



netidee

PROJEKTE

WebSecBot

Endbericht | Call 18 | Projekt ID 6878

Lizenz CC BY-SA

Inhalt

1	Einleitung.....	3
2	Projektbeschreibung.....	3
3	Verlauf der Arbeitspakete.....	8
3.1	Arbeitspaket 1 - <i>Detailplanung und Formales am Projektstart</i>	8
3.2	Arbeitspaket 2 - <i>Evaluierung lokaler KI-Modelle & Finetuning</i>	8
3.3	Arbeitspaket 3 - <i>Server-Setup</i>	9
3.4	Arbeitspaket 4 - <i>Browser-Addon & LLM-Schnittstelle</i>	9
3.5	Arbeitspaket 5 - <i>Prompt-Engineering für interaktive Websecurity-Anwendungen</i>	10
3.6	Arbeitspaket 6 - <i>Testing und Evaluierung</i>	11
3.7	Arbeitspaket 7 - <i>Dokumentation und Formales am Projektende</i>	11
4	Umsetzung Förderauflagen	11
5	Liste Projektergebnisse	11
6	Verwertung der Projektergebnisse in der Praxis.....	12
7	Öffentlichkeitsarbeit/ Vernetzung	12
8	Eigene Projektwebsite	13
9	Geplante Aktivitäten nach netidee-Projektende	13
10	Anregungen für Weiterentwicklungen durch Dritte	13

1 Einleitung

Das Projekt WebSecBot wurde im Zeitraum von Januar 2024 bis Juni 2025 durchgeführt. Das Hauptprojektergebnis ist ein Browser-Addon sowie eine Server-Komponente, welche es Entwickler:innen ermöglicht, Web-Anwendungen auf Sicherheitslücken zu testen und zu analysieren. Unser Ziel ist es, mit WebSecBot die Sicherheit von Webapplikationen zu erhöhen, indem wir Entwickler:innen ohne spezielles Fachwissen im Bereich der IT-Sicherheit ein leicht zugängliches Werkzeug zur Verfügung stellen, mit dem sie ihre Webanwendungen prüfen und verbessern können.

Das WebSecBot-Browser-Addon ist für Google Chrome verfügbar und die dazugehörige Dokumentation und Sourcecode sind auf Github verfügbar.

2 Projektbeschreibung

In einer digital vernetzten Welt ist die Sicherheit von Webapplikationen nicht mehr nur eine Option, sondern eine Notwendigkeit. Mangels Sicherheitsfachwissen und verfügbarer Ressourcen werden Applikationen oft unzureichend geprüft, bevor sie online gehen, was zu ausnutzbaren Schwachstellen führt.

WebSecBot richtet sich primär an Entwickler:innen und kleine Unternehmen, die nicht die finanziellen Mittel für einen professionellen Penetrationstest ihrer Webapplikation zur Verfügung haben, aber diese dennoch absichern wollen. Das Projekt fördert das Shift-Left-Security-Konzept, indem es Entwickler:innen bei der Erkennung und Behebung von Sicherheitsproblemen bereits während der Entwicklungsphase unterstützt.

Eine besondere technische Leistung des WebSecBot-Projekts liegt in der Aufbereitung und Integration des umfangreichen OWASP-Wissens. Wir haben hunderte von Dokumenten, Richtlinien, Best Practices und Sicherheitsempfehlungen aus dem OWASP-Projekt strukturiert, kategorisiert und in einem RAG-fähigen (Retrieval-Augmented Generation) Format aufbereitet. Dies ermöglicht es der KI, präzise und kontextbezogene Informationen aus diesem umfassenden Wissensspeicher abzurufen und in die Analyseergebnisse sowie Beratungsgespräche zu integrieren. Dadurch profitieren Nutzer:innen von hochwertigem Sicherheitswissen, das normalerweise nur Sicherheitsexpert:innen zugänglich ist, in einer benutzerfreundlichen und anwendungsorientierten Form.

Der typische Workflow beginnt mit der Installation des Addons im Chrome-Browser. Nach der Installation erscheint ein neues Symbol in der Browser-Leiste, über das der WebSecBot aktiviert werden kann. Sobald der Nutzer auf einer Webseite ist, die analysiert werden soll, kann er den WebSecBot öffnen. Das Interface bietet zwei Hauptfunktionen: Die Sicherheitsanalyse und den interaktiven Security-Chat.

