Softwareanwendungen konzentriert.

Prepared Statements:
Eine Methode zur sicheren Ausführung von SQL-Abfragen, die

verhindert, dass Benutzereingaben direkt in SQL-Abfragen eingefügt werden.

Output Encoding:
Die Umwandlung von Zeichen in sichere Darstellungen, um sicherzustellen, dass Eingaben nicht als ausführbarer Code interpretiert werden.

DSGVO (Datenschutz-Grundverordnung):
Eine EU-Verordnung, die Regeln für den Schutz personenbezogener Daten vorgibt.

Bcrypt:
Eine Passwort-Hashing-Funktion, die zur sicheren Speicherung von Passwörtern verwendet wird, da sie zusätzliche Sicherheitsmaßnahmen wie Salting integriert.

Salting:
Ein Verfahren, bei dem ein zufälliger Wert zu einem Passwort hinzugefügt wird, bevor es gehasht wird, um Hash-Wiederholungen zu verhindern.