Testy Oprogramowania aplikacji „web-ssh-terminal”

1. **Środowisko Testowe**. Opis środowiska, na którym testowano aplikację:

Nazwa systemu operacyjnego	Microsoft Windows 10 Pro
Wersja	10.0.19045 Kompilacja 19045
Procesor	Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz, 3301 MHz, Rdzenie: 4, Procesory logiczne: 4
Typ systemu	x64-based PC
Karta graficzna	NVIDIA GeForce GT1030
Pamięć RAM4 DDR4	12 GB
Przeglądarka:	Google Chroom Wersja 120.0.6099.217
Python	3.11.4
Moduły python(requirement.txt):	asgiref==3.7.2 attrs==23.1.0 autobahn==23.6.2 Automat==22.10.0 cffi==1.16.0 channels==4.0.0 charset-normalizer==3.3.1 constantly==15.1.0 cryptography==41.0.5 daphne==4.0.0 distlib==0.3.7 Django==4.2.5 docopt==0.6.2 filelock==3.12.4 hyperlink==21.0.0 idna==3.4 incremental==22.10.0 numpy==1.26.1 platformdirs==3.11.0 pyasn1==0.5.0 pyasn1-modules==0.3.0 pycparser==2.21 pyOpenSSL==23.3.0 service-identity==23.1.0 six==1.16.0 sqlparse==0.4.4 Twisted==23.8.0 twisted-iocpsupport==1.0.4 txaio==23.1.1 typing_extensions==4.8.0 tzdata==2023.3 urllib3==2.0.7 virtualenv==20.24.5 zope.interface==6.1 django-bootstrap-v5==1.0.11 django-colorfield==0.11.0 django-components==0.31 django-encrypted-model-fields==0.6.5 paramiko==3.4.0 channels-redis==4.1.0 django-bootstrap5==23.4 django-bootstrap-input-group==1.0.1
Relacyjny silniki bazodanowy	SQLitle 3.44.0
Nierelacyjny silnik bazodanowy	Redis Stable (7.2), Web Vitals
Oprogramowanie Dodatkowe	Memurial – Redis DB for Windows - Version. 4.1.0. 4.1.0. Redis API. 7.2. 7.2. Node.js Puppeteer – Version 21.7.0
Wersje protokołów IP używane w sieci	IPv4, IPv6
Konto testowe w aplikacji	• **test**:**test** – użytkownik bez żadnych uprawnień

	• **sshOnly:ssh** – użytkownik z uprawnieniami tylko do funkcjonalności ssh • **noteOnly**:**note** – użytkownik z uprawnieniami tylko do funkcjonalności notatek • **admin**:**admin** – użytkowni ze wszystkimi uprawnieniami (w tym panel admina)
Ustawienia Środowiska Wykonawczego (Runtime)	python manage.py runserver --settings=web.settings 0.0.0.0:8000

2. **Wprowadzenie**. Opis głównych funkcjonalności aplikacji:

