

**WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI**
POLITECHNIKI RZESZOWSKIEJ

Eryk Delikat

Menadżer haseł

Projekt inżynierski

Opiekun projektu:
dr inż. Grzegorz Drahus

Rzeszów, 2024

Spis treści

1.	Wprowadzanie	5
2.	Przegląd dostępnych rozwiązań	6
2.1.	Menedżer haseł Google Chrome.....	6
2.2.	Menedżer haseł 1Password.....	8
3.	Cel i zakres projektu	9
3.1.	Cel projektu	9
3.2.	Główne założenia projektu.....	9
3.3.	Zadania wyznaczone do wykonania	10
3.4.	Zakres prac	11
4.	Hasła oraz szyfrowanie	12
4.1.	Zasady Wyboru Haseł.....	12
4.2.	Analiza zagrożeń	13
4.2.1.	Metody złamania haseł	13
4.2.2.	Czynniki zwiększające ryzyko	13
4.2.3.	Konsekwencje złamania haseł.....	14
4.2.4.	Strategie obronne	14
4.3.	Algorytm szyfrujący zaimplementowany w projekcie.....	14
4.4.	Zastosowanie Soli i Wektora Inicjalizującego w Procesach Szyfrowania	17
5.	Wybrane narzędzia.....	17
5.1.	ASP.NET w środowisku Microsoft Visual Studio.....	17
5.2.	Firebase	19
5.3.	C#.....	21
5.4.	Javascript	22
5.5.	HTML oraz CSS.....	23
5.6.	Bootstrap	24
5.7.	Baza danych NoSQL.....	24
6.	Opis funkcjonalności aplikacji internetowej.....	27
6.1.	Rejestracja i logowanie.....	27
6.2.	Dodawanie, wyświetlanie, aktualizacja oraz usuwanie haseł	29
6.3.	Generowanie i ocena haseł	33
6.4.	Diagram przypadków użycia.....	38
7.	Podsumowanie	41
	Literatura	43

1. Wprowadzanie

Dynamiczny rozwój technologii informacyjnych oraz coraz większa cyfryzacja życia codziennego prowadzą do wzrostu liczby poświadczeń, które użytkownicy muszą zarządzać w celu dostępu do różnorodnych usług cyfrowych. W dobie cyfryzacji, gdzie średnio użytkownik internetu może posiadać konta na dziesiątkach różnych platform od mediów społecznościowych, przez portale bankowe, sklepy online po menedżery haseł stają się kluczowym elementem w utrzymaniu bezpieczeństwa cyfrowego. Tym samym, bezpieczeństwo tychże poświadczeń staje się kwestią o kluczowym znaczeniu dla ochrony prywatności oraz zapewnienia integralności danych osobowych i firmowych.

Menedżery haseł, jako narzędzia służące do przechowywania i organizacji haseł, muszą odpowiadać na szereg współczesnych wyzwań. W kontekście naukowym, problematyka ta obejmuje zagadnienia z dziedziny kryptografii, bezpieczeństwa informacji oraz interakcji człowiek-komputer. Analiza literatury oraz istniejących rozwiązań ujawnia, że największe trudności pojawiają się w dwóch obszarach: zapewnienie wysokiego poziomu bezpieczeństwa poświadczeń przy jednoczesnym zachowaniu intuicyjnego i prostego w użyciu interfejsu użytkownika. Rozwój nowego menedżera haseł wymaga zatem nie tylko zastosowania zaawansowanych technologii szyfrowania, ale również innowacji w dziedzinie projektowania interfejsów użytkownika, aby sprostać oczekiwaniom i potrzebom użytkowników o różnym stopniu zaawansowania technicznego.

Głównym zadaniem menedżera haseł jest zabezpieczenie dostępu do wrażliwych danych użytkownika poprzez ich szyfrowanie. Użytkownik musi pamiętać tylko jedno główne hasło, tzw. „master password”, które daje dostęp do zaszyfrowanej bazy danych zawierającej wszystkie inne hasła. Dzięki temu nie musi on pamiętać wielu skomplikowanych haseł, co znacznie obniża ryzyko wykorzystania przez osoby trzecie prostych lub wielokrotnie używanych haseł. Dodatkowo, funkcja automatycznego wypełniania pozwala na błyskawiczne logowanie do serwisów internetowych, jednocześnie minimalizując ryzyko pomyłki przy wprowadzaniu danych.

Menedżery haseł oferują również możliwość generowania silnych, losowych haseł, co jest szczególnie ważne w kontekście zapobiegania atakom polegającym na zgadywaniu haseł. Umożliwia również synchronizację haseł między różnymi urządzeniami, co jest nieocenione w świecie, gdzie równocześnie korzystamy z komputerów stacjonarnych, laptopów, tabletów i smartfonów.

