Test Cases

TestID:	Test Titel:
T-R02	SQL-Injection
Anforderung:	
In Eingabefelder des Logins 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. Durch Eingabefelder darf keiner ausführbarer SQL-Code in die DB gelangen.	
Modul:	**Überprüfung:**
Login	Review des Frameworks
Vorgehen:	
Bewusste Verwendung von Django & PostgreSQL um SQL-Injection technisch zu unterbinden.	
Ergebnis:	
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).	
Folgemaßnahme:	**Tester:**
keine	Kai Pistol

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 System Cross-Site-Scripting vulnerable ist.	
Ergebnis:	

Die Usereingabe im Registrierungsformular wurde mit verschiedenen XSS Payloads beschrieben.
Exemplarisch:
'-alert(1)-'
'-alert(1)//
\'-alert(1)//
Hierbei konnte keine Reaktion der Anwendung verzeichnet werden.	

Folgemaßnahme:	**Tester:**

keine	Kai Pistol

TestID:	**Test Titel:**
T-R04	Man-in-the-Middle-Angriff (MitM)
Anforderung:	
Alle Anfragen müssen grundsätzlich über verschlüsselte Kanäle laufen. Alle Anfragen müssen authentifiziert sein, um zu überprüfen, dass es sich um den richtigen User handelt.	
Modul:	**Überprüfung:**
Systemweit	Automatisierter Test
Vorgehen:	
Es muss überprüft werden, ob die Webseite effektiv gegen MitM Attacken vorgeht. Hierfür wird geprüft, ob die Seite TLS-Verschlüsselung Vornimmt und welche Maßnahmen darüber hinaus unternommen wurden.	
Ergebnis:	
Es wurde festgestellt, dass die Seite nicht mit TLS überträgt. Zwar ist die Verwendung von einem SSL-Zertifikat in Verbindung mit der Ablehnung von http Anfragen vorgesehen. Ebenfalls ist Session Hijacking ausgeschlossen, jedoch ist die Anwendung ohne TLS nicht ausreichend geschützt.	
Folgemaßnahme:	**Tester:**
Es muss die Implementierung von TLS erfolgen.	Kai Pistol

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 Cloudflair 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 Kundens 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-Referenz-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 Authentification vornimmt. Hierbei muss jedoch beachtet werden das 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 aufweißt.	
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, dass das Throttling nicht implementiert wurde. Herauszustellen ist jedoch, dass die Datenbank über einen Counter für fehlgeschlagene Logins verfügt und diese ebenfalls geloggt werden. Hierfür wird ein Hash der E-Mail-Adresse geloggt um keine Kundendaten preiszugeben.	
Folgemaßnahme:	**Tester:**
Implementierung von Throttling	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/user_dashboard und 127.0.0.1:8000/billing zugreifen kann, ohne dass dieser eingeloggt ist. Ebenfalls wurde ein Scan mittles Burpsuit vorgenommen, ob es weitere Seiten gibt, welche erreichbar wären, was jedoch zu keinen Ergebnis führte.	
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.	
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 seine Passworts bestätigt.	
Ergebnis:	
Es wurde festgestellt, dass die Änderung durch die Eingabe seines Passworts bestätigen kann. 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