

ABSCHLUSSPROJEKT SECURITY BY DESIGN DOKUMENTATION



Electricity Provider

Teilnehmer in dem Projekt sind

*Kevin Wagner / Luis Eckert / Kai Pistol /
Cynthia Winkler*

Betreuerender Dozent
Herr Schneider

Zusammenfassung

In der folgenden Dokumentation geht es um die Entwicklung eines Stromanbieterportals für einen fiktiven Stromanbieter. Die Webanwendung baut auf dem Django-Framework auf und soll auf allen Geräten mit einem modernen Webbrowser aufrufbar sein. Des Weiteren soll eine Schnittstelle zu einem Messstellenbetreiber implementiert werden, um Messdaten für das Portal abrufen zu können.

Inhaltsverzeichnis

1 Einführung	2
1.1 Aufgabenstellung	2
1.2 Konkrete Abgaben	3
2 Projektplanung	4
2.1 Projektziel	4
2.2 Projektrollen	4
2.3 Projektplanung und Vorgehen	4
3 Use Cases	6
4 Anforderungen	7
5 Bedrohungsanalyse - Threat Modeling	8
5.1 Architekturdiagramm	8
5.2 Gesetzliche Regularien	11
5.3 Schutzobjekte - Assets	12
5.4 Risikoregister	16
6 Gewählte Technologien	17
6.1 Docker	17
6.2 Python-Django (Backend / Frontend)	17
6.3 PostgreSQL-Datenbank	17
7 Komponente und Module	18
8 Testplan	19
8.1 Einleitung	19
8.1.1 Scope - Umfang	19
8.1.2 Qualitätsziele	19
8.1.3 Rollen und Verantwortungen	19
8.2 Test Methodik	20
8.3 Ergebnisse der Prüfung	21
8.4 Notwendige Anwendungen und Programme	22
8.4.1 Testing Tool	22
8.4.2 Testing Umgebung	22
9 Projektumsetzung	23
10 Quellen	25

1 Einführung

1.1 Aufgabenstellung

1. Identifizieren, präzisieren und dokumentieren Sie zunächst die in diesem Geschäftsumfeld (business context) zu erwartenden zu schützenden Objekte, deren Schutzziele und Sicherheitsanforderungen. Berücksichtigen Sie dabei auch Ihnen bekannte rechtliche und regulatorische Vorschriften (regulatory context). Entwerfen Sie dann eine geeignete Architektur und wählen Sie eine geeignete Technologie bzw. Plattform zur Realisierung der Anwendung (technology context). Überprüfen und ergänzen Sie zu schützende Objekte, Schutzziele und Sicherheitsanforderungen mittels einer architektonischen Bedrohungsanalyse (threat modeling).
2. Erstellen Sie ausgehend von den Ergebnissen der architektonischen Bedrohungsanalyse ein Register für die in Ihrem Projekt identifizierten Sicherheitsrisiken. Bestimmen Sie dazu Eintrittswahrscheinlichkeit und mögliche Schäden der identifizierten Bedrohungen und das daraus resultierende Risiko. Entscheiden Sie für jedes Risiko wie es behandelt werden soll und im Falle einer Reduzierung, welche Maßnahmen dafür durchgeführt werden und wie diese überprüft werden sollen. Dokumentieren Sie Ihre Ergebnisse im Risikoregister Ihres Projektes.
3. Implementieren Sie die Anwendung unter Berücksichtigung der identifizierten Anforderungen, Risiken, den zur Risikobehandlung festgelegten Maßnahmen und allgemeinen Grundlagen sicheren Programmierens.
4. Testen und überprüfen Sie die Anwendung hinsichtlich der Grundfunktionalität, den identifizierten Sicherheitsanforderungen und den zur Risikobehandlung implementierten Maßnahmen. Erstellen Sie dazu einen Testplan, nutzen Sie geeignete Testmethoden und - werkzeuge und dokumentieren Sie die Testergebnisse.
5. Führen Sie die Schritte 1.-4. im Entwicklungs- und Bereitstellungsprozess in geeigneter Weise verzahnt und wiederholt durch.

1.2 Konkrete Abgaben

25.10.2023:

- Beschreibung der zu schützenden Objekte (Assets), deren Schutzziele und Sicherheitsanforderungen der Anwendung
- Architekturbeschreibung, -diagramm
- Wahl der Technologie zur Implementierung, Plattform, Komponenten
- Risikoregister

05.01.2024:

- Präsentation - freies Format + Dateien bzw. Angabe GitHub-Repository
- Beschreibung der Modulstruktur, Build-Prozess und -Einstellungen
- Source Code (inklusive Build-Skript)
- Testplan
- Testergebnisse

2 Projektplanung

2.1 Projektziel

Das Ziel des Projektes ist das Schaffen einer “sicheren“ Webanwendung, welche dem Zweck dient Kunden ihre spezifischen Messdaten eines Stromzählers zu übermitteln. Diese Messdaten werden durch einen externen Messstellenbetreiber erhoben und dem Webportal durch eine API-Schnittstelle zur Verfügung gestellt. Dem Kunden des Stromanbieters soll es möglich sein, seine persönlichen Messdaten einzusehen, sowie seine Daten anzupassen oder neue Verträge abzuschließen.

2.2 Projektrollen

Für das vorliegende Projekt wurde eine Aufteilung in diverse Rollen vorgenommen. Diese Aufteilung sieht wie folgt aus:

- Kevin Wagner: Scrum Master, Entwickler
- Luis Eckert: Product Owner, Entwickler
- Kai Pistol: Entwickler
- Cynthia Winkler: Dokumentarin, Entwicklerin

2.3 Projektplanung und Vorgehen

Vorab wurde im Team ein Projektplan definiert, sodass es einen Rahmen für den weiteren Projektlauf gab an dem man sich orientieren konnte. Dieser Projektplan wurde im Nachhinein noch einmal verändert, da der zeitliche Rahmen nicht eingehalten werden konnte. Die wichtigsten Punkte sind im folgenden Abschnitt aufgeführt:

1. Erstellung eines GitHub-Repositories, eines GitHub-Projekts und eine Grundlage für die Projektdokumentation
2. Definieren von Use Cases
3. Zusammenragen der Anforderungen und Übertragung als Issues auf GitHub
4. Erstellung einer ersten Version eines Architekturdiagramms und möglichen notwendigen Komponenten
5. Aufbauend die Identifikation der zu schützenden Objekte (Assets), deren Schutzziele und Sicherheitsanforderungen (Kapitel 5.3)
6. Anlegung eines Risikoregisters und Bestimmung der Security Controls
7. Identifikation der für uns relevanten regulatorischen Vorschriften
8. Festlegung der genutzten Technologien
9. Übergang zur Coding-Phase
10. Erstellung eines Testplans

11. Durchgehende Bearbeitung bzw. wenn notwendig erfolgte bei:

- (a) Dokumentation
- (b) Architekturdiagramm
- (c) Assets-Tabelle
- (d) Risikoregister

3 Use Cases

Zu Beginn wurden die nachfolgenden Use Cases erarbeitet. Diese wurden bei der Erarbeitung der funktionellen und nicht funktionellen Anforderungen überarbeitet, teilweise gelöscht oder beibehalten. Diese sind im folgenden Abschnitt aufgeführt:

Besucher:

1. Als Besucher will ich Tarifinformationen des Stromanbieters einsehen können.
2. Als Besucher will ich Informationen über den Stromanbieter einsehen können.
3. Als Besucher will ich mich bei dem Stromanbieter registrieren können.

Kunde:

1. Als Kunde will ich meine Vertragsdaten einsehen können.
2. Als Kunde will ich meine Rechnungen einsehen können.
3. Als Kunde will ich meine persönlichen Daten anpassen können.
4. Als Kunde will ich meine Rechnungsdaten anpassen können.
5. Als Kunde will ich meine Messdaten einsehen können.
6. Als Kunde will ich die Höhe meiner monatlichen Abschläge ändern können.
7. Als Kunde will ich eine berechnete Kostenprognose basierend auf den aktuellen Messdaten.
8. Als Kunde will ich digital einen Wohnortswechsel angeben können.
9. Als Kunde will ich einen monatlichen Stromverbrauch-Schwellwert einrichten können, der mich benachrichtigt, sobald dieser überschritten wird.