Zapominanie haseł to powszechny problem, który może prowadzić do frustracji, straty czasu, a nawet utraty danych lub dostępu do usług. Osoba pracująca w branży IT może być zmuszona do zarządzania setkami haseł, jeśli zliczymy hasła prywatne oraz służbowe, co sprawia, że praktycznie niemożliwym jest pamiętać wszystkie z nich, szczególnie te używane rzadziej. Oczywiście nie bierzemy pod uwagę takiej możliwości, aby hasła w różnych usługach się powielały. Jest to bardzo niebezpieczne podejście, ponieważ gdy w jednej platformie dojdzie do wycieku danych to osoba mająca dostęp do udostępnionych danych może zalogować się na wszystkie konta, w których jest takie samo hasło. Menedżer haseł eliminuje problem związany z pamiętaniem dziesiątek danych logowania, oferując jedno, centralne miejsce do przechowywania i zarządzania wszystkimi poświadczeniami. To znacząco upraszcza życie cyfrowe użytkownika, jednocześnie podnosząc poziom jego cyfrowego bezpieczeństwa.

2. Przegląd dostępnych rozwiązań

2.1. Menedżer haseł Google Chrome

Menedżer haseł Google Chrome jest wbudowaną funkcją przeglądarki, która automatycznie zbiera, przechowuje i wypełnia poświadczenia użytkownika ze stron internetowych. System ten jest zintegrowany z przeglądarką, co zapewnia wysoką dostępność i łatwość użytkowania. Przechowywanie danych odbywa się zarówno lokalnie na urządzeniu użytkownika, jak i w chmurze za pośrednictwem konta Google, co umożliwia synchronizację haseł między różnymi urządzeniami.

Google Chrome stosuje szereg technik zabezpieczających przechowywane hasła. Hasła są szyfrowane przy użyciu silnych algorytmów, takich jak AES, co stanowi standard w branży bezpieczeństwa cyfrowego. Klucz do deszyfrowania haseł jest zabezpieczony w ramach zarządzanego przez użytkownika konta Google, co wymaga uwierzytelnienia dwuskładnikowego dla dodatkowego poziomu bezpieczeństwa, lecz kiedy użytkownik jest zalogowany domyślnie nic nie chroni zapisanych w nim haseł.

Menedżer haseł umożliwia łatwe zarządzanie poświadczeniami dzięki funkcjom takim jak generowanie silnych haseł, automatyczne wypełnianie formularzy oraz powiadomienia o hasłach, które były w wyciekach danych. Funkcja ta poprawia ogólną higienę haseł poprzez zniechęcanie do ponownego używania tych samych haseł na różnych stronach internetowych.

Przedsiębiorstwo Google, jako globalny gigant technologiczny, angażuje się w rozwijanie i oferowanie usług wysokiej jakości, odpowiadając na dynamiczne potrzeby rynku cyfrowego. Jednakże, pomimo zastosowania nowoczesnych metod szyfrowania i innych

mechanizmów ochrony, istniejące rozwiązanie menedżera haseł nie spełnia pełni standardów wymaganych dla zapewnienia optymalnego poziomu bezpieczeństwa. Brak zaawansowanych funkcji zarządzania i potencjalne ryzyko naruszeń prywatności stanowią wyzwania, które mogą podważać wiarygodność tej usługi. W związku z tym, mimo starań Google o dostarczanie rozwiązań o wysokiej jakości, aspekt bezpieczeństwa ich menedżera haseł pozostaje obszarem, który wymaga dalszych udoskonaleń w celu osiągnięcia poziomu bezpieczeństwa adekwatnego do współczesnych zagrożeń cyfrowych. W poniższych akapitach zostaną poruszone główne problemy dotyczące opisywanego rozwiązania od firmy Google.

Korzystanie z menedżera haseł wbudowanego w przeglądarkę Google Chrome prowadzi do zależności od jednego dostawcy – Google. To oznacza, że wszystkie poświadczenia są zarządzane przez jedną firmę, co może stanowić ryzyko, jeśli dojdzie do naruszenia bezpieczeństwa w infrastrukturze Google. Ponadto, jeśli użytkownik zdecyduje się zmienić przeglądarkę, migracja zapisanych haseł do innego menedżera może być trudna lub niemożliwa.

Chociaż hasła są szyfrowane, ich przechowywanie w chmurze zawsze wiąże się z pewnym ryzykiem. Ataki typu man-in-the-middle, naruszenia danych na serwerach Google, czy inne techniki wykorzystujące luki w bezpieczeństwie mogą prowadzić do nieautoryzowanego dostępu do danych użytkowników. Użytkownicy muszą polegać na środkach bezpieczeństwa stosowanych przez Google, na które nie mają bezpośredniego wpływu.