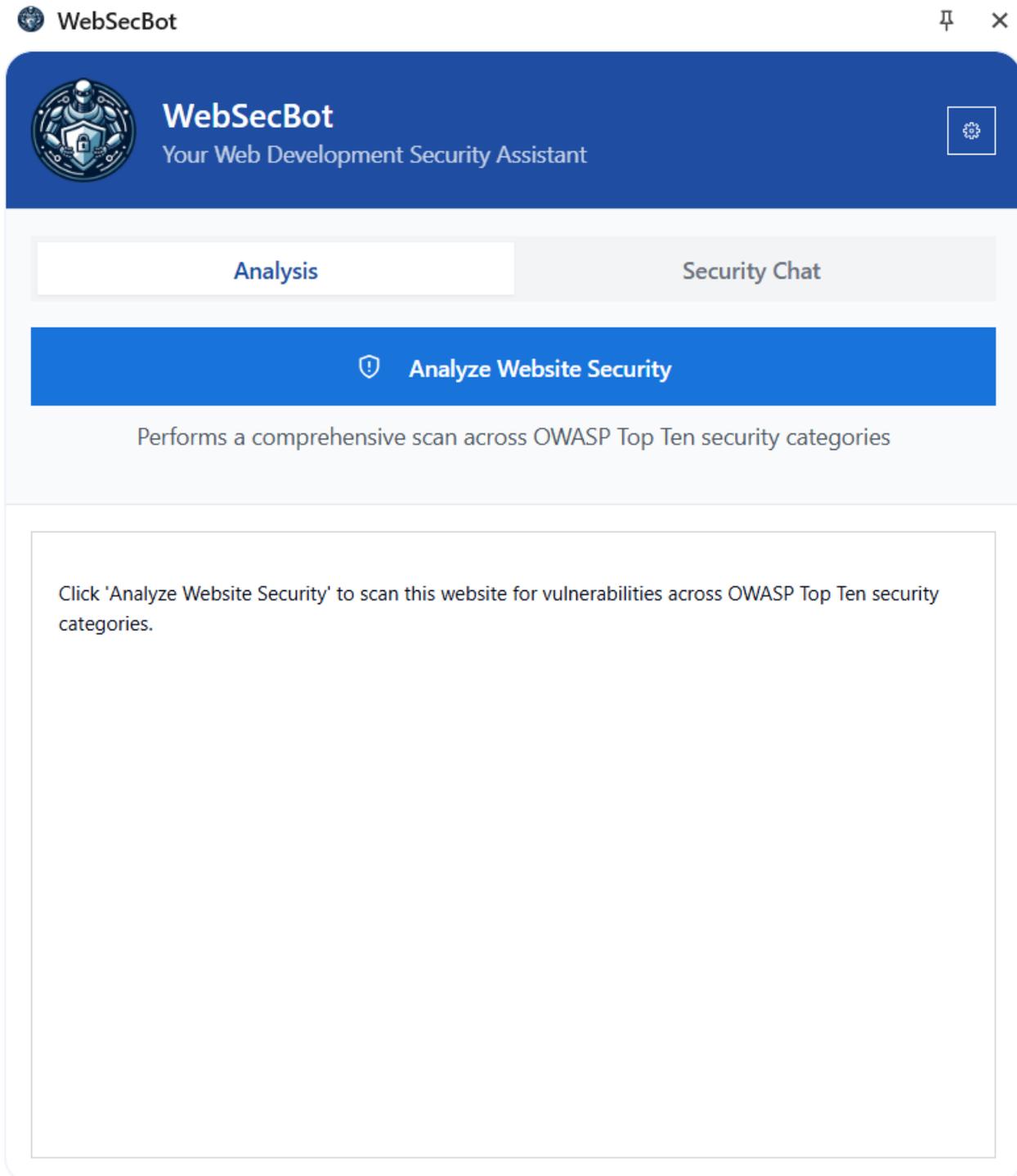


Abbildung 1: Hauptoberfläche des WebSecBots mit den beiden Funktionsbereichen Analysis und Security Chat.

Der Analysis-Tab ermöglicht einen umfassenden Sicherheitsscan der aktuell geöffneten Website und der Security Chat-Tab bietet eine interaktive Beratung zu spezifischen Sicherheitsfragen. Die Benutzeroberfläche ist bewusst einfach gestaltet, um auch für Personen ohne tiefgreifende Sicherheitskenntnisse zugänglich zu sein.

Nach Klick auf "Analyze Website Security" führt WebSecBot einen umfassenden Scan durch, der auf dem OWASP Top Ten Framework basiert – einem international anerkannten Standard für Websicherheit. Der Scan untersucht die Website auf kritische Sicherheitsrisiken wie Broken Access Control, Injection Vulnerabilities, Cryptographic Failures, Security Misconfiguration und Vulnerable & Outdated Components. Während des Scans werden verschiedene Prüfverfahren kombiniert, darunter Header-Analyse, TLS-Konfigurationsprüfungen, Formularanalyse und Tests auf bekannte Schwachstellen.