Opis funkcjonalności	**Implementacja**	**Uwagi**
Użytkownik może tworzyć wirtualne podstrony, tzw. „taby"…	Sesje można otworzyć poprzez kliknięcie przycisku który stworzy podgląd podstrony z wybraną sesją	Sesje można otworzyć równie dobrze wchodzą na link sesji (np. …/ssh/3290, note/233, gdzie cyfra to identyfikator sesji)
Użytkownicy jednocześnie mogą mieć otwarte wiele „tabów"	Wraz z otwarciem/dołączeniem do nowej sesji frontend dodaje nowy przycisk do panelu użytkownika	
Każdemu „tabowi" można zmienić nazwę/kolor	Podwójne kliknięcie pozwala na zmianę nazwy taba. Kolor można zmienić klikają prawym przyciskiem	Nazwa/kolor tabu zapisywana jest per użytkownik
Sesje notatek jak i połączeń SSH można udostępnić innym użytkownikom	Realizowane poprzez wejście na link udostępniony w „sharing center" (prawy przycisk na tab) lub przekazanie kodu sesji który można wpisac w panelu save session.	Pośrednikiem danych przesyłanych w sesji jest serwer. Dane NIE przechodzą bezpośrednio między użytkownikami
Użytkownicy mogą równocześnie pracować w tej samej sesji w czasie rzeczywistym	Komunikacja odbywa się za pomocą websocketów	Czas odpowiedzi jest uzależniony od szybkości odpowiedzi zdalnego hosta z którym się łączymy.
Udostępnianie pozwala na współdzielenie tego samego outputu	Dane outputu sesji przekazywane są jednocześnie przez WebSocket dla konkretnej grupy użytkowników.	Input przesyłany od użytkowników przesyłany jest osobno do serwera, a następnie scalany i przesyłany do zdalnego hosta
Administrator może zarządzać zasobami serwera	Aplikacja pozwala nam skonfigurować ilość sesji którą może maksymalnie otworzyć użytkownik sesji jednocześnie z podziałem na sesje notatek i sesje ssh	Konfiguracja może być zmieniona w pliku setting.py
Użytkownicy mogą otwierać nowe sesje	Otwarcie sesji następuje poprzez wypełnienie krótkiego formularza z danymi dotyczącymi sesji.	Podstawowe dane które można skonfigurować przed otworzeniem sesji to: kolor/nazwa taba i flaga mówiąca o tym czy sesja będzie współdzielona.
W przypadku błędu uwierzytelnienia strona pozwoli na ponowne wpisanie danych logowania po kliknięciu przycisku „reconnect".	Przycisk otwiera formularz który pozwala na bezpiecznie przesłanie informacji uwierzytelniającej.	System ten został zaimplementowany stworzony. aby zapobiec użytkownikom możliwość wykradnięcia naszych danych logowania poprzez przechwycenie danych udostępnionych w sesji
Automatyczne zwalnianie zasobów serwera	Gdy użytkownicy nie korzystają z połączeń SSH te automatycznie się zamykają w celu zmniejszenia zużycia zasobów serwera	Po ponownym wejściu na stronę sesje te otworzą się ponownie.
Dane sesji ssh są automatycznie zapisywane	Dane sesji są przechowywane w buforze i zapisywane po przekroczeniu przyjętego ograniczenia. Również podczas rozłączenia jednego z użytkowników sesji dane zapisywane są do bazy. Po ponownym otwarciu sesji terminal wyświetli poprzednie dane jak i pozwoli na wpisanie nowych	Bufor został zaimplementowany w celu optymalizacji i zmniejszeniu laga podczas korzystania z konsoli.
Dane sesji notatek są automatycznie zapisywane	Dane zapisywane są na bieżąco podczas przetwarzania wiadomości websocketów przez serwer. Aplikacja zapisuje zawartość notatki w formie JSON w formacie delta.	Obecnie aplikacja pobiera całą zawartość notatki i zapisuje ją w bazie. Natomiast uniwersalność kodu pozwala na usprawnienie tego aspektu i dodawanie jedynie zmian korzystając z formatu delta.

Funkcjonalność notatek pozwala na edycję tekstu	Na notatkami jest pasek z narzędziami, który służy do edycji tekstu opisanej w DIW	
Aplikacja pozwala na tworzenie kont oraz przypisywania uprawnień do nich	Konto mogą być wyłącznie przez administratora aplikacji oraz osoby posiadające odpowiednie uprawnienia w panelu admina.	Sesjami, uprawnieniami, kontami można sterować z panelu administratora.
Aplikacja oferuje funkcjonalność szybkiego łączenia się z zdalnymi urządzeniem na podstawie wcześniej zapisanych danych	Podczas tworzenie sesji ssh można kliknąć przycisk „Save and Open" który zapisze dane naszej sesji do bazy danych.	Aby ponownie użyć tej sesji możemy kliknąć przycisk „Open" w panelu „My saved hosts"

3. Cel testów - znalezienie błędów w funkcjonalności w aplikacji:

Poniższe testy zostały przeprowadzone tak, aby narazić aplikację na wejście w stan który nie został początkowo uwzględniany w projekcie aplikacji lub został pominięty podczas jej implementacji. Po każdym przeprowadzonym teście zostaną podane wnioski oraz możliwe lub już zaimplementowane rozwiązania.