W porównaniu z dedykowanymi menedżerami haseł, Google Chrome oferuje ograniczone funkcjonalności. Na przykład, brakuje zaawansowanych opcji zarządzania takich jak kategorie haseł, notatki bezpieczne, czy przechowywanie innych wrażliwych informacji oprócz haseł. Nie ma też wsparcia dla bardziej skomplikowanych schematów udostępniania haseł między użytkownikami.

Jako produkt firmy Google, menedżer haseł może być postrzegany jako narzędzie do gromadzenia danych o użytkownikach w celach reklamowych. Chociaż Google twierdzi, że nie wykorzystuje danych haseł do celów reklamowych, sam fakt przetwarzania i przechowywania dużej ilości danych użytkowników może budzić obawy dotyczące prywatności.

Google Chrome wykorzystuje do deszyfrowania haseł konto Google użytkownika. Jeśli ktoś uzyska dostęp do tego konta, automatycznie ma dostęp do wszystkich zapisanych haseł. Oznacza to, że bezpieczeństwo wszystkich haseł jest tak silne, jak najłatwiejsze do złamania hasło lub metoda uwierzytelniania użyta do zabezpieczenia konta Google.

Podsumowując, menedżer haseł Google Chrome jest narzędziem, które zyskuje na popularności ze względu na swoją integrację z przeglądarką, co przekłada się na znaczną wygodę i łatwość użycia. Użytkownicy cenią sobie możliwość bezproblemowego przechowywania i automatycznego wypełniania haseł na różnych urządzeniach z dostępem do Internetu. Niemniej jednak, ten system zarządzania hasłami posiada istotne ograniczenia, które mogą wpływać na poziom bezpieczeństwa oraz ochronę prywatności osób z niego korzystających.

2.2. Menedżer haseł 1Password

1Password nie oferuje darmowej wersji swojego produktu, co może być barierą dla użytkowników indywidualnych lub małych firm, które nie chcą ponosić miesięcznych lub rocznych kosztów. O ile firma oferuje wersje próbne, długoterminowe korzystanie z menedżera wiąże się z koniecznością zakupu płatnej subskrypcji.

Chociaż 1Password jest chwalony za swoje bogate funkcjonalności, jego interfejs i liczne opcje konfiguracji mogą być początkowo przytłaczające dla nowych użytkowników. Osoby, które wcześniej nie korzystały z zaawansowanych menedżerów haseł, mogą potrzebować czasu na adaptację i naukę obsługi 1Password.

Podobnie jak w przypadku innych menedżerów haseł, użytkownicy 1Password polegają na jednej firmie w kwestii bezpieczeństwa i przechowywania swoich danych. W przypadku naruszenia danych firmy lub awarii, użytkownicy mogą być narażeni na utratę dostępu do swoich haseł lub nawet ich wyciek.

Chociaż 1Password generalnie oferuje dobre wsparcie dla różnych systemów operacyjnych i przeglądarek, mogą występować problemy z integracją w mniej popularnych aplikacjach lub specyficznych środowiskach korporacyjnych. Niektórzy użytkownicy zgłaszali trudności z automatycznym wypełnianiem haseł w niektórych aplikacjach czy systemach.

Wersje próbne 1Password, chociaż użyteczne, mają ograniczenia funkcjonalne, co może nie dać pełnego obrazu możliwości produktu przed dokonaniem zakupu. To może być frustrujące dla użytkowników, którzy chcą dokładnie przetestować produkt przed zobowiązaniem się do subskrypcji.

Podsumowując, mimo że 1Password jest zaawansowanym narzędziem, które zwiększa bezpieczeństwo cyfrowe, nie jest wolne od wad, które mogą wpływać na decyzję o jego wyborze. Potencjalni użytkownicy powinni rozważyć te ograniczenia, szczególnie w kontekście ich specyficznych potrzeb i oczekiwań co do menedżera haseł.

3. Cel i zakres projektu

3.1. Cel projektu

Celem projektu jest stworzenie bezpiecznego menadżera haseł, który funkcjonuje jako niezależna aplikacja, wolna od wpływów i kontroli dużych korporacji technologicznych. Menadżer ten ma na celu zapewnienie użytkownikom pełnej kontroli nad ich danymi uwierzytelniającymi przy jednoczesnym zwiększeniu ich bezpieczeństwa za pomocą zaawansowanych technologii szyfrowania. Projekt ma na celu rozwiązanie problemów związanych z bezpieczeństwem danych osobowych i zapewnienie transparentności działania systemu, aby użytkownicy mogli zrozumieć i zaufać mechanizmom ochrony ich informacji. Inicjatywa ta wpisuje się w rosnącą potrzebę ochrony prywatności w cyfrowej przestrzeni, oferując rozwiązanie, które jest zarówno skuteczne, jak i dostępne dla szerokiej grupy użytkowników.