4 Anforderungen

Die Anforderungen an jedes Projekt lassen sich im Normalfall in **funktionale Anforderungen** und **nichtfunktionale Anforderungen** unterteilen. Der Übersicht halber wurden zunächst einmal für jede Anforderung ein Issue in GitHub geöffnet. Zu diesen konnten Labels hinzugefügt werden, wodurch man sich zum Beispiel alle funktionalen Anforderungen anzeigen lassen konnte. Diese Anforderungen lassen sich wie folgt zusammenfassen:

Funktionale Anforderungen

- Besucher sollen können:
 - Informationen über Tarife und den Stromanbieter einsehen
 - Sich bei dem Stromanbieter als Kunde registrieren können
- Kunden sollen können:
 - Daten einsehen:
 - * Personenspezifische Daten
 - * Vertragsdaten
 - * Messdaten
 - Daten ändern:
 - * E-Mail-Adresse
 - * Passwort
 - * Rechnungsadresse
 - einen neuen Vertrag anlegen
 - ihren Vertrag kündigen

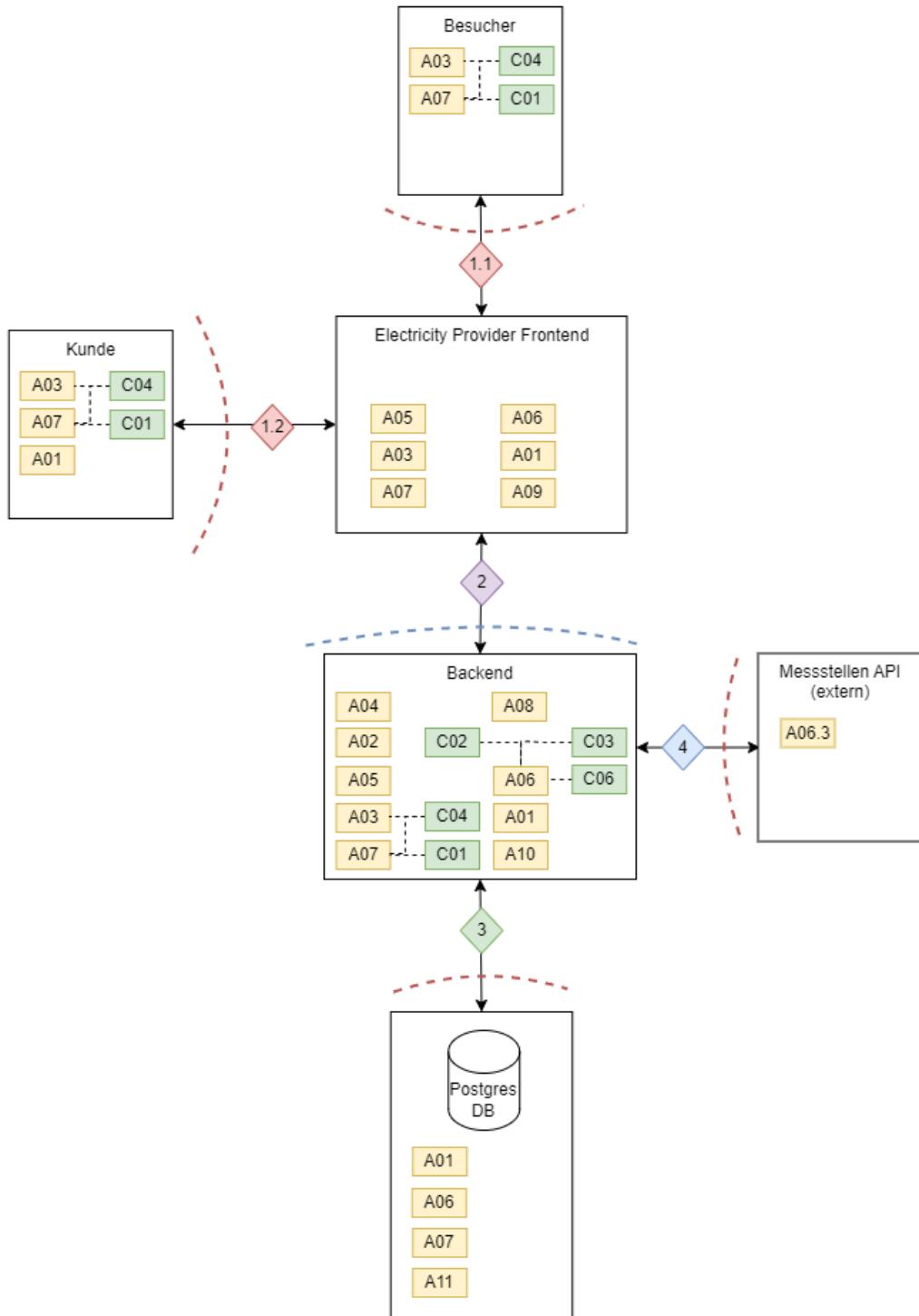
Nicht-Funktionale Anforderungen

- Kunden können nur auf eigene Daten zugreifen
- die Primärschlüssel der Tabellen für personenbezogene Daten verwendet ein nicht-deterministisches Format (UUID4)
- Nutzung eines sicheren API-Secrets
- der Zugriff auf den API-Endpunkt erfolgt nur nach erfolgreicher Authentifizierung
- die Interaktion mit der Datenbank wird über ein ORM-Framework (Object-Relation-Mapping) realisiert, um statischen SQL Code zu vermeiden
- die Weboberfläche ist responsive
- Erstellen Unit-/Integrationstests
- „Secure by Default“
- der User erhält Feedback von der UI über Handlungen im System (z.B. mehrere fehlgeschlagene Loginversuche)

5 Bedrohungsanalyse - Threat Modeling

5.1 Architekturdiagramm

Nach mehrfacher Bearbeitung des Architekturdiagramms sieht dieses wie folgt aus:



Die in dem Diagramm bestimmten Schutzobjekte (Assets) und Sicherheitskontrollen (Security Controls) sind folgende:

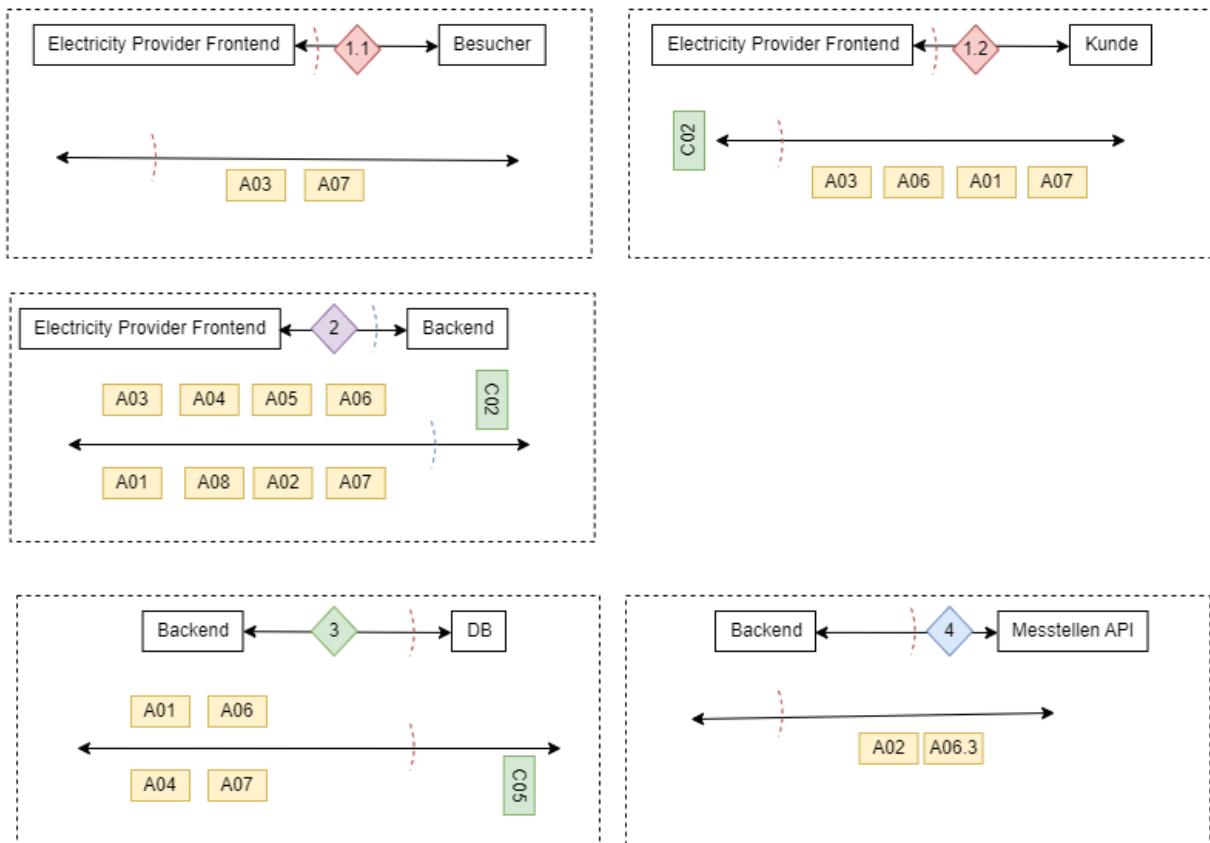
Assets		Security Controls	
ID	Description	ID	Description
A01	Zugangsdaten Kunde	C01	CSRF-Token
A02	Messstellen API Secret	C02	Input Validation
A03	Cookies	C03	Hashing und Salten von PW
A04	DB Secret	C04	Session Security
A05	Tarifinformationen	C05	Authentifizierung
A06.0	Kundendaten (Vertragsdaten)	C06	Logging
A06.1	Kundendaten (Person)		
A06.2	Kundendaten (Bankdaten)		
A06.3	Kundendaten (Messdaten)		
A06.4	Kundendaten (Rechnungen)		
A07	Session-Token		
A08	Zugangsdaten DB Admin		
A09	Frontend Server		
A10	Backend Server		
A11	Datenbank Server		

Die Komponenten „**Besucher**“ und „**Kunden**“ haben einen ähnlichen Inhalt, werden aber getrennt betrachtet. Dies liegt daran, dass nicht jede Person, die die Webseite aufruft, auch automatisch ein Kunde ist. Jede Person kann die Webseite aufrufen und grundlegende Daten zu z.B. Tarifen abrufen. Kunden wiederum haben ein Konto, mit welchem sie sich anmelden können. Hierzu werden vom Kunden Daten erhoben, um ihm personen-spezifische Daten anzeigen zu können.