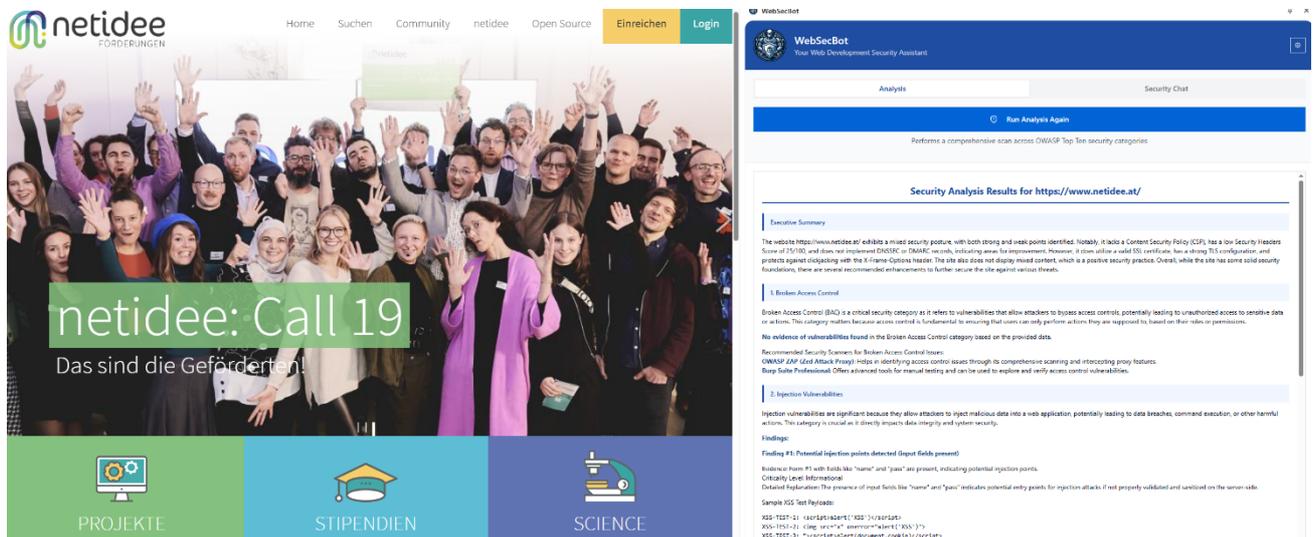


Abbildung 2: Der Analysebereich mit dem Ergebnis einer durchgeführten Analyse.

Nach Abschluss der Analyse generiert WebSecBot einen detaillierten Sicherheitsbericht, der eine Executive Summary mit den wichtigsten Erkenntnissen, die Einteilung der Sicherheitsprobleme in klar definierte Kategorien, konkrete Beweise für jedes identifizierte Problem, eine Bewertung des Schweregrads (Niedrig, Mittel, Hoch), detaillierte Erklärungen der Sicherheitsrisiken in verständlicher Sprache, Beispiel-Payloads für XSS- oder SQL-Injection-Angriffe und Empfehlungen für spezialisierte Sicherheitsscanner umfasst.

Die Analyseergebnisse werden in einer übersichtlichen und leicht verständlichen Form präsentiert, die es auch Nicht-Experten ermöglicht, die Sicherheitslage ihrer Website zu verstehen. Jede identifizierte Sicherheitslücke wird dabei nicht nur benannt, sondern auch mit konkreten Nachweisen belegt und in ihren potenziellen Auswirkungen erklärt. Dieser didaktische Ansatz unterscheidet WebSecBot von vielen anderen Sicherheitstools, die oft nur technische Ergebnisse ohne ausreichende Erklärung liefern.

Security Analysis Results for https://www.netidee.at/

Executive Summary

The website https://www.netidee.at/ exhibits a mixed security posture, with both strong and weak points identified. Notably, it lacks a Content Security Policy (CSP), has a low Security Headers Score of 25/100, and does not implement DNSSEC or DMARC records, indicating areas for improvement. However, it does utilize a valid SSL certificate, has a strong TLS configuration, and protects against clickjacking with the X-Frame-Options header. The site also does not display mixed content, which is a positive security practice. Overall, while the site has some solid security foundations, there are several recommended enhancements to further secure the site against various threats.

1. Broken Access Control

Broken Access Control (BAC) is a critical security category as it refers to vulnerabilities that allow attackers to bypass access controls, potentially leading to unauthorized access to sensitive data or actions. This category matters because access control is fundamental to ensuring that users can only perform actions they are supposed to, based on their roles or permissions.