3.2. Główne założenia projektu

- 1) Niezależność od dużych platform. Menadżer haseł będzie funkcjonował jako samodzielna aplikacja, niezależna od ekosystemów i infrastruktury dużych firm technologicznych. Ta niezależność zapewni użytkownikom większą kontrolę nad ich danymi oraz zmniejszy ryzyko nadużyć i wycieków danych.
- 2) Zastosowanie zaawansowanych technologii szyfrowania. Projekt zakłada implementację najnowszych metod szyfrowania, w tym szyfrowania end-to-end, co oznacza, że klucze do deszyfrowania danych będą znajdować się wyłącznie w posiadaniu użytkownika. Dzięki temu, nawet w przypadku przejęcia danych przez nieautoryzowane osoby, informacje te pozostaną zabezpieczone.
- 3) Interoperacyjność i łatwość użytkowania. Menadżer będzie kompatybilny z różnymi systemami operacyjnymi i platformami, zapewniając łatwość integracji i użytkowania. Interfejs użytkownika zostanie zaprojektowany tak, by był intuicyjny nawet dla osób, które nie posiadają zaawansowanej wiedzy technologicznej.

- 4) Otwarta architektura i audytowalność. Kod źródłowy aplikacji będzie dostępny publicznie, co umożliwi niezależne audyty i recenzje, zwiększając tym samym zaufanie użytkowników oraz społeczności technologicznej. Otwartość projektu ma także na celu zachęcenie społeczności do współtworzenia i ciągłego doskonalenia menadżera.
- 5) Wykonywanie kodu po stronie użytkownika. Większość funkcjonalności jest realizowana bezpośrednio po stronie użytkownika, co zwiększa bezpieczeństwo i prywatność danych. Do tych celów wykorzystywany jest JavaScript, język skryptowy działający na urządzeniu końcowym. Dzięki temu, operacje takie jak szyfrowanie i deszyfrowanie danych, generowanie haseł czy weryfikacja danych wejściowych odbywają się lokalnie, bez konieczności przesyłania wrażliwych informacji do serwera. Ta architektura zapewnia, że kluczowe operacje są wykonywane w bezpiecznym środowisku użytkownika, co minimalizuje ryzyko przechwycenia danych przez nieuprawnione osoby podczas transmisji do i z serwera.

3.3. Zadania wyznaczone do wykonania

Ukończenie projektu oraz osiągnięcie wszystkich celów wymaga, zrealizowania szeregu poniższych zadań.

- 1) Przegląd istniejących rozwiązań menadżerów haseł na rynku, aby zidentyfikować ich mocne i słabe strony.
- 2) Zbieranie potrzeb użytkowników.
- 3) Określenie podstawowych funkcjonalności aplikacji w postaci diagramu przypadków
- 4) Wybór technologii (języki programowania, baza danych)
- 5) Określenie architektury systemu
- 6) Definiowanie strategii szyfrowania danych, autentykacji użytkowników oraz innych mechanizmów bezpieczeństwa.
- 7) Projektowanie interfejsu użytkownika [14]
- 8) Implementacja aplikacji internetowej
- 9) Systematyczna analiza aplikacji, czy bieżąca wersja spełnia założenia i cele
- 10) Dokumentacja techniczna

3.4. Zakres prac

Całość projektu menadżera haseł to złożone przedsięwzięcie technologiczne, które obejmuje nie tylko rozwój oprogramowania, ale również skrupulatne planowanie oraz testy bezpieczeństwa. Projekt ten wymaga interdyscyplinarnego podejścia i połączenia wiedzy z różnych dziedzin, takich jak bezpieczeństwo IT, inżyniera oprogramowania, UX/UI, oraz zarządzanie projektem. Celem jest stworzenie nie tylko funkcjonalnego, ale i bezpiecznego narzędzia, które zabezpieczy wrażliwe dane użytkowników przed zagrożeniami, jednocześnie będąc łatwym w obsłudze i dostępnym na różnych platformach.