Die **Webseite** handelt die verschiedenen Daten, die entweder erhoben werden oder von der externen Messstellen API erhalten werden. Zum Umgang dieser Daten zählt auch das Abspeichern in die Postgress **Datenbank**.

Personenspezifischen Messdaten erhalten wir durch eine externe **Messstellen-API**.

Die Abhängigkeiten bzw. der Datenfluss ist wie folgt bestimmt:



5.2 Gesetzliche Regularien

Stromanbieter gehören zu den kritischen Infrastrukturen (KRITIS). Daraus leiten sich verschiedene Regularien ab, die wir zwingend erfüllen müssen. Die Überwachung der Umsetzung und Instandhaltung übernimmt das BSI. Die Grundinformationen, an die sich KRITIS Unternehmen halten müssen, sind im IT-Sicherheitsgesetz 2.0 zu finden.

Für uns relevante gesetzliche Vorgaben sind unter anderem:

- **DSGVO / BDSG (Datenschutz-Grundverordnung):**

- Die DSGVO regelt den Datenschutz innerhalb der EU. Sie legt fest, wie personenbezogene Daten erhoben, verarbeitet, gespeichert und gelöscht werden dürfen.
- Als Stromanbieter arbeiten wir mit personenbezogenen Daten. Wir müssen sicherstellen, dass wir die Datenschutzbestimmungen der DSGVO einhalten. Dazu zählt die Erhebung von Einwilligungen, den Schutz der Daten und Möglichkeit für Kunden, ihre Daten einsehen, korrigieren oder löschen zu können.

- **PCIDSS (Payment Card Industry Data Security Standard):**

- PCIDSS ist ein von Kreditkartenorganisationen festgelegter Sicherheitsstandard, welcher den Schutz von Kreditkartendaten gewährleisten soll. Er beinhaltet Anforderungen an die Systeme, Prozesse und Richtlinien von Unternehmen, welche Kreditkartentransaktionen durchführen oder verarbeiten.
- Da wir in unseren Verträgen Kreditkartendaten erheben und speichern, müssen wir sicherstellen, dass unser System den Anforderungen der PCIDSS entsprechen. Zu diesen Anforderungen gehören zum Beispiel die regelmäßige Überprüfung unserer Systeme, die Einhaltung von Best Practices oder die regelmäßige Schulung unserer Mitarbeiter.

- **EnWG (Energiewirtschaftsgesetz):**

- Das EnWG ist ein nationales Gesetz, das die Organisation und den Betrieb des Energiemarktes regelt. Durch dieses werden die rechtlichen Rahmenbedingungen für den Zugang zum Netz, des Netzbetriebes, die Energieerzeugung und den Energievertrieb festgelegt.
- Wir unterliegen den Vorgaben des EnWG in Bezug auf den Netzbetrieb, den Zugang zum Netz und den Energievertrieb. Wir müssen die gesetzlichen Anforderungen des EnWG erfüllen, insbesondere die Transparenz, Diskriminierungsfreiheit und den fairen Wettbewerb.

5.3 Schutzobjekte - Assets

Schutzobjekt	Schutzziele	Sicherheitsanforderungen
Persönliche Kundendaten	Vertraulichkeit Integrität Verfügbarkeit Authentizität Identität	<ul style="list-style-type: none"> • Zugriffskontrolle: Die Daten können nur vom Kunden und autorisiertem Personal bearbeitet werden • Verschlüsselung: Datenverschlüsselung während der Übertragung & Ruhezustand • Datensicherung & Wiederherstellung: regelmäßige Sicherung zur Vorbeugung von Datenverlusten, Notfallwiederherstellungspläne • Überwachung, Protokollierung: Überwachungssysteme, Protokollierung zur Erkennung nicht autorisierter Aktivitäten • DPIA - Datenschutz- Folgeabschätzung • Transparenz, Informationspflichten: Betroffene müssen über Erhebung, Verarbeitung der Daten informiert werden, mit Angabe des Verwendungszweckes (EnWG)

Bankdaten	Vertraulichkeit Integrität Verfügbarkeit Authentizität Verbindlichkeit	<ul style="list-style-type: none"> • Zugriffskontrolle: Die Daten können nur vom Kunden und autorisiertem Personal bearbeitet werden • Verschlüsselung: Datenverschlüsselung während der Übertragung & Ruhezustand • Datensicherung & Wiederherstellung: regelmäßige Sicherung zur Vorbeugung von Datenverlusten, Notfallwiederherstellungspläne • Überwachung, Protokollierung: Überwachungssysteme, Protokollierung zur Erkennung nicht autorisierter Aktivitäten • DPIA - Datenschutz-Folgeabschätzung
Vertragsdaten	Vertraulichkeit Integrität Verfügbarkeit Authentizität Verbindlichkeit Identität	<ul style="list-style-type: none"> • Zugriffskontrolle: Die Daten können nur von autorisiertem Personal bearbeitet und vom Kunden eingesehen werden • Verschlüsselung: Datenverschlüsselung während der Übertragung & Ruhezustand • Datensicherung & Wiederherstellung: regelmäßige Sicherung zur Vorbeugung von Datenverlusten, Notfallwiederherstellungspläne • Überwachung, Protokollierung: Überwachungssysteme, Protokollierung zur Erkennung nicht autorisierter Aktivitäten • DPIA - Datenschutz-Folgeabschätzung

Messdaten	Vertraulichkeit Integrität Verfügbarkeit Identität	<ul style="list-style-type: none"> • Datenschutz: Schutz der Messdaten vor unbefugtem Zugriff, Diebstahl; Zugriffskontrollen (Daten können nur von autorisiertem Personal & Kunden eingesehen werden) • Datensparsamkeit: Erfassung, Speicherung nur von notwendigen Messdaten • Transparenz, Informationspflichten: Betroffene müssen über Erhebung, Verarbeitung der Daten informiert werden, mit Angabe des Verwendungszweckes (EnWG) • Rechte der Betroffene (DSGVO): Recht auf Auskunft, Recht auf Berichtigung und Löschung, Recht auf Verarbeitungseinschränkung der Daten, Recht auf Widerspruch der Datenverarbeitung und Recht auf Datenübertragbarkeit • DPIA - Datenschutz-Folgeabschätzung
Systemchnittstellen	Vertraulichkeit Integrität Verfügbarkeit Widerstandsfähigkeit	<ul style="list-style-type: none"> • Zugriffskontrolle • Gesicherte Kanäle: Zugriff nur über verschlüsselte Kanäle • DB, Mitarbeiterportal ist nicht nach außen verfügbar

Produktivsystem	Vertraulichkeit Integrität Verfügbarkeit Verbindlichkeit Widerstandsfähigkeit Autorisierung	<ul style="list-style-type: none"> • Zugriffskontrolle/Rollenverteilung • Validierung/Bereinigung User Eingaben • Schutz vor DDOS-Angriffen
Cookies	Integrität Authentizität	<ul style="list-style-type: none"> • Cookies dürfen nicht von Angreifern genutzt werden können, um z.B. Session Hijacking oder CSRF zu betreiben

5.4 Risikoregister

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R01	DDoS-Angriff	Denial of Service	[Hoch]	Hoch	[Hoch]	[Transferieren]
Beschreibung						
Der Angreifer schickt sehr viele Anfragen an den Server, um dessen Verfügbarkeit einzuschränken.						
Anforderungen						
Der Server darf bei vielen Anfragen keine Performance verlieren oder andersweitig eingeschränkt sein.						
Maßnahmen				Überprüfung	Test-ID	
Der Server darf nur eine begrenzte Anzahl an Anfragen akzeptieren und bearbeiten. Verwendung einer WAF oder eines Load Balancers mithilfe von CloudFlare.				Last Test Design Review	Out Of Scope	

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R02	SQL-Injection	Tampering, Information Discloser, Denial of Service, Elevation of Privilege	[Sehr hoch]	[Sehr hoch]	[Sehr hoch]	[Reduzieren]
Beschreibung						
In Eingabefelder werden SQL-Statements geschrieben, welche unerlaubte Anfragen an die Datenbank schickt, mit dem Ziel, Informationen aus der DB zu erhalten oder deren Integrität zu kompromittieren						
Anforderungen						
Durch Eingabefelder darf keiner ausführbarer SQL-Code in die DB gelangen.						
Maßnahmen				Überprüfung	Test-ID	
Alle Eingaben werden geprüft und bereinigt, dass sie nur als Text gesehen werde und niemals dazu in der Lage sind SQL-Code ausgeführt zu werden Benutzung eines ORM-Frameworks (Django).				[Automatisierter Test] [Pentest]	T-R02	