No evidence of vulnerabilities found in the Broken Access Control category based on the provided data.

Recommended Security Scanners for Broken Access Control Issues:

OWASP ZAP (Zed Attack Proxy): Helps in identifying access control issues through its comprehensive scanning and intercepting proxy features.

Burp Suite Professional: Offers advanced tools for manual testing and can be used to explore and verify access control vulnerabilities.

2. Injection Vulnerabilities

Injection vulnerabilities are significant because they allow attackers to inject malicious data into a web application, potentially leading to data breaches, command execution, or other harmful actions. This category is crucial as it directly impacts data integrity and system security.

Findings:

Finding #1: Potential injection points detected (input fields present)

Evidence: Form #1 with fields like "name" and "pass" are present, indicating potential injection points.

Criticality Level: Informational

Detailed Explanation: The presence of input fields like "name" and "pass" indicates potential entry points for injection attacks if not properly validated and sanitized on the server-side.

Sample XSS Test Payloads:

XSS-TEST-1: `<script>alert('XSS')</script>`

XSS-TEST-2: ``

XSS-TEST-3: `"><script>alert(document.cookie)</script>`

Sample SQL Injection Test Payloads:

SQL-TEST-1: `' OR '1'='1`

SQL-TEST-2: `' UNION SELECT 1,2,3--`

SQL-TEST-3: `admin' --`

Recommendations for Injection Vulnerabilities:

Ensure server-side validation and sanitization of all user inputs.

Implement Content Security Policy (CSP) to define allowed sources of content.

Recommended Security Scanners for Injection Vulnerabilities:

OWASP ZAP (Zed Attack Proxy): Useful for identifying potential injection points and testing them.

Burp Suite Professional: Offers tools for manual testing and verification of injection vulnerabilities.

Abbildung 3: Die Ergebnisse der Sicherheitsanalyse mit einer Executive Summary und den identifizierten Sicherheitsproblemen.

Besonders wertvoll ist der priorisierte Abhilfeplan am Ende des Berichts, der dem Benutzer einen klaren Handlungsleitfaden gibt, welche Sicherheitsprobleme mit welcher Priorität behoben werden sollten.

Prioritized Remediation Plan

High Priority: Finding #3 (Missing Security Headers) - Implement comprehensive security headers to protect against various attacks.

Medium Priority: Finding #1 (Potential Injection Points) - Ensure proper server-side validation and sanitization of user inputs.

Low Priority: Finding #2 (SSL Certificate Expiration) - Renew the SSL certificate before its expiration.

Additional Security Recommendations

Regularly update and patch all software components.

Implement a Web Application Firewall (WAF) to provide an additional layer of security.

Consider using a security scanner regularly to identify and address vulnerabilities.

Recommended Security Scanners

For comprehensive security testing, consider using:

OWASP ZAP (Zed Attack Proxy)

Burp Suite Professional

SQLmap

Nmap with NSE scripts

Dependency-Check

Snyk

Abbildung 4: Der priorisierte Abhilfeplan mit konkreten Handlungsempfehlungen nach Wichtigkeit

Ein Alleinstellungsmerkmal von WebSecBot ist der integrierte Security Chat. Hier können Benutzer spezifische Fragen zu den Scan-Ergebnissen stellen, detaillierte Erklärungen zu Sicherheitskonzepten erhalten, konkrete Code-Beispiele für die Behebung von Sicherheitsproblemen anfordern und Schritt-für-Schritt-Anleitungen zur Implementierung von Sicherheitsmaßnahmen bekommen. Der Security Chat kann den Kontext der Analyse in seine Antworten einbeziehen, was besonders relevante und maßgeschneiderte Beratung ermöglicht. Basierend auf dem priorisierten Abhilfeplan und der interaktiven Beratung kann der Benutzer systematisch die identifizierten Sicherheitsprobleme beheben.

Your Question

Can you create a detailed step-by-step action plan to address these security issues with clear examples?

Expert Response

Response includes analysis context

Introduction to the Action Plan

Given the security analysis of <https://www.netidee.at/>, this action plan will focus on addressing the identified security issues. The primary goals are to enhance the site's security posture by fixing vulnerabilities, improving configurations, and implementing best practices.

Step-by-Step Action Plan

1. Implementing Comprehensive Security Headers

To address Finding #3: Missing Security Headers, follow these steps:

Define a Content Security Policy (CSP): Create a CSP that defines which sources of content are allowed to be executed within a web page. For example:

```
Content-Security-Policy: default-src 'self'; script-src 'self' https://cdn.example.com; object-src 'none'
```

This policy allows scripts to be loaded from the site itself and a specific CDN, while blocking all other sources.

