**Dokumentacja Testów - System Helpdesk**

**Spis Treści**

1. **Wprowadzenie**
2. **Testy Funkcjonalne**
3. **Testy Bezpieczeństwa**
4. **Testy Automatyczne**
5. **Środowisko Testowe**
6. **Metodologia Testów**

**Wprowadzenie**

**Niniejsza dokumentacja opisuje kompleksowy plan testów dla systemu helpdesk obejmujący testy funkcjonalne, bezpieczeństwa oraz automatyczne. System składa się z trzech głównych ról użytkowników: klient, agent/superagent oraz administrator.**

**Cele Testów**

* **Weryfikacja poprawności funkcjonalności systemu**
* **Zapewnienie bezpieczeństwa danych i dostępu**
* **Automatyzacja kluczowych scenariuszy testowych**
* **Walidacja ścieżek użytkowników**

**Testy Funkcjonalne**

**1. Formularz Zgłoszenia**

**TC-F-001: Poprawne ustawienie wartości pól formularza**

**Cel: Weryfikacja poprawnego wypełnienia i walidacji pól formularza zgłoszenia**

**Kroki testowe:**

1. **Otwórz formularz tworzenia zgłoszenia**
2. **Wypełnij wszystkie wymagane pola poprawnymi danymi:** 
   * **Tytuł zgłoszenia**
   * **Opis problemu**
   * **Kategoria**
   * **Akceptacja regulaminu**
3. **Sprawdź walidację pól wymaganych**
4. **Zweryfikuj ograniczenia formatów danych**

**Oczekiwany rezultat: Formularz akceptuje poprawne dane i wyświetla błędy walidacji dla niepoprawnych**

**TC-F-002: Wywołanie funkcji onSubmit**

**Cel: Sprawdzenie czy po zatwierdzeniu poprawnego formularza wywoływana jest właściwa funkcja**

**Kroki testowe:**

1. **Wypełnij formularz poprawnymi danymi**
2. **Kliknij przycisk "Wyślij"**
3. **Zweryfikuj czy zgłoszenie zostało utworzone**
4. **Sprawdź czy użytkownik został przekierowany na właściwą stronę**

**Oczekiwany rezultat: Zgłoszenie zostaje utworzone i zapisane w systemie**

**TC-F-003: Filtracja zgłoszeń wg kategorii/statusu**

**Cel: Weryfikacja działania filtrów w liście zgłoszeń**

**Kroki testowe:**

1. **Przejdź do listy zgłoszeń**
2. **Zastosuj filtr według kategorii**
3. **Zastosuj filtr według statusu**
4. **Zweryfikuj wyniki filtrowania**

**Oczekiwany rezultat: Lista wyświetla tylko zgłoszenia spełniające kryteria filtra**

**TC-F-004: Reducer zmienia status zgłoszenia**

**Cel: Sprawdzenie mechanizmu zmiany statusu zgłoszenia**

**Kroki testowe:**

1. **Otwórz zgłoszenie o statusie "open"**
2. **Zmień status na "closed"**
3. **Zapisz zmiany**
4. **Zweryfikuj aktualizację statusu w systemie**

**Oczekiwany rezultat: Status zgłoszenia zostaje poprawnie zaktualizowany**

**2. Panel Administratora**

**TC-F-005: Dodawanie użytkowników**

**Cel: Weryfikacja funkcjonalności dodawania nowych użytkowników przez administratora**

**Kroki testowe:**

1. **Zaloguj się jako administrator**
2. **Przejdź do panelu zarządzania użytkownikami**
3. **Kliknij "Dodaj użytkownika"**
4. **Wypełnij formularz danych użytkownika**
5. **Zapisz nowego użytkownika**

**Oczekiwany rezultat: Nowy użytkownik zostaje utworzony w systemie**

**TC-F-006: Nadawanie ról**

**Cel: Sprawdzenie możliwości przypisywania ról użytkownikom**

**Kroki testowe:**

1. **Otwórz panel użytkowników**
2. **Wybierz użytkownika**
3. **Przypisz rolę (admin, agent, klient)**
4. **Zapisz zmiany**
5. **Zweryfikuj uprawnienia użytkownika**

**Oczekiwany rezultat: Rola zostaje przypisana i uprawnienia są aktywne**

**TC-F-007: Blokowanie/odblokowywanie konta**

**Cel: Weryfikacja funkcjonalności blokowania dostępu do kont użytkowników**

**Kroki testowe:**

1. **Wybierz aktywne konto użytkownika**
2. **Zablokuj konto**
3. **Spróbuj zalogować się na zablokowane konto**
4. **Odblokuj konto**
5. **Ponownie sprawdź możliwość logowania**