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R03	Cross-Site-Scripting-Angriff (XSS)	Spoofing, Tampering, Information Discloser	[Hoch]	[Hoch]	[Hoch]	[Reduzieren]
Beschreibung						
Cross-Site-Scripting-Angriffe zielen darauf ab, dass in Eingabefeldern oder Kommentarfeldern schädlicher Java-Script-Code geschrieben werden kann. Bei anderen Usern wird dieser Code dann ausgeführt, sollten sie auf die Seite gehen, auf welchem sich der Code befindet (z.B. Kommentare lesen).						
Anforderungen						
In Eingabefeldern darf kein Code geschrieben werden oder stehen.						
Maßnahmen				Überprüfung	Test-ID	
Alle Eingaben müssen geprüft und bereinigt werden, dass sie nur als Text gesehen werde und niemals dazu in der Lage sind Code auszuführen.				[Pentest] [Manueller Test]	T-R03	

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R04	Man-in-the-Middle-Angriff (MitM)	Spoofing, Denial of Service	[Hoch]	[Hoch]	[Hoch]	[Reduzieren]
Beschreibung						
Beim MitM-Angriff fängt der Angreifer User-Anfragen ab und kann dessen Daten bzw. Inhalte einsehen und verändern. Diese kann er wiederum zurück an die eigentliche Seite schicken, um es so aussehen zu lassen, als wäre nichts passiert. Außerdem sind die Daten extrem komromittiert, sollten sie in unverschlüsselter Form vorliegen.						
Anforderungen						
Alle Anfragen müssen grundsätzlich über verschlüsselte Kanäle laufen. Alle Anfragen müssen authentifiziert sein, um zu überprüfen, dass es sich um den richtigen User handelt.						
Maßnahmen Keine Anfragen dürfen unverschlüsselt sein. Hierzu wird HTTPS genutzt und HTTP Anfragen werden abgelehnt. Überprüfung, dass es sich um den richtigen User handelt.				[Automatisierter Test]	Überprüfung	Test-ID T-R04

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R05	Automatisierte Angriffe		[Hoch]	[Mittel]	[Mittel]	[Reduzieren]
Beschreibung						
Automatisierte Angriffe könnte beinhalten, dass Bots Registrierungs oder Anmelde-Formulare ausfüllen und damit das System zumüllen. Da eine Registrierung mit einem initialen Mitarbeiter verbunden ist, können hier erhöhte Kosten anfallen.						
Anforderungen						
Formulare dürfen nicht von Bots ausgefüllt bzw. abgeschickt werden.						
Maßnahmen CAPTCHA-Überprüfung notwendig zum abschicken des Formulares				[Manueller Test]	Überprüfung	Test-ID T-R05

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R06	Cross-Site-Referenz-Forgery (CSRF)	Spoofing, Repudation, Information Discloser	[Hoch]	[Hoch]	[Hoch]	[Reduzieren]
Beschreibung						
Beim CSRF-Angriff will der Angreifer den User dazu zu verleiten Aktionen auszuführen, die der User eigentlich gar nicht ausführen möchte.						
Anforderungen						
Es dürfen keine CSRF-Angriffe möglich sein bzw. dürfen sie nur geringe Auswirkungen auf das laufende System haben. Es muss sichergestellt werden, dass Anfragen tatsächlich vom User stammen.						
Maßnahmen Implementierung eines Anti-CSRF-Token: Hierbei handelt es sich um einen eindeutigen Token, der in jedes Formular eingebettet wird. Beim Absenden oder Anfragen wird dieses verglichen, dass es gültig ist und mit der korrekten Sitzung verknüpft ist.				[Pentest] [Manueller Test]	Überprüfung	Test-ID T-R06

Nutzung Same-Site-Cookie-Attribut: Attribut, dass für Cookies gesetzte wird (Strict/Lax). Cookie wird eingeschränkt und kann von Angreifern nicht mehr genutzt werden.		
--	--	--

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R07	Unsicherheit Inlinescripten		[Hoch]	[Hoch]	[Hoch]	[Vermeiden] [Reduzieren]
Beschreibung						
Inlinescripte sind anfällig für XSS-Angriffe. Sie stellen eine unsicher Codepraktik dar, die zu bösartiger Ausführung von JavaScript führen kann. In HTML durch Java Script neue Erzeugung von HTML, die Variablen nutzen, die ein Angreifer haben könnte.						
Anforderungen						
Keine Sicherheitsrisiken durch Inlinescripte.						
Maßnahmen					Überprüfung	Test-ID
# Keine Nutzung von Inlinescripten bzw. nur bedachte Nutzung. # Bereinigung des Codes.					[Design Review] [Code Review]	T-R07

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R08	Brute-Force-Angriffe	Spoofing, Tampering, Information Discloser, Elevation of Privilege	[Sehr hoch]	[Sehr hoch]	[Sehr hoch]	[Reduzieren]
Beschreibung						
Brute-Force-Angriffe beschreiben das wiederholte Ausprobieren von Benutzernamen und/oder Passwörtern, bis die richtige Kombination gefunden wurde, um Zugriff auf das Konto zu erhalten. Dies gilt für User-Kontos.						
Anforderungen						
Brute-Force-Angriffe sollen verhindert oder verringert bzw. erschwert werden.						
Maßnahmen					Überprüfung	Test-ID
Maximale Anzahl an Login Versuchen auf 5 Versuche. Nach Aufbrauchen dieser wird die Login-Funktion für eine Minute nicht mehr möglich sein. Alle Auffälligkeiten werden geloggt.					[Manueller Test] [Automatisierter Test]	T-R08

Risikoid	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R09	URL Traversal	Tampering, Information Discloser	[Hoch]	[Hoch]	[Hoch]	[Vermeiden]
Beschreibung						
Bei einem URL-Traversal-Angriff versucht ein Angreifer, auf Dateien oder Seiten zuzugreifen, für die er keine Berechtigung hat, indem er die Seitenstruktur einer Website ausnutzt. Dazu gehört auch, dass manipulieren von Parametern.						
Anforderungen						
Es muss sichergestellt werden, dass keine Seiten welche nur mittels Logins erreichbar sind und kundenbezogene Daten aufweisen für Angreifer einsehbar sind.						
Maßnahmen					Überprüfung	Test-ID

Verwendung eines individuellen Authentication-Tokens je User/Usersession.	[Manueller Test] [Automatisierter Test]	T-R09
---	--	-------

RisikoID	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R10	Session Hijacking	Spoofing, Information Discloser	[Mittel]	[Mittel]	[Mittel]	[Reduzieren]
Beschreibung						
Bei einem Session-Hijacking Angriff übernimmt der Angreifer die Session eines Users. Dies kann durch die Session-ID erfolgen, welche z.B. In Cookies enthalten ist. Mit dieser kann der Angreifer die Kontrolle übernehmen ohne Benutzername oder Passwort zu kennen.						
Anforderungen						
Die Generierung der Session ID darf nicht deterministisch sein. Session IDs dürfen nicht unbegrenzt gültig sein, sollte es doch dazu kommen, dass ein Angreifer diese erlangt.						
Maßnahmen	Überprüfung	Test-ID				
Zur Generierung der Session ID nutzen wir die von Django mitgelieferten Funktionen. Session ID innerhalb des Cookies sind nur für eine begrenzte Zeit gültig. Nach Ablauf der Gültigkeit wird der Token vom System nicht mehr akzeptiert, somit können keine Aktionen durchgeführt werden.	[Pentest]	T-R10				

RisikoID	Bedrohung	STRIDE-Kriterien	Eintrittswahrscheinlichkeit	Auswirkungen	Risiko	Behandlung
R11	Ungewollte Änderungen	Spoofing, Tampering, Information Discloser	[Mittel]	[Hoch]	[Mittel]	[Reduzieren]
Beschreibung						
Ein Angreifer erhält Zugriffe/Kontrolle auf ein User-Konto und führt Änderungen an diesem durch.						
Anforderungen						
Ein User muss für bestimmte Handlungen im System gesondert autorisiert werden, um ungewollte Änderungen zu vermeiden.						
Maßnahmen	Überprüfung	Test-ID				
Erneutes Abfragen des User-Passwortes ist notwendig für Änderungen an z.B. User-Daten, Bankdaten oder Vertragsdaten. CAPTCHA-Abfrage für Änderungen im Zusammenhang mit dem Abfragen des erneuten Passwortes. Multi-Faktor-Authentifizierung.	[Manueller Test] [Code Review]	T-R11				