Główne etapy wytwarzania oprogramowania

- 1) analiza wymagań
- 2) projektowanie systemu
- 3) implementacja
- 4) testowanie
- 5) wdrożenie
- 6) eksploatacja i utrzymanie

Pierwszym i jednocześnie kluczowym etapem jest w procesie tworzenia oprogramowania jest analiza wymagań. Zespół projektowy (w tym przypadku jednoosobowy) pracuje w ścisłej współpracy z użytkownikami końcowymi, aby dokładnie zrozumieć ich potrzeby i oczekiwania. Celem tego etapu jest zebranie kompleksowych informacji, które posłużą do stworzenia dokładnej specyfikacji funkcjonalnej i technicznej produktu.

Po zebraniu wymagań, następuje faza projektowania systemu, gdzie na podstawie zebranych danych tworzone są schematy, modele i prototypy oprogramowania. Definiuje architekturę systemu, wybierają stos technologiczny, a także planują interfejsy użytkownika i interakcje między modułami systemu. Dokumentacja z tej fazy stanowi fundament dla deweloperów.

Następnie z dokumentacją projektową przechodzi się do fazy implementacji, gdzie programista piszą kod zgodnie z zaplanowanymi specyfikacjami. Praca jest zwykle podzielona na moduły lub komponenty, co pozwala na równoległe programowanie i efektywniejsze zarządzanie projektem. Całość projektu powinna być podpięta do systemu kontroli źródeł. Najczęściej stosowaną platformą jest GitHub.

Testowanie jest nieodłącznym elementem procesu tworzenia oprogramowania. Tester używa różnych metod i narzędzi do wykrywania błędów i niedociągnięć w kodzie. Testy obejmują sprawdzanie funkcjonalności, wydajności, bezpieczeństwa oraz kompatybilności

systemu. Celem tego etapu upewnić się, że oprogramowanie działa zgodnie z oczekiwaniami i jest wolne od krytycznych błędów.

Gdy oprogramowanie jest już przetestowane i gotowe, następuje jego wdrożenie. Może to odbywać się etapami (wdrożenie fazowe) lub jednorazowo dla wszystkich użytkowników. W tym czasie szczególnie ważne jest, aby monitorować system pod kątem wszelkich problemów, które mogą wystąpić podczas przejścia z środowiska testowego do produkcyjnego.

Po wdrożeniu systemu zaczyna się faza eksploatacji i utrzymania, podczas której regularnie aktualizuje się oprogramowanie, naprawia błędy, dodaje nowe funkcje i poprawia istniejące funkcjonalności. To także czas na optymalizację wydajności i reagowanie na zmieniające się potrzeby użytkowników [11].

Te etapy współdziałają ze sobą, tworząc cykl, który może być iterowany wielokrotnie w miarę rozwijania i doskonalenia oprogramowania, aby lepiej spełniać wymagania użytkowników i radzić sobie z dynamicznie zmieniającym się środowiskiem technologicznym.

4. Hasła oraz szyfrowanie

4.1. Zasady Wyboru Haseł

Wybór solidnego hasła jest kluczowym elementem zabezpieczania dostępu do danych cyfrowych. W świetle współczesnych standardów cyberbezpieczeństwa, istnieje kilka zasad, które powinny być stosowane podczas tworzenia haseł:

- 1) Długość hasła: Zaleca się, aby hasła miały co najmniej 8 znaków. Większa liczba znaków zwiększa trudność złamania hasła przez ataki siłowe.
- 1) Złożoność: Hasła powinny zawierać kombinację dużych liter, małych liter, cyfr oraz znaków specjalnych. Użycie różnych znaków w hasle minimalizuje ryzyko udanego ataku słownikowego.
- 2) Unikatowość: Każde hasło powinno być unikatowe dla różnych kont i usług, aby zapobiec efektowi domina w przypadku uzyskania przez osoby trzecie dostępu do jednego z haseł.
- 3) Nieregularność: Hasła nie powinny zawierać łatwo dostępnych lub przewidywalnych sekwencji, takich jak nazwy, daty czy popularne frazy.
- 4) Regularna aktualizacja: Regularne zmiany haseł mogą pomóc w ograniczeniu ryzyka nieautoryzowanego dostępu, choć niektóre nowsze wytyczne sugerują zmniejszenie częstotliwości zmian na rzecz zwiększenia siły i unikalności haseł.