**Oczekiwany rezultat: Zablokowane konto nie może się zalogować, odblokowane może**

**3. Ścieżka Użytkownika: Klient**

**TC-F-008: Rejestracja nowego użytkownika**

**Cel: Weryfikacja procesu rejestracji nowego klienta**

**Kroki testowe:**

1. **Otwórz formularz rejestracji**
2. **Wypełnij wszystkie wymagane pola**
3. **Zaakceptuj regulamin**
4. **Wyślij formularz rejestracji**
5. **Sprawdź potwierdzenie rejestracji**

**Oczekiwany rezultat: Konto zostaje utworzone i oczekuje na aktywację**

**TC-F-009: Zalogowanie i dodanie zgłoszenia**

**Cel: Sprawdzenie pełnej ścieżki od logowania do utworzenia zgłoszenia**

**Kroki testowe:**

1. **Zaloguj się jako klient**
2. **Przejdź do sekcji zgłoszeń**
3. **Utwórz nowe zgłoszenie**
4. **Wypełnij wszystkie pola**
5. **Wyślij zgłoszenie**

**Oczekiwany rezultat: Zgłoszenie zostaje utworzone i jest widoczne w panelu**

**TC-F-010: Przeglądanie zgłoszeń w panelu klienta**

**Cel: Weryfikacja widoku zgłoszeń dla klienta**

**Kroki testowe:**

1. **Zaloguj się jako klient**
2. **Przejdź do "Moje zgłoszenia"**
3. **Sprawdź listę zgłoszeń**
4. **Otwórz szczegóły zgłoszenia**

**Oczekiwany rezultat: Klient widzi tylko swoje zgłoszenia z pełnymi szczegółami**

**TC-F-011: Odpowiedź do zgłoszenia przez formularz komentarzy**

**Cel: Sprawdzenie możliwości dodawania komentarzy do zgłoszeń**

**Kroki testowe:**

1. **Otwórz szczegóły zgłoszenia**
2. **Przejdź do sekcji komentarzy**
3. **Dodaj komentarz**
4. **Wyślij komentarz**
5. **Zweryfikuj wyświetlenie komentarza**

**Oczekiwany rezultat: Komentarz zostaje dodany i jest widoczny w historii**

**TC-F-012: Wylogowanie**

**Cel: Weryfikacja procesu wylogowania**

**Kroki testowe:**

1. **Będąc zalogowanym, kliknij opcję wylogowania**
2. **Spróbuj uzyskać dostęp do chronionej strony**
3. **Sprawdź przekierowanie na stronę logowania**

**Oczekiwany rezultat: Sesja zostaje zakończona, dostęp do panelu zablokowany**

**4. Ścieżka Użytkownika: Agent/Superagent**

**TC-F-013: Przeglądanie zgłoszeń przypisanych do siebie**

**Cel: Weryfikacja widoku zgłoszeń przypisanych do agenta**

**Kroki testowe:**

1. **Zaloguj się jako agent**
2. **Przejdź do listy zgłoszeń**
3. **Sprawdź filtr "Przypisane do mnie"**
4. **Zweryfikuj czy wyświetlane są tylko właściwe zgłoszenia**

**Oczekiwany rezultat: Agent widzi tylko zgłoszenia przypisane do siebie**

**TC-F-014: Zmiana statusu zgłoszenia**

**Cel: Sprawdzenie możliwości zmiany statusu przez agenta**

**Kroki testowe:**

1. **Otwórz przypisane zgłoszenie**
2. **Zmień status na "przyjęte"**
3. **Następnie zmień na "rozwiązane"**
4. **Zapisz zmiany**

**Oczekiwany rezultat: Status zostaje zaktualizowany w systemie**

**TC-F-015: Przypisanie zgłoszenia innemu agentowi**

**Cel: Weryfikacja możliwości przekazywania zgłoszeń między agentami**

**Kroki testowe:**

1. **Otwórz zgłoszenie przypisane do siebie**
2. **Wybierz opcję "Przypisz do innego agenta"**
3. **Wybierz docelowego agenta**
4. **Potwierdź zmianę**
5. **Sprawdź aktualizację przypisania**