Configure X-XSS-Protection: This header helps prevent cross-site scripting (XSS) attacks by enabling the browser's built-in XSS protection. Set it to 1; mode=block to enable protection and block the page from loading if an XSS attack is detected.

```
X-XSS-Protection: 1; mode=block
```

Implement Referrer-Policy: Control how much referrer information is sent with requests. A good starting point is same-origin, which sends the full URL as the referrer when following same-origin links and no referrer information when following cross-origin links.

```
Referrer-Policy: same-origin
```

Enable Permissions-Policy: This header allows you to control which features can be used by a page, helping to prevent unnecessary permissions and reduce the attack surface. For example, to disable the geolocation feature:

```
Permissions-Policy: geolocation=()
```

Test Security Headers: Use tools like SecurityHeaders.com to test and verify the implementation of these headers.

2. Addressing Potential Injection Points

For Finding #1: Potential injection points detected, take the following actions:

Server-Side Validation and Sanitization: Ensure that all user inputs are validated and sanitized on the server-side. This includes checking the format and content of inputs to prevent malicious data from being processed.

Implement Content Security Policy (CSP): As mentioned earlier, CSP helps define allowed sources of content, which can prevent XSS attacks by limiting where scripts can be loaded from.

Regularly Update and Patch Software: Keep all software components up to date to protect against known vulnerabilities.

Use Parameterized Queries or Prepared Statements: When interacting with databases, use parameterized queries or prepared statements to prevent SQL injection attacks. For example, in Python using SQLite:

```
import sqlite3

# Incorrect - vulnerable to SQL injection
cursor.execute("SELECT * FROM users WHERE name='" + username + "' AND password='" + password + "'")

# Correct - using parameterized query
cursor.execute("SELECT * FROM users WHERE name=? AND password=?", (username, password))
```

3. Renewing the SSL Certificate

To address Finding #2: SSL certificate expiration in less than 60 days:

Renew the SSL Certificate: Obtain a new certificate from a trusted Certificate Authority (CA) before the current one expires.

Automate Certificate Renewal: Consider using tools like Certbot (from Let's Encrypt) to automate the renewal process, ensuring that your site always has a valid SSL certificate.

Test the SSL Configuration: Use tools like SSL Labs by Qualys to test and verify the SSL/TLS configuration after renewing the certificate.

Conclusion

This action plan provides a step-by-step guide to addressing the security issues identified in the analysis of <https://www.netidee.at/>. Implementing comprehensive security headers, addressing potential injection points, and renewing the SSL certificate are crucial steps in enhancing the site's security posture. Regular security testing and maintaining up-to-date software components are also vital for ongoing security. By following these recommendations, the site can significantly reduce its vulnerability to various attacks and improve its overall security.

Abbildung 5: Schritt-für-Schritt-Anleitungen zur Behebung von Sicherheitsproblemen mit konkreten Code-Beispielen

Während klassische Tools lediglich Probleme identifizieren, bietet WebSecBot auch kontextbezogene Erklärungen und praxisnahe Lösungen. Die KI-Komponente ermöglicht es, auf spezifische Fragen einzugehen und individualisierte Ratschläge zu geben. Die Anwendung übersetzt komplexe Sicherheitskonzepte in verständliche Sprache und konkrete Handlungsempfehlungen. Die Benutzeroberfläche ist intuitiv gestaltet und vermeidet Fachjargon, wo immer möglich. WebSecBot ist nicht nur ein Analyse-Tool, sondern auch ein Lernwerkzeug, das Entwickler:innen hilft, bessere Sicherheitspraktiken zu verstehen und anzuwenden. Jede Erklärung dient auch der Bildung und Sensibilisierung. Durch die klare Kennzeichnung von Schweregrad und Dringlichkeit verschiedener Sicherheitsprobleme können Benutzer:innen ihre Ressourcen effektiv einsetzen und die kritischsten Probleme zuerst angehen.

3 Verlauf der Arbeitspakete

3.1 Arbeitspaket 1 - *Detailplanung und Formales am Projektstart*

Arbeitspaket 1 wurde zu Beginn des Projekts mit der Prüfung und Unterzeichnung des Fördervertrags abgeschlossen. Die Projektplanung wurde auf Basis der Vorlage erstellt und an den Fördergeber übermittelt. Im Rahmen dieses Arbeitspakets wurde außerdem die Projektwebsite mit den ersten Inhalten gefüllt und ein initialer Blogbeitrag veröffentlicht, der das Projektteam sowie die Ziele des Projekts vorstellt.