RID	Bedrohung	Asset ID
R01	DDos-Angriff	Frontend Server
R02	SQL-Injection	A01, A06.1, A06.2
R03	XSS-Angriff	A01, A06.1, A06.2
R04	MitM-Angriff	(A01, A03,) User Client
R05	Automatisierte Angriffe	A01, A06.1, A06.2, Frontend/Backend Server
R06	CSRF-Angriffe	A03, wird Client geklaut, aus DB geklaut
R07	Inlinescripte	Frontend Server
R08	Brute-Force-Angriffe	A01, A06.1, A06.2, Frontend Server
R09	URL Traversal	Frontend Server, Backend
R10	Session Hijacking	A03, A07
R11	Ungewollte Änderungen	A01, A06.0, A06.1, A06.2

6 Gewählte Technologien

6.1 Docker

Docker ist eine Container-Software. Diese Container ermöglichen es, bestimmte Softwarekomponenten (Webserver, Programmiersprachen und Datenbanken) ohne den Overhead einer virtuellen Maschine auszuführen. Durch die Verwendung von Docker kann innerhalb des vorliegenden Projektes auf die Nutzung einer virtuellen Maschine oder eines Webservers verzichtet werden, zumindest bei der Entwicklung. Die für das Stromanbieter-Portal benötigten Komponenten sind in dem von uns für das Projekt erstellten Docker -Container enthalten. Docker wird innerhalb des Projektes genutzt, um die Entwicklungs- / Hostsysteme unter den Entwicklern homogen zu gestalten, sodass mit möglichst wenig Aufwand der Tech-Stack stabil bei jedem Entwickler läuft. Des Weiteren wird das Auftreten von vielen Fehlern auf den lokalen Maschinen der Entwickler verhindert.

6.2 Python-Django (Backend / Frontend)

Django ist ein in Python geschriebenes Webframework, welches dem sogenannten Model-View-Presenterschema folgt. Dabei handelt es sich um ein Entwurfsmuster in der Softwareentwicklung, das aus dem Model View Controller (MVC) hervorgegangen ist. Das Django Framework soll für das Backend genutzt werden und dient zur Verwaltung der eigentlichen Anwendung. Nachfolgend wurden weiter Django-Funktionalitäten eingebunden. Dazu zählen unter anderem ein HTML-Frontend (mit Bootstrap 5), CSRF-Token, Input-Validierung und RestAPI. Ein Vorteil von Django, welcher relevant für die Auswahl dieses Frameworks war, ist die vielseitige Auswahl an diversen Security-Modulen, welche zur Absicherung der Webanwendung und Daten genutzt werden können.

6.3 PostgreSQL-Datenbank

PostgreSQL ist ein Open-Source-Objektrelationsdatenbanksystem. Ein Vorteil von diesem Datenbank-Managementsystem ist, dass die Datenbankgröße nicht durch das System, sondern durch den zur Verfügung stehenden Speicher begrenzt wird. Die Datenbank wird über die in das Framework Django integrierte Datenbankabstraktions-API angesprochen mit dem Objekte erstellt, aktualisiert oder gelöscht werden können. Eine Umstellung auf ein anderes DBMS (Datenbankmanagementsystem) wäre aufgrund der dynamisch durch Django generierten Abfragen möglich, da kein natives SQL ausgeführt wird.

7 Komponente und Module

Die vorliegende Webanwendung besteht aus vielen einzelnen Teilmodulen. Die Idee dahinter ist, dass diese Teilmodule jeweils in sich geschlossen sind und nur alle Informationen haben, die sie auch wirklich benötigen. Jede Komponente besteht aus ein oder mehreren Endpunkten, welche die Interaktionen für die Datenbank, einer Serialisierungsinstanz und der UI-Repräsentation definieren. Zu den Komponenten bzw. Modulen unserer Anwendung gehören:

- account_delete (Backend: Löschen des eigenen Benutzeraccounts)
- account_update (Backend: Aktualisieren des eigenen Benutzeraccounts)
- adress (Backend: Adresdaten)
- authentication (Backend: Authentifizierung)
- bank_account (Backend: Zahlweise pro Vertrag)
- contract (Backend: Vertrag)
- energy_tiff (Backend: Energie Tarif)
- main (Backend: Django-Basis)
- measurement_point (Backend: API Messstellenbetreiber)
- registration (Backend: Registrierung)
- static (Globale Dateien: CSS, Javascript)
- templates (Frontend: Templates für UI)
- ui_contract_page (Frontend: Verträge)
- ui_customer_page (Frontend: Kundendashboard)
- ui_customer_profile_page (Frontend: Kundenprofil)
- ui_login_page (Frontend: Login)
- ui_logout_page (Frontend: Logout)
- ui_main_page (Frontend: Startseite)
- ui_pricing_page (Frontend: Strompreise)
- ui_register_page (Frontend: Registrierung)
- utils (Logging, Authentifizierungsfehler)

8 Testplan

8.1 Einleitung

8.1.1 Scope - Umfang

In Scope Konkret zu betrachten in den Tests sind folgende Module:

Modul Name	Beschreibung
Login	Der Login des Users soll nur bei Eingabe der korrekten E-Mail-Adresse und des Passwortes auf das Dashboard des Kunden weiterleiten.
Registrierung	Der User soll sich über das bereitgestellte Anmelde-Formular registrieren können, was zwingend zum Abschluss eines Tarifes führt.
Authentifizierung	Nach erfolgreichem Login des Users, soll dieser sich frei auf der Webseite bewegen können, ohne dass dieser sich erneut authentifizieren muss.

Out of Scope In diesem Testplan wird die Absicherung gegen DDoS Angriffe nicht weiter betrachtet, da das dadurch entstehende Risiko an den Hosting-Dienstleiter ausgelagert wird.

8.1.2 Qualitätsziele

Modul Name	Beschreibung
Login	Der Login des Kunden darf weder gebrute-forced werden, noch darf dieser für SQL-Injektion anfällig sein.
Registrierung	Es darf kein fremder Code in die Seite eingeschleust werden. Des Weiteren muss sichergestellt werden, dass Bots keine automatisierte Anmeldungen oder Registrierungen vornehmen können.
Authentifizierung	Nach erfolgreichem Login des Users, soll dieser sich frei auf der Webseite bewegen können, ohne dass dieser sich erneut authentifizieren zu müssen. Ebenfalls soll der User, beim Ändern seiner Daten zusätzlich sein Passwort eingeben, um somit seine Willenserklärung zu bekunden.
Information Leakage	Es muss sichergestellt werden, dass keine Informationen bei der HTTP-Anfrage mit übergeben werden, die den Nutzer zu unautorisierten Handlungen ermächtigt. Dies impliziert ebenfalls die Ausnutzung von URL Traversal.

8.1.3 Rollen und Verantwortungen

Zur Durchführung der Tests haben wir zusätzliche Rollen bzw. Verantwortungen und damit Aufgaben vergeben:

- Test-Manager: Definiert die Testanforderungen

- Entwickler: Schreiben und Durchführen der Tests, sowie Fehlerbehebung und Entwicklung der eigentlichen Anwendung.

8.2 Test Methodik

Test-Methodik Zum Testen wird die Wasserfall-Methodik verwendet.

Test Level In unserem Testplan testen wir auf den Leveln Integration Testing und System Testing.

Integrationstests prüfen die Interaktion und Integration verschiedener Komponente bzw. Module. Das Ziel hierbei ist die gute Zusammenarbeit zwischen diesen einzelnen Teilen. Durch Systemtest wird die Anwendung als Ganzes getestet. Hierbei werden sowohl die funktionalen Anforderungen als auch die nicht funktionalen Anforderungen (Kapitel 4) überprüft, um die korrekte und effiziente Funktionalität des Systems sicherzustellen.

Vollständigkeit der Prüfung

- Alle Testfälle müssen abgearbeitet sein
- Ausführung und Dokumentation aller manuellen und automatischen Testfälle
- Ein Test gilt als bestanden, wenn keine sicherheitsrelevanten Themen offen bzw. nicht abgedeckt sind und möglich auftretende Bugs dokumentiert sind
- Alle offenen dokumentierten Bugs werden entweder zum Zeitpunkt der Entdeckung behoben oder im kommenden Patch.