**Oczekiwany rezultat: Zgłoszenie zostaje przypisane do wybranego agenta**

**5. Ścieżka Administratora**

**TC-F-016: Zarządzanie zgłoszeniami**

**Cel: Weryfikacja uprawnień administratora do zarządzania zgłoszeniami**

**Kroki testowe:**

1. **Zaloguj się jako administrator**
2. **Otwórz dowolne zgłoszenie**
3. **Edytuj szczegóły zgłoszenia**
4. **Usuń wybrane zgłoszenie**
5. **Sprawdź zmiany w systemie**

**Oczekiwany rezultat: Administrator może modyfikować i usuwać zgłoszenia**

**TC-F-017: Przypisywanie ról**

**Cel: Sprawdzenie zarządzania rolami użytkowników**

**Kroki testowe:**

1. **Przejdź do zarządzania użytkownikami**
2. **Wybierz użytkownika**
3. **Zmień jego rolę**
4. **Zapisz zmiany**
5. **Zweryfikuj nowe uprawnienia**

**Oczekiwany rezultat: Rola zostaje zmieniona i uprawnienia są aktywne**

**Testy Bezpieczeństwa**

**1. Uwierzytelnianie i Autoryzacja**

**TC-S-001: Próba wysłania formularza bez uprawnień (401)**

**Cel: Weryfikacja ochrony przed nieautoryzowanym dostępem**

**Kroki testowe:**

1. **Wyloguj się z systemu**
2. **Spróbuj wysłać żądanie POST do chronionego endpointu**
3. **Sprawdź kod odpowiedzi HTTP**

**Oczekiwany rezultat: System zwraca błąd 401 Unauthorized**

**TC-S-002: Próba dostępu do panelu admina bez uprawnień**

**Cel: Sprawdzenie ochrony panelu administratora**

**Kroki testowe:**

1. **Zaloguj się jako zwykły użytkownik**
2. **Spróbuj uzyskać dostęp do panelu admina**
3. **Sprawdź czy następuje przekierowanie lub błąd**

**Oczekiwany rezultat: Dostęp zostaje zablokowany**

**TC-S-003: Brak dostępu do danych innego użytkownika (IDOR)**

**Cel: Weryfikacja ochrony przed Insecure Direct Object Reference**

**Kroki testowe:**

1. **Zaloguj się jako użytkownik A**
2. **Zanotuj ID zgłoszenia użytkownika A**
3. **Zmień ID w URL na ID należące do użytkownika B**
4. **Sprawdź czy dane są dostępne**

**Oczekiwany rezultat: System blokuje dostęp do cudzych danych**

**TC-S-004: Wymuszenie silnego hasła**

**Cel: Sprawdzenie polityki haseł**

**Kroki testowe:**

1. **Spróbuj utworzyć konto ze słabym hasłem**
2. **Testuj różne kombinacje:** 
   * **Krócej niż 8 znaków**
   * **Bez wielkich liter**
   * **Bez cyfr**
   * **Bez znaków specjalnych**

**Oczekiwany rezultat: Słabe hasła są odrzucane z odpowiednim komunikatem**

**TC-S-005: Wygasanie tokenu JWT**

**Cel: Weryfikacja mechanizmu wygasania sesji**

**Kroki testowe:**

1. **Zaloguj się do systemu**
2. **Poczekaj na wygaśnięcie tokena**
3. **Spróbuj wykonać chronioną operację**
4. **Sprawdź czy następuje przekierowanie na logowanie**

**Oczekiwany rezultat: Po wygaśnięciu tokena wymagane jest ponowne logowanie**

**2. Ochrona przed XSS**

**TC-S-006: Wprowadzenie skryptu w formularzu**

**Cel: Weryfikacja ochrony przed Cross-Site Scripting**

**Kroki testowe:**

1. **W polu opisu zgłoszenia wprowadź: <script>alert(1)</script>**
2. **Wyślij formularz**
3. **Otwórz zgłoszenie i sprawdź czy skrypt się wykonuje**

**Oczekiwany rezultat: Skrypt nie zostaje wykonany, dane są filtrowane**

**3. Ochrona CSRF**

**TC-S-007: Żądania wymagają tokenu CSRF**

**Cel: Sprawdzenie ochrony przed Cross-Site Request Forgery**

**Kroki testowe:**

1. **Przygotuj żądanie POST/PUT/DELETE bez tokenu CSRF**
2. **Wyślij żądanie**
3. **Sprawdź odpowiedź serwera**