3.2 Arbeitspaket 2 - *Evaluierung lokaler KI-Modelle & Finetuning*

Um die Unabhängigkeit von externen Dienstleistern zu ermöglichen, wurden verschiedene lokale Sprachmodelle evaluiert, um das Modell auszuwählen, das am besten für Web-Sicherheitstests geeignet ist. Im Test standen GPT-4o, Vicuna, Llama 3.1, Llama 3.3 und Mixtral, die hinsichtlich ihres inhärenten Wissens in Bezug auf Sicherheitstests sowie ihrer allgemeinen Leistungsfähigkeit in verschiedenen Szenarien miteinander verglichen wurden.

Nach einer gründlichen Evaluierung erwies sich Llama 3.3 als das am besten geeignete Modell, im Hinblick auf die Verarbeitung sicherheitsrelevanter Informationen und die Effizienz bei der Beantwortung komplexer Fragen. Auf Grundlage dieser Ergebnisse wurde entschieden, Llama 3.3 in den nächsten Schritten als Kernmodell zu verwenden.

Zusätzlich wurden im Zuge dieses Arbeitspakets die relevanten Quellen für das spätere RAG-System recherchiert. Der Fokus lag auf der OWASP-Dokumentation, einschließlich der OWASP Top Ten, des Web Security Testing Guide und der Cheatsheet Series, um sicherzustellen, dass das System über eine fundierte Wissensbasis verfügt.

Außerdem wurde das Llama 3.3 Modell in ein Retrieval-Augmented Generation (RAG)-System integriert. Dabei wurde das zuvor recherchierte sicherheitsrelevante Wissen, insbesondere OWASP-Ressourcen, in das RAG-System eingebunden. Dies ermöglichte es dem Modell, detailliertere und präzisere Vorschläge für sicherheitsrelevante Aufgaben zu machen.

Das RAG-System wurde daraufhin evaluiert, und die Ergebnisse zeigten, dass durch die Einbindung externer Wissensquellen signifikant an Leistung gewann. Diese Verbesserung bestätigte den Nutzen der Kombination aus lokalem Modell und externem Wissenszugang.

3.3 Arbeitspaket 3 - Server-Setup

Für das Server-Setup wurde eine stabile und flexible Infrastruktur entwickelt, die die Ausführung von AnythingLLM und das darin integrierte Retrieval-Augmented Generation (RAG)-System unterstützen sollte. In der Testphase zeigte sich jedoch, dass die serverseitige Ausführung der gesamten Umgebung zu erheblichen Performance-Engpässen führte. Obwohl eine funktionsfähige Server-Implementierung erfolgreich eingerichtet wurde, waren die Antwortzeiten für eine flüssige Benutzerinteraktion zu lang.

Basierend auf diesen Erkenntnissen empfehlen wir stattdessen eine vollständig lokale Einrichtung des gesamten Setups - AnythingLLM, das RAG-System mit den aufbereiteten OWASP-Dokumenten sowie die API-Schnittstelle zu Anbietern wie Groq. Für Nutzer mit leistungsstarken Rechnern besteht zudem die Option, das KI-Modell vollständig lokal auszuführen, was die Unabhängigkeit von externen Diensten weiter erhöht. Diese lokale Konfiguration bietet zusätzlich den Vorteil, dass sensible Webanalyse-Daten die lokale Umgebung nicht verlassen müssen.

Die Dokumentation wurde entsprechend erweitert und enthält nun detaillierte Anleitungen zur lokalen Installation und Konfiguration aller notwendigen Komponenten, einschließlich des RAG-Systems, der API-Schnittstellen zu verschiedenen Modellanbietern und optional der lokalen Modellausführung.

3.4 Arbeitspaket 4 – Browser-Addon & LLM-Schnittstelle

Im Rahmen dieses Arbeitspakets lag der Schwerpunkt auf der Entwicklung des WebSecBot als Browser-Addon, das Entwickler:innen dabei unterstützt, Webseiten auf potenzielle Sicherheitslücken zu analysieren. Das Addon wurde erfolgreich in React implementiert und bietet eine intuitive Benutzeroberfläche mit zwei Hauptfunktionen: einer Sicherheitsanalyse-Komponente und einem interaktiven Security-Chat.

Die Sicherheitsanalyse-Komponente erlaubt es Nutzer:innen, die aktuell geöffnete Webseite automatisch zu scannen und liefert strukturierte Ergebnisse zu möglichen Schwachstellen basierend auf OWASP-Standards. Hierbei werden DOM-Elemente, Header-Informationen und

andere sicherheitsrelevante Aspekte der Webseite untersucht und potenzielle Risiken identifiziert.