4.2. Analiza zagrożeń

Ryzyko złamania haseł stanowi jedno z głównych zagrożeń w dziedzinie cyberbezpieczeństwa. Jest to proces, w którym nieautoryzowane osoby próbują uzyskać dostęp do niedostępnych danych poprzez odgadnięcie lub skuteczne odkrycie haseł użytkowników. Istnieje wiele metod i technik, które złodzieje danych mogą wykorzystać do złamania haseł, co prowadzi do różnorodnych form ryzyka [13]. Poniżej przedstawiam główne aspekty związane z ryzykiem złamania haseł:

4.2.1. Metody złamania haseł

- 1) Ataki siłowe (brute force): W tej technice, atakujący używa oprogramowania do automatycznego generowania i próbowania ogromnej liczby kombinacji haseł aż do znalezienia prawidłowego.
- 2) Ataki słownikowe: Metoda ta polega na używaniu gotowych list słów, które często są używane jako hasła, takich jak powszechnie używane słowa, popularne frazy czy standardowe kombinacje liczbowe.
- 3) Phishing: Technika polegająca na manipulacji użytkownika, aby sam podał swoje hasło, często poprzez fałszywe strony internetowe czy e-maile wyglądające na oficjalne.
- 4) Inżynieria społeczna: Polega na wykorzystaniu informacji uzyskanych od samego użytkownika lub z jego otoczenia (np. daty urodzenia, nazwiska matki przed ślubem), które mogą być użyte do odgadnięcia haseł.

4.2.2. Czynniki zwiększające ryzyko

- 1) Słabe hasła: Używanie krótkich, prostych lub powszechnie znanych haseł znacznie zwiększa ryzyko ich złamania.
- 2) Ponowne używanie haseł: Używanie tego samego hasła w wielu miejscach stanowi ryzyko, ponieważ wyciek z jednego serwisu naraża wszystkie pozostałe konta.
- 3) Niewystarczające zabezpieczenia: Brak dodatkowych środków bezpieczeństwa, takich jak szyfrowanie danych czy wieloskładnikowa autentykacja, może ułatwić dostęp do haseł.

4.2.3. Konsekwencje złamania haseł

- 1) Utrata danych: Nieautoryzowany dostęp może prowadzić do kradzieży wrażliwych danych osobowych, finansowych lub biznesowych.
- 2) Utrata kontroli nad kontami: Złodziej hasła może przejąć kontrolę nad kontami e-mail, kontami w mediach społecznościowych i innymi usługami cyfrowymi.
- 3) Wycieki danych: Złamanie haseł może prowadzić do masowych wycieków danych, wpływając negatywnie na reputację firmy i prowadząc do konsekwencji prawnych związanych z naruszeniem prywatności.

4.2.4. Strategie obronne

- 1) Wdrażanie polityk haseł: Stosowanie zasad tworzenia silnych haseł przez organizacje może zmniejszyć ryzyko złamania.
- 2) Wieloskładnikowa autentykacja (MFA): Używanie więcej niż jednej metody uwierzytelniania znacznie utrudnia nieautoryzowany dostęp.
- 3) Szkolenia i świadomość użytkowników: Edukowanie użytkowników o ryzykach i strategiach bezpieczeństwa może ograniczyć skuteczność ataków phishingowych i inżynierii społecznej.
- 4) Regularne aktualizacje i monitoring bezpieczeństwa: Monitorowanie podejrzanych aktywności i regularne aktualizacje systemów bezpieczeństwa mogą wykrywać i zapobiegać próbom złamania haseł.

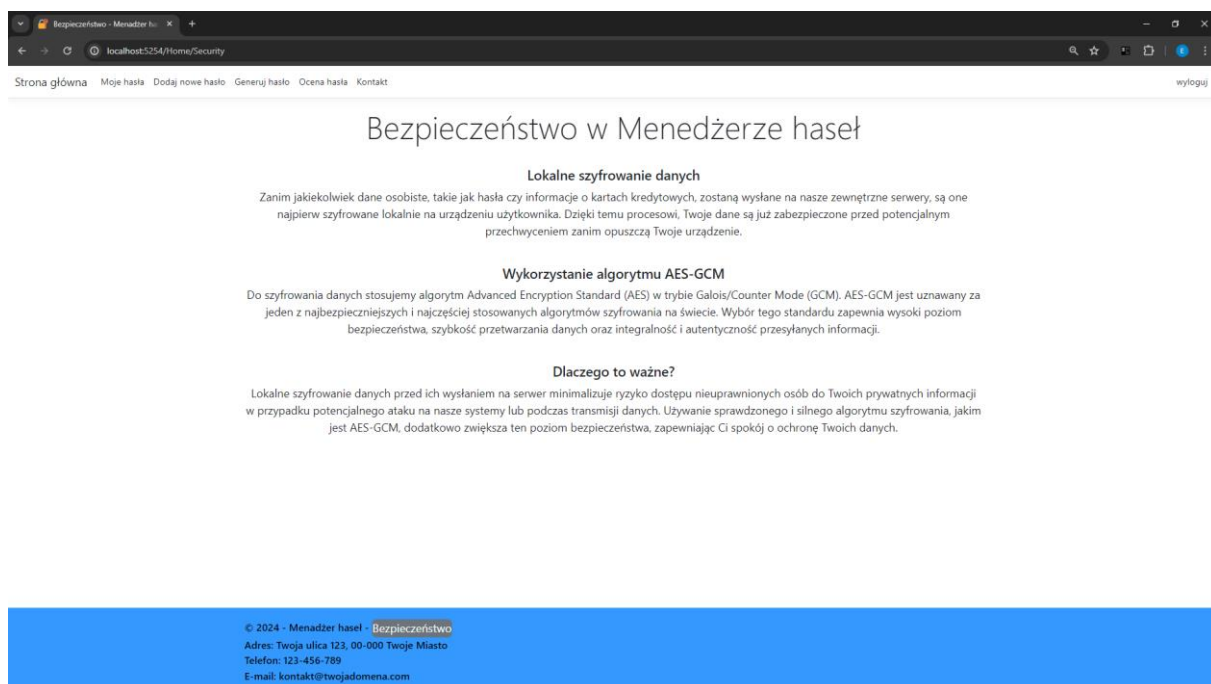
4.3. Algorytm szyfrujący zaimplementowany w projekcie