**Oczekiwany rezultat: Żądanie zostaje odrzucone**

**TC-S-008: Blokada żądań z innego originu**

**Cel: Weryfikacja polityki CORS**

**Kroki testowe:**

1. **Z innej domeny wyślij żądanie AJAX**
2. **Sprawdź nagłówki CORS w odpowiedzi**
3. **Zweryfikuj czy żądanie zostaje zablokowane**

**Oczekiwany rezultat: Żądania z niepożądanych originów są blokowane**

**4. Ochrona przed wstrzykiwaniem danych**

**TC-S-009: Próba SQL Injection**

**Cel: Weryfikacja ochrony przed wstrzykiwaniem SQL**

**Kroki testowe:**

1. **W polu formularza wprowadź: '; DROP TABLE users;--**
2. **Wyślij formularz**
3. **Sprawdź czy dane w bazie pozostają nienaruszone**

**Oczekiwany rezultat: Próba wstrzykiwania SQL nie wpływa na bazę danych**

**5. Skanowanie Bezpieczeństwa (Kali Linux)**

**TC-S-010: Skanowanie portów - Nmap**

**Cel: Identyfikacja otwartych portów i usług**

**Narzędzie: Nmap (Kali Linux)  
Polecenie: nmap -sV -Pn <target\_ip>**

**Dodatkowe skanowania:**

**# Skanowanie agresywne**

**nmap -A -T4 <target\_ip>**

**# Skanowanie podatności**

**nmap --script vuln <target\_ip>**

**# Skanowanie ukrytych portów**

**nmap -p- <target\_ip>**

**Oczekiwany rezultat: Tylko niezbędne porty są otwarte (80, 443), brak niepotrzebnych usług**

**TC-S-011: Wykrywanie podatności - Nikto**

**Cel: Identyfikacja podatności aplikacji webowych**

**Narzędzie: Nikto (Kali Linux)  
Polecenie: nikto -h http://betulait.usermd.net**

**Dodatkowe opcje:**

**# Skanowanie z zapisem do pliku**

**nikto -h http://betulait.usermd.net -o nikto\_report.html -Format htm**

**# Skanowanie przez proxy**

**nikto -h http://betulait.usermd.net -useproxy http://127.0.0.1:8080**

**Oczekiwany rezultat: Brak krytycznych podatności aplikacji webowej**

**8. Testowanie Aplikacji Webowych (Kali Linux)**

**TC-S-017: Skanowanie podatności OWASP - ZAP**

**Cel: Automatyczne wykrywanie podatności aplikacji webowych**

**Narzędzie: OWASP ZAP (Kali Linux)**

**Procedura:**

1. **Uruchom ZAP: zaproxy**
2. **Skonfiguruj proxy przeglądarki (127.0.0.1:8080)**
3. **Przeglądaj aplikację manualnie (spider)**
4. **Uruchom Active Scan**
5. **Przeanalizuj wyniki**

**Polecenia CLI:**

**# Skanowanie automatyczne**

**zap-cli quick-scan --self-contained http://betulait.usermd.net**

**# Skanowanie z raportem**

**zap-cli quick-scan --self-contained --reportPath zap\_report.html http://betulait.usermd.net**

**Oczekiwany rezultat: Brak podatności o wysokim i średnim ryzyku**

**TC-S-018: Testowanie Burp Suite**

**Cel: Manuale testowanie bezpieczeństwa aplikacji**

**Narzędzie: Burp Suite Community (Kali Linux)**

**Procedura testowa:**

1. **Skonfiguruj proxy Burp (127.0.0.1:8080)**
2. **Przechwytuj ruch HTTP/HTTPS**
3. **Testuj:** 
   * **Parameter manipulation**
   * **Session management**
   * **Authentication bypass**
   * **Input validation**