Opis przeprowadzonego testu	**Input**	**Output**	**Aktualnie rozwiązanie**	**Proponowane możliwe/alternatywne rozwiązania**	**Uwagi**
Udostępnienie sesji samemu sobie z użyciem linku	Przetestowano dla kont: sshOnly, noteOnly, admin	Po wejściu na link wyświetla się komunikat *„Session already joined!"* . „Tab" udostępnionej sesji zostaje otarty.	Użytkownik nie może dołączyć do sesji którą sam udostępnia. Efekt takie działanie zostaje pomijany przez aplikacje	- W przypadku dołączenia do takiej sesji interfejs graficzny może otworzyć nowy tab z dokładnie tym samym outputem sesji, dzięki czemu użytkownik będzie mógł stworzyć wiele okien które pokazują ten sam output	
Udostępnienie sesji samemu sobie z użyciem klucza	Przetestowano dla kont: sshOnly, noteOnly, admin	Po wpisaniu klucza sesji do inputu: pasek boczy się zamyka. „Tab" udostępnionej sesji zostaje otarty.	Użytkownik nie może dołączyć do sesji którą sam udostępnia. Efekt takie działanie zostaje pomijany przez aplikacje		W tym wypadku użytkownik nie jest informowany o tym, że jest już dołączony do tej sesji
Wejście na link udostępnionej sesji notatk/ssh przez konto bez uprawnień do podanej funkcjonalności	Przetestowano na koncie: test	Po wejściu na link wyświetla się komunika *„You have locked access to this functionality. Please contact administrator to recive access to this feature!!"*	Użytkownik nie może dołączyć do sesji ssh/notatke jeśli jego kontu nie nadano odpowiednich uprawnień		Domyślnie użytkownicy nie mają żadnych uprawnień. Więc nie mogą korzystać z sesji ssh, ani z sesji notatek.
Próba dołączenia do sesji za pomocą klucza przez konto bez uprawnień do podanej funkcjonalności	Przetestowano na koncie: test	Po wpisaniu złego klucz w input: pasek boczny się nie zamyka, wpisana wartość zostaje usunięta	Aplikacja ignoruje klucz wpisany przez użytkownika	W przypadku wpisywania niepoprawnego klucza użytkownik nie zostaje poinformowany prze GUI o tym, że nie ma uprawnień żeby dołączyć do tej sesji	
Wpisanie niepoprawnych wartości klucza do zakładki „My saved hosts"	Przetestowano na koncie: sshOnly, noteOnly, admin	- Po wejściu na link – wyświetla się alert „Wrong Url". - Po wpisaniu złego klucz w input: pasek	Niepoprawny link/klucz dostępu są pomijane przez aplikacje i nie wyrzucają błędów	W przypadku wpisywania niepoprawnego klucza użytkownik nie zostaje poinformowany prze GUI on tym, że klucz jest zły	

		boczny się nie zamyka, wpisana wartość zostaje usunięta	związanych z aplikacją		
Odebranie praw do korzystania z funkcjonalności sesji ssh/note użytkownikowi posiadającemu otwarte, zapisane sesje oraz sesje które współdzieli	Przetestowano na koncie: admin	Po kliknięciu na taby z oknami sesji wyświetla się komunikat „Error 403” Formularz tworzenia nowej sesji wyświetla *„This account have locked access to use ssh/note functionalities. Please contact app administrator to receive access to specific functionalities. ”* Zakładka my save hots po odświeżeniu jest zablokowane. Otwartych sesji nie można zamknąć	Okna załadowane po stronie użytkowników zostają otwarte do momentu odświeżenia strony. Użytkownik podczas próby otwarcia nowych okien sesji dostaje komunika „403”. Od momentu odebrania mu praw nie może wykonywać żadnej operacji na zapisanych sesjach po stronie serwera (po stronie fronendu może zmieniać kolor taba) w tym nie może zamykać otwartych sesji	1.Odebranie praw może nastąpić dopiero gdy usuniemy/zamkniemy wszystkie sesje użytkownika 2.W momencie odebrania praw wszystkie sesje użytkownika zostają zamknięta wraz z sesjami współdzielonymi. 3. Pozbycie się możliwość odebrania praw do danych funkcjonalności aplikacji	Aktualnie rozwiązanie daje administratorom sesji przejęcie kontroli nad sesją poprzez dołączenie się do niej i zablokowanie użytkownika który ja otworzył
Wprowadzenie niepoprawnych danych w formularzu tworzenia sesji SSH	Przetestowano na koncie: sshOnly, noteOnly, admin.	Niepoprawnie wprowadzone dane w formularzu zostają obramowane na czerwono. Wszystkie poprawne pola zostają obramowane na zielono.	Formularz jest automatycznie sprawdzany pod kątem wymaganych poł: nr portu, ip lub hostname, nazwa sesji. Np. w przypadku nie wprowadzenia nazwy sesji wyświetla „This field is required. Session name can not be empty”		
Stworzenie sesji po przekroczeniu maksymalnej liczby sesji na użytkownika	Przetestowano na koncie: sshOnly, noteOnly, admin.	- Strona zwróci Internal Server Error	Aplikacja blokuje zapytanie o utworzenie nowej sesji		

4. **Testy wydajnościowe**.