Algorytm AES-GCM (Advanced Encryption Standard - Galois/Counter Mode) to współczesny standard szyfrowania, który łączy w sobie symetryczne szyfrowanie blokowe AES z trybem operacyjnym GCM, zapewniając zarówno szyfrowanie, jak i uwierzytelnienie danych. Jest to algorytm o dużej wydajności i bezpieczeństwie, stosowany w wielu aplikacjach wymagających ochrony danych.

AES jest symetrycznym szyfrem blokowym, co oznacza, że zarówno do szyfrowania, jak i deszyfrowania używa się tego samego klucza. Algorytm może operować na kluczach o długości 128, 192 lub 256 bitów, a rozmiar bloku danych wynosi 128 bitów [1]. AES jest szeroko stosowany ze względu na swoją efektywność i bezpieczeństwo, a jego implementacja jest możliwa w wielu środowiskach - od sprzętowych po oprogramowanie. Tryb Galois/Counter Mode (GCM) jest trybem operacyjnym, który zapewnia nie tylko szyfrowanie, ale także uwierzytelnianie wiadomości. Jest to istotne w kontekście zabezpieczania integralności danych oraz ich poufności. GCM używa kombinacji licznika (podobnie jak w trybie CTR - Counter) i

blokowego szyfrowania, co pozwala na równoległe przetwarzanie bloków, zwiększając wydajność szyfrowania.

Proces szyfrowania AES-GCM. Dane są dzielone na bloki o wielkości 128 bitów. Każdy blok jest szyfrowany niezależnie, z wykorzystaniem wartości licznika, który jest inkrementowany dla każdego bloku. Po zaszyfrowaniu bloku, wartość ta jest mieszana z szyfrogramem w celu utworzenia macierzy autentykacyjnej (Galois field), co jest kluczowym elementem zapewniającym uwierzytelnienie danych.



Rys. 3.1. Widok podstrony bezpieczeństwo

AES-GCM jest powszechnie stosowany w protokołach komunikacyjnych, takich jak TLS i IPSec, ze względu na swoje właściwości zapewniające zarówno szyfrowanie, jak i uwierzytelnianie. Jego efektywność i bezpieczeństwo sprawiają, że jest on preferowany w aplikacjach wymagających wysokiej ochrony danych, takich jak systemy bankowe, aplikacje rządowe oraz inne krytyczne systemy informacyjne. Wybrany algorytm jest uważany za bardzo bezpieczny, pod warunkiem poprawnej implementacji i stosowania odpowiednio silnych kluczy. Zastosowanie trybu GCM zwiększa złożoność obliczeniową, ale dzięki możliwości równoległego przetwarzania bloków, szyfrowanie jest bardzo wydajne nawet w środowiskach o dużym natężeniu transmisji danych.

Podsumowując, Advanced Encryption Standard (AES) jest jednym z najbardziej rozpowszechnionych i zaufanych algorytmów szyfrowania stosowanych w dzisiejszym