**Oczekiwany rezultat: Aplikacja odporna na podstawowe ataki**

**TC-S-019: Analiza SSL/TLS - SSLyze**

**Cel: Weryfikacja konfiguracji SSL/TLS**

**Narzędzie: SSLyze (Kali Linux)**

**Polecenia:**

**# Podstawowa analiza SSL**

**sslyze betulait.usermd.net**

**# Szczegółowa analiza**

**sslyze --regular betulait.usermd.net**

**# Sprawdzenie podatności**

**sslyze --heartbleed --openssl\_ccs --robot betulait.usermd.net**

**Oczekiwany rezultat: Bezpieczna konfiguracja SSL/TLS, brak znanych podatności**

**TC-S-020: Testowanie SQLi - SQLmap**

**Cel: Automatyczne wykrywanie podatności SQL Injection**

**Narzędzie: SQLmap (Kali Linux)**

**Polecenia testowe:**

**# Test podstawowy**

**sqlmap -u "http://betulait.usermd.net/tickets?id=1" --batch**

**# Test POST parameters**

**sqlmap -u "http://betulait.usermd.net/login" --data="username=test&password=test" --batch**

**# Test z cookies**

**sqlmap -u "http://betulait.usermd.net/profile" --cookie="sessionid=abc123" --batch**

**# Test z custom headers**

**sqlmap -u "http://betulait.usermd.net/api/tickets" --headers="Authorization: Bearer token123" --batch**

**Oczekiwany rezultat: Brak podatności SQL Injection**

**9. Testowanie Sieci i Infrastruktury (Kali Linux)**

**TC-S-021: Skanowanie sieci - Netdiscover**

**Cel: Mapowanie sieci i aktywnych hostów**

**Narzędzie: Netdiscover (Kali Linux)**

**Polecenia:**

**# Skanowanie pasywne**

**netdiscover -r 192.168.1.0/24 -P**

**# Skanowanie aktywne**

**netdiscover -r 192.168.1.0/24**

**Oczekiwany rezultat: Identyfikacja wszystkich hostów w sieci**

**TC-S-022: Analiza ruchu sieciowego - Wireshark**

**Cel: Monitorowanie i analiza komunikacji sieciowej**

**Narzędzie: Wireshark (Kali Linux)**

**Procedura:**

1. **Uruchom przechwytywanie pakietów**
2. **Wykonaj operacje w aplikacji**
3. **Analizuj przechwyconą komunikację:** 
   * **Czy dane są szyfrowane**
   * **Czy występują niezabezpieczone protokoły**
   * **Czy hasła są przesyłane w plain text**

**Oczekiwany rezultat: Cała komunikacja zabezpieczona, brak wrażliwych danych w plain text**

**TC-S-023: Test Man-in-the-Middle - Ettercap**

**Cel: Weryfikacja odporności na ataki MITM**

**Narzędzie: Ettercap (Kali Linux)**

**Polecenia:**

**# ARP Spoofing**

**ettercap -T -M arp:remote /192.168.1.1// /192.168.1.100//**

**# DNS Spoofing**

**ettercap -T -M arp:remote -P dns\_spoof /192.168.1.1// /192.168.1.100//**

**Oczekiwany rezultat: Aplikacja wykrywa lub jest odporna na ataki MITM**

**10. Testowanie Wireless (Kali Linux)**

**TC-S-024: Audit WiFi - Aircrack-ng Suite**

**Cel: Testowanie bezpieczeństwa sieci bezprzewodowej**

**Narzędzia: Aircrack-ng Suite (Kali Linux)**

**Polecenia:**

**# Monitor mode**

**airmon-ng start wlan0**

**# Skanowanie sieci**

**airodump-ng wlan0mon**

**# Przechwytywanie handshake**

**airodump-ng -c 6 --bssid AA:BB:CC:DD:EE:FF -w capture wlan0mon**

**# Testowanie hasła**

**aircrack-ng -w /usr/share/wordlists/rockyou.txt capture.cap**

**Oczekiwany rezultat: WiFi używa silnego szyfrowania i bezpiecznych haseł**

**TC-S-012: Ochrona plików konfiguracyjnych**

**Cel: Sprawdzenie dostępności wrażliwych plików**

**Kroki testowe:**

1. **Spróbuj uzyskać dostęp do: /.env, /.git, /package-lock.json**
2. **Sprawdź odpowiedź serwera**

**Oczekiwany rezultat: Pliki konfiguracyjne nie są dostępne przez HTTP**

**TC-S-013: Ograniczenie uploadu plików**

**Cel: Weryfikacja kontroli przesyłanych plików**

**Kroki testowe:**

1. **Spróbuj przesłać plik o niedozwolonym rozszerzeniu**
2. **Spróbuj przesłać plik przekraczający limit rozmiaru**
3. **Sprawdź walidację typu MIME**