Der interaktive Security-Chat ermöglicht es Entwickler:innen, detaillierte Fragen zu erkannten Sicherheitsproblemen zu stellen und spezifische Anleitungen zur Behebung zu erhalten. Diese Komponente wurde zunächst mit der OpenAI API entwickelt, um die Funktionalität zu validieren, wurde jedoch im weiteren Verlauf an die selbst gehostete AnythingLLM-Instanz angebunden.

Eine besondere Herausforderung stellte die Integration zwischen dem Browser-Addon und der LLM-Schnittstelle dar. Hier wurden verschiedene Ansätze evaluiert, um eine zuverlässige und effiziente Kommunikation zu gewährleisten. Die entwickelte Lösung ermöglicht es, den Kontext der aktuellen Webseite zusammen mit den Analyseergebnissen an das LLM zu übermitteln, sodass kontextbezogene und relevante Sicherheitsempfehlungen generiert werden können.

3.5 Arbeitspaket 5 – *Prompt-Engineering für interaktive Websecurity-Anwendungen*

Im Rahmen des Arbeitspakets 5 wurde ein umfassendes Prompt-Engineering für den WebSecBot durchgeführt, um die KI-Komponente optimal für interaktive Websicherheitsanwendungen auszurichten. Zunächst erfolgte eine gründliche Recherche von vorhandenen Texten und Beispielen aus dem Bereich Websicherheit, die als Grundlage für die Entwicklung effektiver System-Prompts dienten.

Das Team erstellte präzise System-Prompts, die das LLM befähigten, Websicherheitsprüfungen basierend auf standardisierten Sicherheitsrichtlinien wie den OWASP-Guidelines durchzuführen. Dabei wurden verschiedene Prompt-Strategien entwickelt und getestet, um sicherzustellen, dass das Modell gezielt auf sicherheitsrelevante Eingaben reagiert und dabei potentielle Gefahren korrekt identifiziert.

Die finalen System-Prompts wurden in verschiedenen Szenarien evaluiert und iterativ verfeinert, bis sie zuverlässig relevante und umsetzbare Sicherheitsempfehlungen lieferten. Der Fokus lag dabei auf der Balance zwischen technischer Tiefe und Verständlichkeit, um sowohl erfahrenen Sicherheitsexperten als auch Entwicklern mit begrenztem Sicherheitswissen nützliche Hinweise zu geben.

Die optimierten Prompts wurden schließlich erfolgreich in das Browser-Addon integriert, wodurch eine nahtlose Nutzererfahrung in Echtzeitanwendungen gewährleistet wurde. Das abgeschlossene Prompt-Engineering bildete eine solide Grundlage für die anschließenden Test- und Evaluierungsphasen des Projekts.

3.6 Arbeitspaket 6 – Testing und Evaluierung

Im Rahmen des Arbeitspakets 6 wurde eine umfassende Evaluierung des WebSecBot-Systems durchgeführt, wobei sowohl die funktionalen Aspekte des Browser-Addons als auch die inhaltliche Qualität der Sicherheitsanalysen im Fokus standen. Das Testing erfolgte in realen Szenarien sowie in kontrollierten Testumgebungen, um die Effektivität des Systems unter verschiedenen Bedingungen zu bewerten.

Als Testumgebung diente primär der OWASP Juice Shop, eine bewusst verwundbare Webanwendung, die eine Vielzahl bekannter Sicherheitslücken enthält. Diese Plattform ermöglichte es, die Erkennungsfähigkeiten des WebSecBots systematisch zu evaluieren und mit einem bekannten Referenzstandard zu vergleichen. Zusätzlich wurden ausgewählte produktive Websites unterschiedlicher Komplexität in die Tests einbezogen, um die Leistung des Tools unter realen Bedingungen zu überprüfen. Die vom System generierten Empfehlungen wurden von den Testern als präzise und praxisnah bewertet, wobei besonders die Fähigkeit des Systems, kontextbezogene Erläuterungen zu liefern, positiv hervorgehoben wurde.

3.7 Arbeitspaket 7 – Dokumentation und Formales am Projektende

Dieses Arbeitspaket umfasste die abschließende Dokumentation des Projekts, einschließlich der Erstellung des Projektendberichts, der Zusammenfassung sowie der Anwender:innen- und Entwickler:innen-Dokumentation. Die Projektwebsite wurde aktualisiert und alle Ergebnisse unter Angabe der Lizenzen der Öffentlichkeit zur Verfügung gestellt.

4 Umsetzung Förderauflagen

Das Projekt hatte keine speziellen Förderauflagen.