8.3 Ergebnisse der Prüfung

Basierend auf dem Risikoregister aus (Kapitel 5.4) ergeben sich folgende Tests:

Test Cases

TestID:	Test Titel:
T-R02	SQL-Injection
Anforderung:	
In Eingabefelder des Logins werden SQL-Statements geschrieben, welche unerlaubte Anfragen an die Datenbank senden. Das Ziel ist es, Informationen aus der DB zu erhalten oder deren Integrität zu kompromittieren. Durch Eingabefelder dürfen keine SQL-Statements geparsed werden.	
Modul:	Überprüfung:
Login	Review des Frameworks
Vorgehen:	
Bewusste Verwendung von Django & PostgreSQL, um SQL-Injection technisch zu unterbinden (Django-Object Relational Mapper)	
Ergebnis:	
Die Interaktion mit der Datenbank erfolgt nicht über festcodierte SQL-Statements. Es wird ein ORM durch Django-Zusatzmodule genutzt.	
Folgemaßnahme:	Tester:
keine	Kai Pistol, Kevin Wagner

TestID:	Test Titel:					
T-R03	Cross-Site-Scripting-Angriff (XSS)					
Anforderung:						
Cross-Site-Scripting-Angriffe zielen darauf ab, dass in Eingabefeldern oder Kommentarfeldern schädlicher Java-Script-Code geschrieben werden kann. Bei anderen Usern wird dieser Code dann ausgeführt, sollten sie auf die Seite gehen, auf welchem sich der Code befindet.						
Modul:	Überprüfung:					
Registrierung, Änderung der User Daten	Pentest					
Vorgehen:						
Eingabe von vorgefertigten Payloads und anschließender Überprüfung des Systems, um zu Prüfen, ob das System für Cross-Site-Scripting verwundbar ist.						
Ergebnis:						
Die Anwendung wurde auf XSS Vulnurabilität geprüft. Dabei wurden XSS Payloads in die möglichen Tarifnamen der Vertragserstellung eingespeist, um den Messstellenbetreiber nicht mit Anfragen zu überlasten.						
Exemplarisch:						
'-alert(1)' '-alert(1)//' \'-alert(1)//'						
Von über 6000 möglichen Payloads führte keine zum Erfolg						
<p style="text-align: center;">Create new Contract</p> <p>Firstname: <input type="text" value="-prompt(8)-"/> Lastname: <input type="text" value=""/> Tariff: <input type="button" value="Select Tariff"/></p> <p>Address</p> <p>Street: <input <input="" formaction="javascript:>" house="" number:="" type="text" value="22"/> Postal code: <input type="text" value="65456"/> City: <input type="text" value="teststadt"/></p> <p>Billing address</p> <p><input checked="" type="checkbox"/> Billing address is the same as address</p> <p>Street: <input <input="" formaction="javascript:>" house="" number:="" type="text" value="22"/> Postal code: <input type="text" value="65456"/> City: <input type="text" value="teststadt"/></p> <p>Bank details</p> <p>Account owner: <input type="text" value="onmouseover=alert(1)//"/> IBAN: <input type="text" value="DE7575884833"/> BIC: <input type="text" value="34634643456"/></p> <p style="text-align: center;"><input type="button" value="Cancel"/> <input type="button" value="Create Contract"/></p>						
<p>(https://github.com/payloadbox/xss-payload-list/blob/master/Intruder/xss-payload-list.txt) .</p> <table border="1"> <tr> <td>Folgemassnahme:</td> <td>Tester:</td> </tr> <tr> <td>keine</td> <td>Kai Pistol, Kevin Wagner</td> </tr> </table>			Folgemassnahme:	Tester:	keine	Kai Pistol, Kevin Wagner
Folgemassnahme:	Tester:					
keine	Kai Pistol, Kevin Wagner					

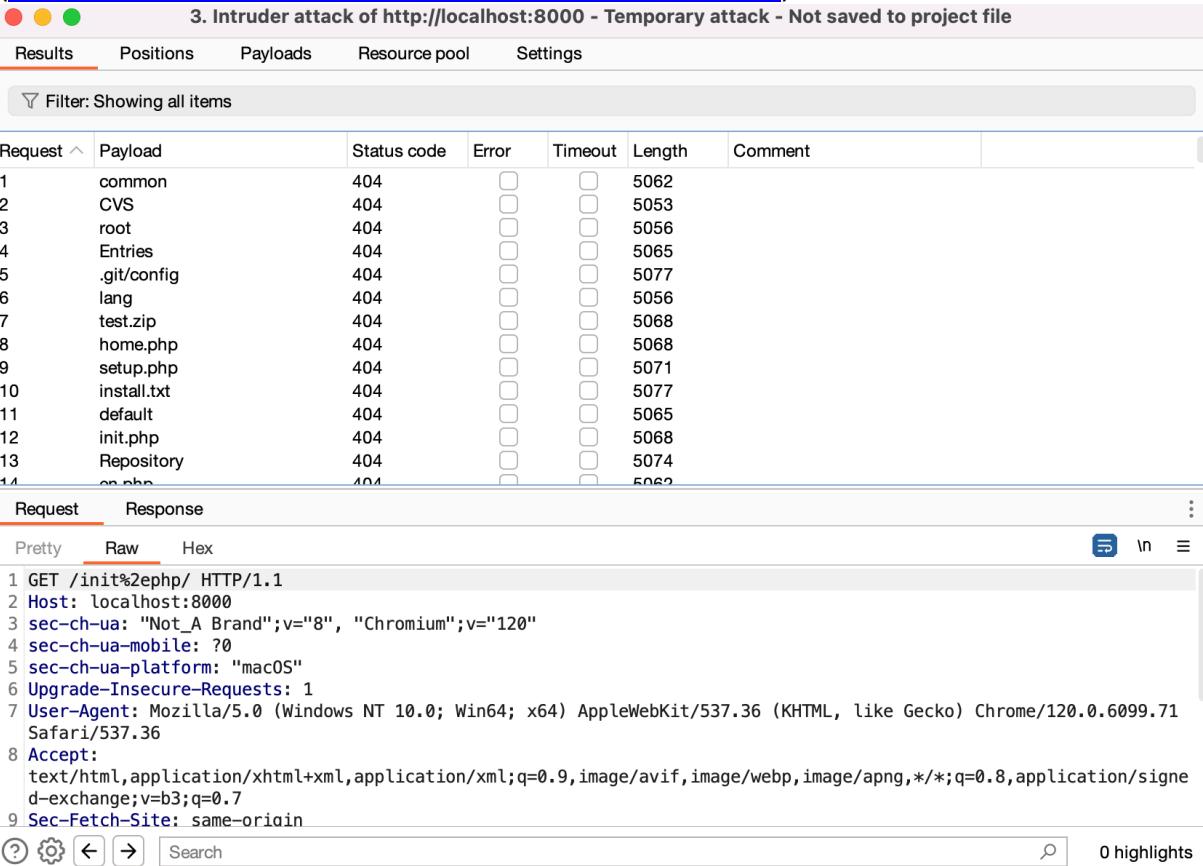
TestID:	Test Titel:
T-R04	Man-in-the-Middle-Angriff (MitM)
Anforderung:	
Alle Anfragen müssen grundsätzlich über verschlüsselte Kanäle laufen und authentifiziert sein, um zu überprüfen, ob es sich um den richtigen User handelt.	
Modul:	Überprüfung:
Systemweit	Code Review
Vorgehen:	
Es muss überprüft werden, ob die Webseite effektiv gegen MitM Attacken vorgeht. Hierfür wird geprüft, ob die Seiten mittels TLS kommunizieren sowie die Identität entsprechend prüfen und ob Maßnahmen darüber hinaus unternommen wurden.	
Ergebnis:	
Es wurde festgestellt, dass die Kommunikation zwischen Stromanbieter und Messstellenbetreiber mittels TLS erfolgt. Dies geht aus der Schnittstellenkonfiguration hervor.	
Folgemaßnahme:	Tester:
keine	Kai Pistol, Luis Eckert

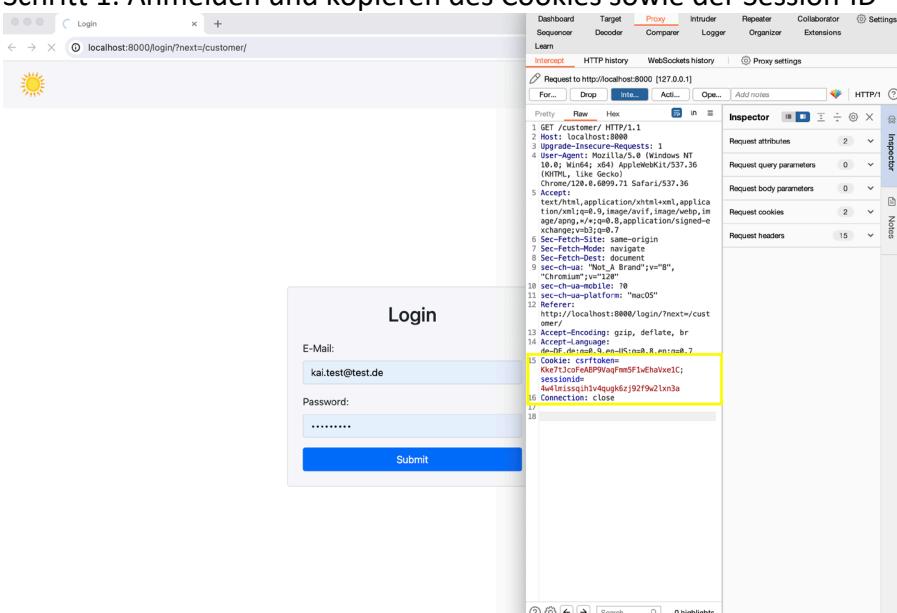
TestID:	Test Titel:
T-R05	Automatisierte Angriffe
Anforderung:	
Formulare dürfen nicht von Bots ausgefüllt bzw. abgeschickt werden.	
Modul:	Überprüfung:
Registrierung	Manueller Test
Vorgehen:	
Überprüfung ob Maßnahmen getroffen wurden, welche gegen die Automatisierung von Anmeldungen vorbeugen oder dieses Reduzieren.	
Ergebnis:	
Um Botnetzen Stand zu halten wurde entschieden einen entsprechenden Schutz des Hosters zu verwenden. Hierfür wird der Netzwerk Traffic über den Anbieter Cloudflare umgeleitet und die Nutzung einer Web- Applikation Firewall von Selbigen genutzt. Des Weiteren wird eine Region Block Liste verwendet um Traffic aus Ländern mit hoher Cyber-Kriminalitätsrate abzufangen. Ebenfalls ist ein Vertrag erst mit der Unterschrift des Kunden aktiv und nicht mit der Registrierung dessen. Jedoch ist herauszustellen, dass keine Maßnahme im Code gefunden, welche gegen Botnetzen vorbeugen.	
Folgemaßnahme:	Tester:
Es wird die Einführung von Captchas empfohlen.	Kai Pistol