**Oczekiwany rezultat: Nieprawidłowe pliki są odrzucane**

**7. Testy Rate Limiting i Brute Force (Kali Linux)**

**TC-S-014: Ograniczenie prób logowania - Hydra**

**Cel: Weryfikacja ochrony przed atakami brute-force**

**Narzędzie: Hydra (Kali Linux)**

**Polecenia testowe:**

**# Atak słownikowy na login**

**hydra -l admin -P /usr/share/wordlists/rockyou.txt betulait.usermd.net http-post-form "/login/:username=^USER^&password=^PASS^:Invalid"**

**# Test z listą użytkowników**

**hydra -L users.txt -P passwords.txt betulait.usermd.net http-post-form "/login/:username=^USER^&password=^PASS^:Invalid"**

**# Ograniczony atak czasowy**

**hydra -l admin -P /usr/share/wordlists/rockyou.txt -t 1 -W 30 betulait.usermd.net http-post-form "/login/:username=^USER^&password=^PASS^:Invalid"**

**Oczekiwany rezultat: Po określonej liczbie prób (np. 5) dostęp zostaje tymczasowo zablokowany**

**TC-S-015: Fuzzing aplikacji - WFUZZ**

**Cel: Testowanie odporności aplikacji na nieprawidłowe dane**

**Narzędzie: WFUZZ (Kali Linux)**

**Polecenia testowe:**

**# Fuzzing parametrów GET**

**wfuzz -c -z file,/usr/share/wfuzz/wordlist/general/common.txt --hc 404 http://betulait.usermd.net/tickets?FUZZ=test**

**# Fuzzing hidden directories**

**wfuzz -c --hc 404 -z file,/usr/share/dirb/wordlists/common.txt http://betulait.usermd.net/FUZZ**

**# POST parameter fuzzing**

**wfuzz -c -z file,payloads.txt -d "username=admin&password=FUZZ" http://betulait.usermd.net/login**

**Oczekiwany rezultat: Aplikacja obsługuje nieprawidłowe dane bez crashowania**

**TC-S-016: Skanowanie katalogów - Dirb/Gobuster**

**Cel: Wykrywanie ukrytych katalogów i plików**

**Narzędzie: Dirb/Gobuster (Kali Linux)**

**Polecenia testowe:**

**# Skanowanie podstawowe - Dirb**

**dirb http://betulait.usermd.net /usr/share/dirb/wordlists/common.txt**

**# Skanowanie zaawansowane - Gobuster**

**gobuster dir -u http://betulait.usermd.net -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt**

**# Skanowanie plików**

**gobuster dir -u http://betulait.usermd.net -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,html,txt,js**

**Oczekiwany rezultat: Brak dostępu do wrażliwych katalogów i plików**

**Testy Automatyczne**

**Opis Frameworka**

**Testy automatyczne zostały zaimplementowane przy użyciu Selenium WebDriver w języku Python. Framework zapewnia automatyzację kluczowych scenariuszy użytkownika.**

**Struktura Testów Automatycznych**

**TA-001: test\_login()**

**Cel: Automatyczna weryfikacja procesu logowania administratora**

**Implementacja:**

* **Otwiera stronę logowania**
* **Wprowadza dane administratora**
* **Weryfikuje przekierowanie na panel główny**

**TA-002: test\_create\_organization()**

**Cel: Automatyczne tworzenie nowej organizacji**

**Implementacja:**

* **Logowanie jako administrator**
* **Wypełnienie formularza organizacji losowymi danymi**
* **Zatwierdzenie utworzenia**

**TA-003: test\_edit\_organization()**

**Cel: Automatyczna edycja istniejącej organizacji**

**Implementacja:**

* **Wybór losowej organizacji z listy**
* **Edycja danych organizacji**
* **Zapisanie zmian**

**TA-004: test\_create\_user()**

**Cel: Automatyczna rejestracja nowego użytkownika**

**Implementacja:**

* **Wypełnienie formularza rejestracji**
* **Generowanie losowych danych użytkownika**
* **Przesłanie formularza**

**TA-005: accept\_user()**

**Cel: Automatyczne zatwierdzanie oczekujących użytkowników**

**Implementacja:**

* **Przejście do listy oczekujących zatwierdzeń**
* **Zatwierdzenie nowego użytkownika**

**TA-006: login\_and\_ticket()**

**Cel: Automatyczne tworzenie zgłoszenia przez klienta**

**Implementacja:**

* **Logowanie jako klient**
* **Utworzenie nowego zgłoszenia z losowymi danymi**
* **Wybór kategorii i zatwierdzenie**