Poniższe testy wydajnościowa oparte są o rozszerzenie „Web Vitals” dostępnego na przeglądarce Google Chrome. Mierzyliśmy nimi, ile zajmuje ładowanie się poszczególnych podstrona aplikacji. Wyniki przeprowadzone poniżej są średnią wyciągniętą z serii testów dla każdej podstrony (dla każdej średnia z 10 testów). Testy przeprowadzono na lokalnej maszynie (parametry zadane wyżej) na czystym serwerze z jednym użytkownikiem. W naszej opinii testy ładowania podstrony są dość ważne ze

względu na fakt, iż ~70% frontendu aplikacji generowana jest poprzez JavaScript co znacząco może spowolnić działanie aplikacji.

Krótki opis parametrów:

- **LCP** (Largest Contentful Paint) - skupia się na czasie, jaki potrzebny jest na wyświetlenie największego elementu treści na stronie
- **FCP** (First Contentful Paint) - mierzy czas od rozpoczęcia nawigacji do chwili, gdy przeglądarka renderująca pierwszy element treści na stronie
- **TTFB** (Time to First Byte) mierzy, ile upływa od momentu wysłania żądania HTTP przez przeglądarkę do serwera, do momentu otrzymania pierwszego bajtu odpowiedzi od tego serwera
- **CLS** (Cumulative Layout Shift) - ma na celu pomiar stabilności układu strony internetowej. Oznacza to, jak bardzo elementy strony przesuwają się w trakcie jej ładowania. Metryka ta bierze pod uwagę wszelkie zmiany wizualne, które mogą wpłynąć na doświadczenie użytkownika.
- **FID** (First Input Delay) - czas, jaki upływa od momentu, gdy użytkownik wykonuje pierwszą interakcję (na przykład kliknięcie, naciśnięcie klawisza) na stronie do chwili, gdy przeglądarka reaguje na tę interakcję