5 Liste Projektergebnisse

1	Projektzwischenbericht	CC BY-SA 4.0	https://www.netidee.at/websecbot
2	Projektendbericht	CC BY-SA 4.0	https://www.netidee.at/websecbot
3	Entwickler_innen-DOKUMENTATION	CC BY-SA 4.0	https://www.netidee.at/websecbot

4	Anwender_innen-DOKUMENTATION	CC BY-SA 4.0	https://www.netidee.at/websecbot
5	Veröffentlichungsfähiger Einseiter / Zusammenfassung	CC BY-SA 4.0	https://www.netidee.at/websecbot
6	Dokumentation Externkommunikation zur Erreichung Sichtbarkeit /Nachhaltigkeit (Teil des Endberichtes)	CC BY-SA 4.0	In dieses Dokument integriert
7	Browser-Addon	MIT	https://www.github.com/sschritt/WebSecBot https://www.netidee.at/websecbot
8	OWASP-basierte Wissensdatenbank für RAG	MIT	https://www.github.com/sschritt/WebSecBot https://www.netidee.at/websecbot

6 Verwertung der Projektergebnisse in der Praxis

Das WebSecBot-Addon und die begleitende OWASP-basierte Wissensdatenbank werden als Open-Source-Lösungen zu Verfügung gestellt. Entwickler:innen können das Addon installieren und einfach auf die eigenen Webapplikatione anwenden. Sie werden dabei unterstützt, Sicherheitslücken frühzeitig zu erkennen und beheben. Um weiteres Feedback zur Praxistauglichkeit zu erhalten, haben wir die erste Version des Tools dem Pentesting-Team von SBA Research präsentiert und darum gebeten, es bei passenden Projekten einzusetzen und zu evaluieren. Darüber hinaus kommt der WebSecBot bereits in ersten Softwareprojekten und Schulungsszenarien zum Einsatz. Durch die modulare Architektur ist eine einfache Integration in bestehende Entwicklungsumgebungen möglich. Eine Weiterentwicklung der Konzepte ist im Rahmen eines größeren Forschungsprojekts ebenfalls geplant.

7 Öffentlichkeitsarbeit/ Vernetzung

Für das Projektende ist eine Präsentation des entwickelten Add-ons im Rahmen eines Security-Meetups bei SBA Research geplant. Diese Veranstaltung bietet eine ideale Gelegenheit, das Add-

on einer breiteren Fachöffentlichkeit vorzustellen und den Austausch mit anderen Expert:innen aus der IT-Sicherheitsbranche zu fördern.

8 Eigene Projektwebsite

Es wurde keine eigene Projektwebseite aufgesetzt. Der Code wird über Github zur Verfügung gestellt, worüber auch in Zukunft die Weiterentwicklung und Kommunikation (Issues, Discussions) stattfinden soll.

9 Geplante Aktivitäten nach netidee-Projektende

Nach Projektende sind mehrere Aktivitäten geplant, um die Nachhaltigkeit und Weiterentwicklung von WebSecBot sicherzustellen. Zum einen ist ein umfangreicher Forschungsantrag im Bereich der Automatisierung von Cybersecurity in Vorbereitung, bei dem die im Rahmen des WebSecBot-Projekts erzielten Ergebnisse als wichtige Vorarbeiten dienen. Zum anderen soll der WebSecBot weiterentwickelt und gezielt in der anvisierten Zielgruppe verbreitet und vorgestellt werden.

10 Anregungen für Weiterentwicklungen durch Dritte

WebSecBot bietet zahlreiche Möglichkeiten für Weiterentwicklungen durch Dritte:

- Integration weiterer Sicherheits-Scanning-Tools und -Techniken
- Erweiterung für zusätzliche Browser wie Firefox, Safari oder Edge
- Implementierung von Custom-Checks für unternehmensspezifische Sicherheitsrichtlinien
- Verbesserung der LLM-Komponente durch Training mit spezialisierten Datensätzen
- Entwicklung einer IDE-Integration für beliebte Entwicklungsumgebungen
- Erweiterung um API-Sicherheitstests für Backend-Services
- Unterstützung für CI/CD-Pipelines zur automatisierten Sicherheitsprüfung
- Aufbau einer Community-Plattform zum Austausch von Custom-Prompts und Sicherheitsprüfungen

Die umfangreiche OWASP-basierte Wissensdatenbank, die wir erstellt haben, kann als Grundlage für weitere Sicherheitstools dienen. Entwickler:innen können diese Daten für eigene RAG-Systeme nutzen oder sie für spezifische Sicherheitsdomänen erweitern.

Die modulare Architektur des WebSecBot ermöglicht es Entwickler:innen, an einzelnen Komponenten zu arbeiten, ohne das gesamte System verstehen zu müssen, was die Einstiegshürde für Beiträge senkt.