**TA-007: login\_agent\_and\_przypisanie\_ticket()**

**Cel: Automatyczne przypisanie zgłoszenia przez agenta**

**Implementacja:**

* **Logowanie jako agent**
* **Znalezienie nieprzypisanego zgłoszenia**
* **Przypisanie zgłoszenia do siebie**

**TA-008: login\_admin\_and\_przypisanie\_ticket()**

**Cel: Automatyczne przypisanie zgłoszenia przez administratora**

**Implementacja:**

* **Logowanie jako administrator**
* **Znajdowanie zgłoszenia oczekującego na akcję**
* **Przypisanie do wybranego agenta**

**TA-009: login\_admin\_change\_status\_ticket()**

**Cel: Automatyczna zmiana statusu zgłoszenia**

**Implementacja:**

* **Znajdowanie zgłoszenia do zamknięcia**
* **Zmiana statusu na "zamknięte"**

**TA-010: login\_admin\_panel()**

**Cel: Automatyczne zarządzanie panelem administratora**

**Implementacja:**

* **Dostęp do panelu Django Admin**
* **Zarządzanie komentarzami i użytkownikami**
* **Blokowanie/odbłokowanie kont**

**TA-011: login\_client\_and\_comment\_to\_random\_ticket\_and\_logout()**

**Cel: Automatyczne dodawanie komentarzy i wylogowanie**

**Implementacja:**

* **Logowanie jako klient**
* **Wybór losowego zgłoszenia**
* **Dodanie komentarza**
* **Wylogowanie z systemu**

**Funkcje Pomocnicze**

**random\_string(length=8)**

**Generuje losowy ciąg znaków o określonej długości**

**random\_login(length=8)**

**Generuje losową nazwę użytkownika**

**random\_phone()**

**Generuje losowy numer telefonu w formacie polskim**

**Uruchamianie Testów Automatycznych**

**if \_\_name\_\_ == "\_\_main\_\_":**

**test\_login()**

**test\_create\_organization()**

**test\_edit\_organization()**

**test\_create\_user()**

**# accept\_user() # Odkomentować w razie potrzeby**

**login\_and\_ticket()**

**login\_agent\_and\_przypisanie\_ticket()**

**login\_and\_ticket()**

**login\_admin\_and\_przypisanie\_ticket()**

**login\_and\_ticket()**

**login\_admin\_change\_status\_ticket()**

**login\_admin\_panel()**

**login\_client\_and\_comment\_to\_random\_ticket\_and\_logout()**

**Środowisko Testowe**

**Konfiguracja**

* **URL testowy: http://betulait.usermd.net/**
* **Przeglądarka: Chrome WebDriver**
* **Python: Selenium WebDriver**
* **Framework testowy: Selenium**
* **System operacyjny testów bezpieczeństwa: Kali Linux 2023.x**
* **Narzędzia penetracyjne: Pełna kolekcja Kali Linux**

**Dane Testowe**

* **Administrator: admin / u9rKvvtfN(VtxjcHfFor**
* **Agent: agent1 / agent123**
* **Klient: client1 / client123**
* **Superagent: superagent / [hasło]**

**Wymagania Środowiska**

**Dla testów automatycznych:**

* **Selenium WebDriver**
* **ChromeDriver**
* **Python 3.x**
* **Stabilne połączenie internetowe**

**Dla testów bezpieczeństwa (Kali Linux):**

* **Skanowanie portów: nmap, masscan**
* **Testowanie aplikacji web: nikto, dirb, gobuster, wfuzz**
* **Ataki brute-force: hydra, medusa**
* **Analiza SSL/TLS: sslyze, testssl.sh**
* **Testowanie SQLi: sqlmap**
* **Proxy/Intercepting: Burp Suite, OWASP ZAP**
* **Analiza sieci: wireshark, ettercap, netdiscover**
* **Testowanie WiFi: aircrack-ng suite**
* **Słowniki: rockyou.txt, dirb wordlists, seclists**

**Konfiguracja Kali Linux:**

**# Aktualizacja systemu**

**sudo apt update && sudo apt upgrade -y**

**# Instalacja dodatkowych narzędzi**

**sudo apt install -y zaproxy burpsuite sqlmap**

**# Przygotowanie słowników**

**sudo gunzip /usr/share/wordlists/rockyou.txt.gz**

**Metodologia Testów**

**Podejście do Testowania**

1. **Testy funkcjonalne - weryfikacja wszystkich funkcjonalności systemu**
2. **Testy bezpieczeństwa - sprawdzenie odporności na ataki z wykorzystaniem Kali Linux**
3. **Testy automatyczne - automatyzacja kluczowych scenariuszy**
4. **Testy penetracyjne - symulacja rzeczywistych ataków**