Strona logowania *„login/”:*

- LCP 64 ms
- FCP 64 ms
- TTFB 10 ms
- CLS 0.00 ms
- FID: 1 ms

Strona logowania *„terminal/” dla 1 otwartych sesji **SSH**:*

- LCP 179 ms
- FCP 179 ms
- TTFB 13 ms
- CLS 0.00 ms
- FID: 1 ms

Strona logowania *„terminal/” dla 10 otwartych sesji **SSH**:*

- LCP 214 ms
- FCP 214 ms
- TTFB 48 ms
- CLS 0.00 ms
- FID: 4 ms

Strona logowania *„terminal/” dla 1 otwartych sesji **notatek**:*

- LCP 268 ms
- FCP 215 ms

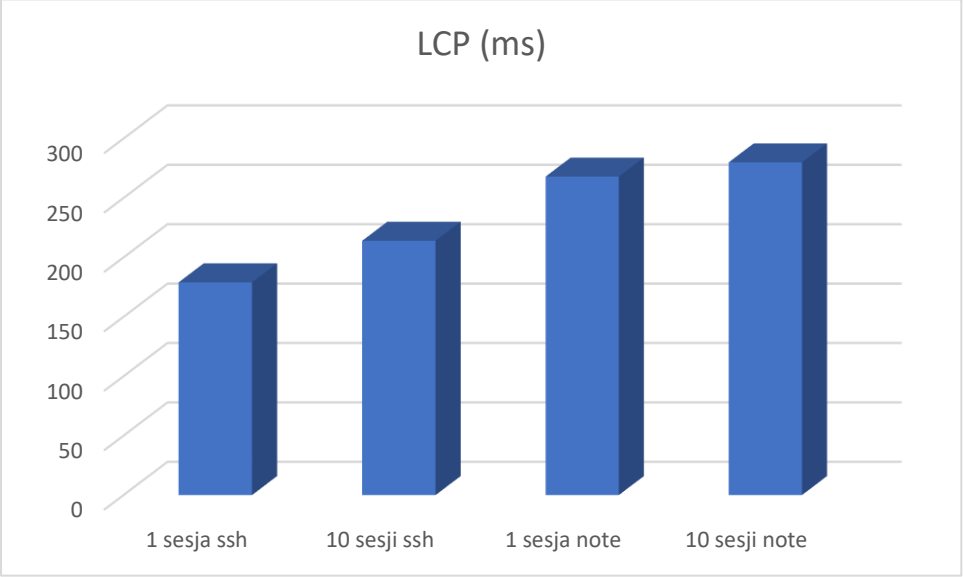
- TTFB 13 ms
- CLS 0.00 ms
- FID: 2 ms

Strona logowania *„terminal/” dla 10 otwartych sesji* ***notatek****:*

- LCP 280 ms
- FCP 240 ms
- TTFB 20 ms
- CLS 0.00 ms
- FID: 4 ms

Powyższe testy dla sesji notatek/ssh przeprowadzone na sesjach z bez kontentu – oznacza to, że sprawdzany był czas potrzebny wyłączny na załadowanie samej funkcjonalności, a nie jest przesłania dodatkowych zapisanych informacji przez użytkownika.

Strona Logowania	Sesje SSH	LCP (ms)	FCP (ms)	TTFB (ms)	CLS (ms)	FID (ms)
terminal/	1	179	179	13	0.00	1
terminal/	10	214	214	48	0.00	4
terminal/	1	268	215	13	0.00	2
terminal/	10	280	240	20	0.00	4



Wyniki testów wydajnościowych sugerują, że nasza aplikacje działa optymalnie, wspiera szybkie ładowanie podstron i zapewnia stabilność układu strony. Na podstawie parametru FID jesteśmy wstanie stwierdzić, że interakcja użytkownika ze stroną mieści się w akceptowalnych normach (0-10ms), co oznacza, że interakcje użytkownika są szybko obsługiwane.

5. **Testy wydajnościowe serwera**:

W ramach testów wydajnościowych serwera przeprowadzimy ocenę zdolności serwera do obsługi różnych obciążeń związanych z sesjami SSH. Celem jest zrozumienie, jak serwer radzi sobie z utrzymaniem stabilnego i efektywnego działania przy różnych poziomach obciążenia. Przeprowadzimy testy, w których sprawdzimy, ile zasobów serwer utrzymuje przy jednoczesnym działaniu 0, 1, 2, 10, 20, 50 oraz 100 sesji SSH. Ostateczne wyniki pomogą w zidentyfikowaniu potencjalnych punktów krytycznych oraz dostarczą informacji na temat optymalnych ustawień serwera dla różnych obciążeń. Wyniki przeprowadzone poniżej są średnią wyciągniętą z zużycia zasobów komputera w ciągu jednej minut dla każdego testu. Przed rozpoczęciem każdego testu odczekano 15 sekund w celu ustabilizowania się działania aplikacji (czas na otwarcie połączenia do zdalnego hosta), a następnie wpisano przesłano do terminala 100 losowych znaków. Poniższa statystyka nie wlicza zasobów zużytych przez program Memurial (Redis For windows)

Ilość sesji ssh	Zużycie ramu [MB]	Zużycie procesora [%]
0	128.2	1.7
1	129.2	3.4
2	130.8	3.8
10	138.4	9.1
20	139,6	17
50	144.3	34.6
100	152.9	46

6. **Wnioski**:

Serwer prezentuje niskie zużycie zasobów w przypadku braku sesji SSH, co sugeruje efektywne zarządzanie zasobami w spoczynku. Wprowadzenie danych do sesji początkowo generuje skokowe zwiększenie zużycia zasobów, jednak z czasem (>60s od nadania sekwencji) obserwujemy jego spadek do stanu zużycia bliski 0 sesji ssh. Może to wskazywać, że w środowisku produkcyjnym, gdzie wielu użytkowników nie korzysta jednocześnie z aplikacji, zużycie zasobów znacznie zmaleje. Nieliniowe zwiększanie się zużycia zasobów jest rezultatem zastosowanych optymalizacji w kodzie. Ostatecznie, serwer efektywnie radzi zwraca odpowiedzi w czasie do 1s dla liczby sesji poniżej ~50. Przekroczenie tego progu generuje wydłużone czasy oczekiwania (>2s na sesję) na odpowiedź z zdalnego hosta.

Testy przeprowadzone przez:
Krzysztof Stefański,
Mateusz Setkowicz
WIET, Teleinformatyka III