TestID:	Test Titel:
T-R06	Cross-Site-Request-Forgery (CSRF)
Anforderung:	
Es dürfen keine CSRF-Angriffe möglich sein bzw. dürfen sie nur geringe Auswirkungen auf das laufende System haben. Es muss sichergestellt werden, dass Anfragen tatsächlich vom User stammen.	
Modul:	Überprüfung:
Datenänderung	[Code Review] [Pentest]
Vorgehen:	
Implementierung eines Anti-CSRF-Token: Hierbei handelt es sich um einen eindeutigen Token, der in jedes Formular eingebettet wird. Beim Absenden oder Anfragen wird dieses verglichen, dass es gültig ist und mit der korrekten Sitzung verknüpft ist.	
Nutzung Same-Site-Cookie-Attribut: Attribut, dass für Cookies gesetzte wird (Strict/Lax). Cookie wird eingeschränkt und kann von Angreifern nicht mehr genutzt werden.	
Ergebnis:	
Es wurde festgestellt, dass das System im Produktiven Betrieb einen Session Based Authentication vornimmt. Hierbei muss jedoch beachtet werden, dass diese nicht im Debug Modus verwendet wird.	
Folgemaßnahme:	Tester:
Der Debug Modus sollte zusätzlich abgesichert werden.	Kai Pistol

TestID:	Test Titel:
T-R07	Inlinescripte
Anforderung:	
Inlinescripte sind anfällig für XSS-Angriffe. Sie stellen eine unsicher Codepraktik dar, die zu bösartiger Ausführung von JavaScript führen kann. Keine Sicherheitsrisiken durch Inlinescripte.	
Modul:	Überprüfung:
Systemweit	Design Review Code Review
Vorgehen:	
Programmentwurf und Code-Review wird überprüft um festzustellen ob tatsächlich keine Inlinescripte verwendet werden.	
Ergebnis:	
Es wurde festgestellt, dass die Anwendung keine Inlinescripte aufweist.	
Folgemaßnahme:	Tester:
Keine	Luis Eckert, Kevin Wagner

TestID:	Test Titel:
T-R08	Bruteforce Angriffe
Anforderung:	
Das System muss solche Bruteforce Angriffe erkennen und den betroffenen Account nach mehrmaligen Versuchen sperren.	
Modul:	Überprüfung:
Login	Manueller Test
Vorgehen:	
Wiederholte Eingabe falscher Passwörter um Log2Fail zu überprüfen.	
Ergebnis:	
Es wurde festgestellt das Log2Fail nicht implementiert wurde. Jedoch wird eine tägliches Log-File erstellt wird, welche fehlerhafte Login-Versuche dokumentiert. Dabei zeichnet die Datenbank über einen Counter für fehlgeschlagene Logins auf. Hierfür wird ein Hash der E-Mail-Adresse geloggt um keine Kundendaten preiszugeben. Der Kunde wird nach sechs fehlgeschlagenen Anmeldeversuchen per Mail informiert.	
<pre>2024-01-05 22:49:34 ----- 2024-01-05 22:49:34 WARNING 05/01/2024 21:49:34 - Unauthorized: /api/v1/auth/ 2024-01-05 22:49:34 Unauthorized: /api/v1/auth/ 2024-01-05 22:49:34 [05/Jan/2024 21:49:34] "POST /api/v1/auth/ HTTP/1.1" 401 31 2024-01-05 22:49:40 WARNING 05/01/2024 21:49:40 - failed authentication for user 369c34cc4d37fefef17780dc262a53e80ab7cb1c051a17ed889d711d5f94f78f3 2024-01-05 22:49:40 Content-Type: text/plain; charset="utf-8" 2024-01-05 22:49:40 MIME-Version: 1.0 2024-01-05 22:49:40 Content-Transfer-Encoding: 7bit 2024-01-05 22:49:40 Subject: Failed Login Attempts 2024-01-05 22:49:40 From: webmaster@localhost 2024-01-05 22:49:40 To: test@test.de 2024-01-05 22:49:40 Date: Fri, 05 Jan 2024 21:49:40 -0000 2024-01-05 22:49:40 Message-ID: <170449138037.45.1630088558057644158@159febfb2fcce6> 2024-01-05 22:49:40 2024-01-05 22:49:40 2024-01-05 22:49:40 Hi kevin wagner, 2024-01-05 22:49:40 2024-01-05 22:49:40 you have 15 failed login attempts. Please change your password to prevent your account from being hacked. 2024-01-05 22:49:40 2024-01-05 22:49:40 http://localhost:8000 2024-01-05 22:49:40 2024-01-05 22:49:40 If you didn't request this, please ignore this email. 2024-01-05 22:49:40 2024-01-05 22:49:40 Your's, 2024-01-05 22:49:40 Your TINF21CS2 SBD Devs 2024-01-05 22:49:40 2024-01-05 22:49:40 ----- 2024-01-05 22:49:40 WARNING 05/01/2024 21:49:40 - Unauthorized: /api/v1/auth/ 2024-01-05 22:49:40 Unauthorized: /api/v1/auth/</pre>	
Folgemaßnahme:	Tester:
Vollständige Implementierung von Log2Fail	Kai Pistol

TestID:	Test Titel:
T-R09	URL Traversal
Anforderung:	
Es muss sichergestellt werden, dass keine Seiten welche nur mittels Logins erreichbar sind und kundenbezogene Daten aufweisen für Angreifer einsehbar sind.	
Modul:	Überprüfung:
Systemweit	[Manueller Test]
Vorgehen:	
Überprüfen welche URLs erreichbar sind.	
Ergebnis:	
Im Rahmen des Tests wurde geprüft ob ein User auf die URLs 127.0.0.1:8000/customer und 127.0.0.1:8000/billing zugreifen kann, ohne dass dieser eingeloggt ist. Ebenfalls wurde ein Scan mittels Burpsuite vorgenommen, ob es weitere Seiten gibt, welche erreichbar wären. Diese Maßnahme umfasste 43137 verschiedene Möglichkeiten von häufigen Pages. Alle Seiten wurden mit der http Responds 404 beantwortet. https://github.com/omurugur/Path_Travsal_Payload_List	
 <p>The screenshot shows the Burp Suite interface. At the top, there's a status bar with three colored circles (red, yellow, green) and the text "3. Intruder attack of http://localhost:8000 - Temporary attack - Not saved to project file". Below the status bar are tabs: Results (which is selected), Positions, Payloads, Resource pool, and Settings. A search bar below the tabs contains the text "Filter: Showing all items". The main area is a table with columns: Request, Payload, Status code, Error, Timeout, Length, and Comment. The table lists 14 rows of 404 errors for various paths like "common", "CVS", "root", etc. Below this table is another section titled "Request Response" with tabs for Pretty, Raw (which is selected), and Hex. It shows a detailed HTTP request header and body. The header includes "GET /init%2ephp/ HTTP/1.1", "Host: localhost:8000", and various user-agent strings. The body is mostly empty with some minor exchange data. At the bottom of the interface are several small icons and a search bar with the text "Search" and "0 highlights".</p>	
Folgemaßnahme:	Tester:
keine	Kai Pistol