**Metodologia Testów Bezpieczeństwa (Kali Linux)**

**Faza 1: Reconnaissance (Rozpoznanie)**

* **Skanowanie portów (nmap)**
* **Mapowanie sieci (netdiscover)**
* **Zbieranie informacji o domenie (whois, dig)**

**Faza 2: Scanning (Skanowanie)**

* **Skanowanie podatności (nikto, nmap scripts)**
* **Fuzzing aplikacji (wfuzz, gobuster)**
* **Analiza SSL/TLS (sslyze)**

**Faza 3: Enumeration (Wyliczanie)**

* **Identyfikacja technologii (whatweb)**
* **Skanowanie katalogów (dirb, gobuster)**
* **Analiza aplikacji (Burp Suite, ZAP)**

**Faza 4: Exploitation (Eksploitacja)**

* **Testowanie SQL Injection (sqlmap)**
* **Ataki brute-force (hydra)**
* **Testowanie XSS i CSRF**

**Faza 5: Post-Exploitation (Po eksploitacji)**

* **Analiza logów bezpieczeństwa**
* **Sprawdzenie eskalacji uprawnień**
* **Testowanie trwałości dostępu**

**Harmonogram Testów**

**Tydzień 1: Testy Funkcjonalne**

* **Dni 1-3: Testy modułów podstawowych**
* **Dni 4-5: Testy ścieżek użytkowników**

**Tydzień 2: Testy Bezpieczeństwa (Kali Linux)**

* **Dzień 1: Skanowanie i reconnaissance**
* **Dzień 2: Testowanie aplikacji webowych**
* **Dzień 3: Ataki brute-force i SQLi**
* **Dzień 4: Analiza SSL/TLS i sieci**
* **Dzień 5: Raportowanie i weryfikacja**

**Tydzień 3: Testy Automatyczne**

* **Dni 1-2: Implementacja testów Selenium**
* **Dni 3-4: Integracja z CI/CD**
* **Dzień 5: Optymalizacja i dokumentacja**

**Kryteria Akceptacji**

* **Wszystkie testy funkcjonalne muszą przejść pomyślnie**
* **Brak krytycznych podatności bezpieczeństwa**
* **Testy automatyczne działają stabilnie**
* **System spełnia wymagania wydajnościowe**

**Raportowanie Błędów**

1. **Identyfikacja błędu**
2. **Kroki reprodukcji**
3. **Oczekiwany vs rzeczywisty rezultat**
4. **Priorytet i wpływ na system**
5. **Środowisko testowe**
6. **Załączniki (screenshoty, logi)**

**Cykl Testowy**

1. **Planowanie - określenie zakresu testów**
2. **Projektowanie - tworzenie przypadków testowych**
3. **Wykonanie - uruchomienie testów**
4. **Raportowanie - dokumentacja wyników**
5. **Retesty - weryfikacja poprawek**

**Metryki Testowe**

* **Pokrycie testami (%)**
* **Liczba znalezionych błędów (krytyczne, wysokie, średnie, niskie)**
* **Czas wykonania testów**
* **Wskaźnik powodzenia testów**
* **Liczba przypadków testowych**
* **Metryki bezpieczeństwa (Kali Linux):** 
  + **Liczba zidentyfikowanych podatności**
  + **Poziom ryzyka podatności (CVSS score)**
  + **Skuteczność ataków brute-force**
  + **Czas potrzebny na kompromitację**
  + **Pokrycie testami penetracyjnymi**

**Raportowanie z Kali Linux**

**Narzędzia do generowania raportów:**

**# Nmap XML output**

**nmap -sV -oX nmap\_scan.xml target**

**# Nikto HTML report**

**nikto -h target -o nikto\_report.html -Format htm**

**# SQLmap raport**

**sqlmap --flush-session --batch --report**

**# ZAP HTML report**

**zap-cli report -o zap\_report.html -f html**

**Struktura raportu bezpieczeństwa:**

1. **Executive Summary - podsumowanie dla kierownictwa**
2. **Metodologia - użyte narzędzia i techniki**
3. **Znalezione podatności - szczegółowy opis z CVSS**
4. **Rekomendacje - sposoby naprawy**
5. **Załączniki - logi, screenshoty, PoC**