TestID:	Test Titel:
T-R10	Session Hijacking
Anforderung:	
Die Generierung der Session ID darf nicht deterministisch sein. Session IDs dürfen nicht unbegrenzt gültig sein, sollte es doch dazu kommen, dass ein Angreifer diese erlangt.	
Modul:	Überprüfung:
Systemweit	Design Review
Vorgehen:	
Es muss geprüft werden, ob die Generierung der Session ID Cookies deterministisch erfolgt.	
Ergebnis:	
Im Design der Anwendung wurde sichergestellt, dass die Session ID's zufällig vergeben werden. Dies hierfür wurde die Django Dokumentation und die Anwendung Django 4.2.7 auf aktuelle CVE's geprüft. Ebenfalls wurde versucht mittels Burp Suit auf eine Terminierte Session zuzugreifen. Ziel dabei war es, zu prüfen ob der Token unmittelbar nach dem Logout ungültig ist. Der durchgeführte Testcase, wies dabei keine Ergebnisse auf, woraus sich schließen lässt, dass die Applikation die Sitzung korrekt beendet.	
Schritt 1: Anmelden und kopieren des Cookies sowie der Session-ID  <p>The screenshot shows a Burp Suite interface. On the left, there is a login form with fields for 'E-Mail:' containing 'kai.test@test.de' and 'Password:' containing '.....'. A 'Submit' button is at the bottom. On the right, the 'Proxy' tab is selected, showing a list of captured requests. The 15th request, which is the login POST request, has its 'Request headers' section highlighted with a yellow box. It contains the 'Cookie' header with the value 'csrftoken=Kx7fJ1oPABPVaWfmF1vEhVwv1C1; sessionid=4w1m1sqshvh4qupk6zj92f9w2lx3a'. This indicates that the session ID is being passed as a cookie, which is a common practice for session management.</p>	

Schritt 2: Abmelden und aufrufen der URL: „<http://localhost:8000/customer/>“ einfügen des Cookies sowie der Session-ID:

What we stand for

Harnessing Nature's Gifts

Commitment to Sustainability

At EcoPower Solutions, we are dedicated to forging a sustainable future powered by renewable energy sources. Our commitment lies in harnessing the potential of clean and green technologies to meet the growing demand for electricity without compromising the health of our planet. Through innovative solutions and cutting-edge technology, we strive to create a world where energy is abundant, affordable, and, most importantly, environmentally friendly. Join us in the journey towards a brighter, cleaner, and more

```

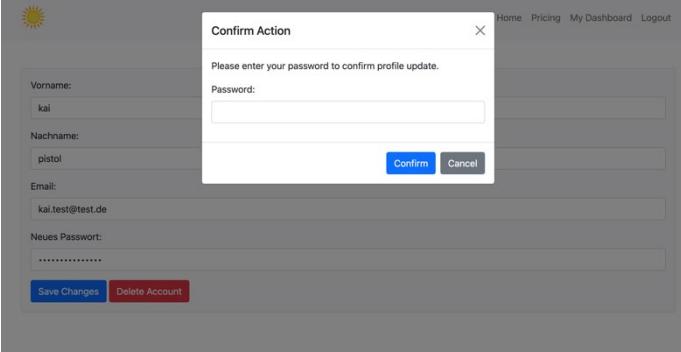
1 GET /login/?next=/customer/ HTTP/1.1
2 Host: localhost:8000
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.71 Safari/537.36
5 Sec-Purpose: prefetch
6 Purpose: prefetch
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
8 Sec-Fetch-Site: none
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Dest: document
11 Sec-Ch-Ua: "Not_A Brand";v="8", "Chromium";v="120"
12 Sec-Ch-Ua-Mobile: ?0
13 Sec-Ch-Ua-Platform: "macOS"
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: de-DE,de@0.9,en-US;q=0.8,en;q=0.7
16 Connection: close
17 Cookies: csrfToken=Kxe7tjcof4Bp9VaqFmm5F1wEhaVxe1C; sessionId=4w4lmissqih1v4qugk6zj92f9w2lxn3a
18 Connection: close
19
20
EcoP
comm
energ
our e
provid
Our p
empow
partic
By inv
prom
to cre

```

Ergebnis:

User wird auf die Login Page weitergeleitet. Session ist folglich richtig terminiert wurden und lässt nicht eine erneute Anmeldung mit dem gleichen Cookie & Session-ID zu.

Folgemaßnahme:	Tester:
keine	Kai Pistol

TestID:	Test Titel:
T-R11	Ungewollte Änderungen
Anforderung:	
Ein User muss für bestimmte Handlungen im System gesondert autorisiert werden, um ungewollte Änderungen zu vermeiden.	
Modul:	Überprüfung:
Datenänderungen	[Manueller Test] [Code Review]
Vorgehen:	
Es muss geprüft werden, dass ein User seine Datenänderung im System aktiv durch die Eingabe seines Passworts bestätigt.	
Ergebnis:	
Abbildung 1: Es wurde festgestellt, dass die Änderung durch die Eingabe seines Passworts bestätigen kann. Hierfür wurde dies Anhand einer Passwortänderung versucht.	
 <p>Durch eine Code Review wurde ebenfalls sichergestellt, dass dieses Vorgehen auch beim Löschen von Verträgen zum tragen kommt.</p>	
Ebenfalls ist aber auch festzuhalten, dass der Einsatz von Multi-Faktor Authentifizierung (MFA) präferiert werden würde.	
Folgemaßnahme:	Tester:
Umstellung auf MFA	Kai Pistol, Luis Eckert

8.4 Notwendige Anwendungen und Programme

8.4.1 Testing Tool

Test Methode	Beschreibung
Pentesting	<ul style="list-style-type: none">• Burp Suite• Postman• Log Review
Code Review/ Design Review	<ul style="list-style-type: none">• Static Application Security Testing (SAST)

8.4.2 Testing Umgebung

Die Minimalanforderungen für die Testumgebung sind:

- Docker
- Internetbrowser

9 Projektumsetzung

Zur Projektumsetzung ist zu sagen, dass wir uns so gut wie möglich an die am Anfang gestellte Planung gehalten haben. Wie erwähnt nutzen wir die Plattform GitHub zum Austausch unserer Dokumente sowie Code. Hier haben wir ein Git-Projekt erstellt, in welchem wir Aufgaben, an denen wir aktuell gearbeitet haben bzw. Aufgaben für den nächsten Sprint, definiert haben. Hierbei haben wir zwei „Views“ genutzt. Bei der ersten handelt es sich um eine Gesamtübersicht aller angelegter Issues, wem diese zugeordnet sind und welchen Status sie haben. Hier wurde zwischen den Kriterien

- Backlog
- ToDo
- In Progress
- Done

unterschieden. Die zweite View beinhaltet grundsätzlich die gleichen Informationen, nur dass in dieser eine Aufteilung nach den Kriterien gemacht wurde. Diese View wurde genutzt, um die Planung der Sprints zu erleichtern und zu gucken, an welchen Aufgaben wir gerade arbeiten bzw. welche Aufgaben noch zu tun sind.

In unserer Bearbeitung sind wir iterativ vorgegangen. Das heißtt, wir haben in einem Meeting Aufgaben zugeteilt, welche bis zum nächsten Meeting bearbeitet werden sollen. In diesem wurde dann verglichen, welche Aufgaben wie weit erledigt waren, um dann neue Aufgaben zu bilden. Außerdem bedeutet es, dass wir mit voranschreitenden Arbeiten auch bereits „fertige“ Dokumente nochmal überarbeitet haben. Hierzu zählt z.B. das Architekturdiagramm. In diesen Meetings haben wir uns grundsätzlich abgesprochen, ob es Probleme gibt oder die Ergebnisse reviewt.

Zur Code-Erstellung ist zu sagen, dass wir nach den Prinzipien des Pair-Programming bzw. des 4-Augen-Prinzips vorgegangen sind. das beinhaltet, dass geschriebener Code von mindestens einer anderen Person reviewt wurde, um die Funktionalität bzw. Sinnhaftigkeit des Codes zu bewerten. Je nach der Relevanz bzw. des Umfangs der Funktion fiel diese Review mehr oder weniger detailreich aus. Sollte die Funktionalität des Codes nicht gewährleistet sein, darf dieser auch nicht auf das laufende GitHub Projekt landen, um die Beeinträchtigung des Systems zu verhindern. Des Weiteren war bei größeren Änderungen, wie z.B. die Änderung der Architektur eine Absprache mit dem PO notwendig.

Für die Entwicklung und dem Testing des gesamten Systems wurde eine containerisierte Umgebung aufgesetzt und in dieser gearbeitet. Diese ermöglicht annähernd realistische Umstände eines Produktivsystems. Für das Testen der Module wurde darauf geachtet, dass hierbei die vermittelten Inhalte der Vorlesung Security by Design umgesetzt wurden. Ebenfalls wurde in der Beurteilung der Folgemaßnahmen, der Fokus über die im Zeitraum implementierten Sicherheitsmaßnahmen erweitert und sinnvoll ergänzt. Für die Tests selbst wurden sowohl Code-, sowie Design Reviews vorgenommen, als auch Pentesting Maßnahmen in Form von Burpsuit oder manuelle Tests zum Prüfen der Anwendung.

Aufgrund mangelnder Erfahrung mit gewissen Tools konnten einzelne Kontrollmechanismen nicht vollständig implementiert werden.

Der gesamte Code zum Projekt ist auf GitHub unter folgendem Link zu finden:
https://github.com/CelMur/TINF21CS2_Security_by_Design_Gruppenprojekt
Zusätzlich sind weitere Informationen in der READ.me-Datei zu finden. Zu diesen gehören die Aufsetzung des Projekts sowie zusätzliche Informationen.

10 Quellen

- Öggl Bernd, Koffler Michael: Docker - Das Praxisbuch für Entwickler
- Ernesti Johannes, Kaiser Peter: Python3 - Das umfassende Handbuch