bezpieczeństwie cyfrowym. Od czasu jego przyjęcia jako standardu przez National Institute of Standards and Technology (NIST) w 2001 roku, AES zyskał uznanie ze względu na swoją solidność i efektywność w zabezpieczaniu poufnych danych. Algorytm ten jest używany w wielu aplikacjach rządowych i komercyjnych na całym świecie, włączając w to systemy bankowe, oprogramowanie VPN i protokoły komunikacyjne, takie jak TLS i IPSec [6]. Dzięki swojej odporności na większość znanych ataków kryptograficznych, w tym ataki brute-force, AES jest uważany za bardzo bezpieczny. Wersje AES z kluczami o długości 128 i 256 bitów umożliwiają elastyczność w zależności od wymaganego poziomu bezpieczeństwa, a jego zdolność do szybkiego przetwarzania w środowiskach o dużej przepustowości czyni go idealnym wyborem dla nowoczesnych aplikacji wymagających zarówno bezpieczeństwa, jak i wydajności. Ogólnie rzecz biorąc, AES pozostaje kluczowym elementem w arsenale narzędzi kryptograficznych, efektywnie służąc jako fundament dla zabezpieczania danych w obliczu rosnących zagrożeń cyfrowych.

Możliwe są różne warianty implementacji, które w wielu przypadkach mogą oferować lepszą wydajność lub inne zalety. Jeśli przy tym samym kluczu wejściowym i danych (tekst jawny lub zaszyfrowany), dana implementacja produkuje ten sam wynik (tekst zaszyfrowany lub jawny), co algorytm określony w tej normie, jest to równoważna implementacja algorytmu AES.

W projekcie został użyty klucz szyfrujący o długości 256 bitów. Większa długość klucza zwiększa bezpieczeństwo przez podniesienie trudności przeprowadzenia skutecznego ataku, jednak wiąże się to również z większymi wymaganiami obliczeniowymi podczas szyfrowania i deszyfrowania, co może wpływać na wydajność systemu.

- Klucz 128-bitowy oferuje 2^{128} możliwych kombinacji.
- Klucz 256-bitowy oferuje 2^{256} możliwych kombinacji.

Liczba 2^{256} jest o rzędy wielkości większa niż 2^{128} . Po podzieleniu jednej wartości przez drugą otrzymamy o ile więcej kombinacji otrzymujemy przy zastosowaniu dłuższego klucza.

$$\frac{2^{256}}{2^{128}}$$

Po uproszczeniu:

$$2^{256-128} = 2^{128}$$

Po wypotęgowaniu otrzymujemy 3,4028236692093846346337460743177e+38, czyli liczbę składającą się z 39 cyfr. Obrazuje to jak bardzo złożony jest wybrany algorytm.

4.4. Zastosowanie Soli i Wektora Inicjalizującego w Procesach Szyfrowania

Sól to losowo generowany ciąg danych dodawany do hasła przed jego zahashowaniem. Cele stosowania soli są dwa: po pierwsze, zapobiega ona wykorzystaniu pre-kompilowanych tablic hashy, znanych jako "rainbow tables". Po drugie, sprawia, że identyczne hasła po zahashowaniu dają różne wyniki. Dzięki temu, nawet jeśli dwa hasła są identyczne, ich hashe będą różne, co utrudnia wykonanie ataku na hasła przechowywane w bazie danych.

Wektor Inicjalizujący (IV) to blok danych używany w szyfrowaniu blokowym, który jest potrzebny do zainicjowania procesu szyfrowania. IV musi być unikalny (ale niekoniecznie tajny) dla każdej operacji szyfrowania, aby zapewnić, że te same dane wejściowe szyfrowane wielokrotnie dają różne dane wyjściowe. Zapobiega to pewnym rodzajom ataków analizy wzorców, które mogą wystąpić, gdy ten sam blok danych jest szyfrowany kilkakrotnie bez zmiany IV.

Sól i IV to fundamentalne komponenty w zaawansowanych systemach szyfrowania. Ich właściwe zastosowanie może znacząco zwiększyć odporność systemów kryptograficznych na różnorodne techniki ataków. Zapewnienie, że sól jest wystarczająco losowa i że IV jest zawsze unikalny, jest kluczowe dla zachowania bezpieczeństwa w cyfrowych systemach komunikacji i przechowywania danych. Dalsze badania w tej dziedzinie mogą prowadzić do nowych odkryć, które jeszcze bardziej zabezpieczą dane użytkowników internetu.

5. Wybrane narzędzia

5.1. ASP.NET w środowisku Microsoft Visual Studio

ASP.NET to rozbudowana platforma przeznaczona do tworzenia aplikacji webowych, która została opracowana przez firmę Microsoft. Jest ona zbudowana na fundamencie frameworku .NET i pozwala na konstrukcję dynamicznych stron internetowych, rozbudowanych aplikacji webowych oraz zaawansowanych usług internetowych. Jako platforma wielojęzykowa, ASP.NET wspiera różnorodne języki programowania, takie jak C#, VB.NET, i F#, co umożliwia programistom wybór odpowiednich narzędzi zależnie od specyfiki projektu i preferencji.

Jedną z kluczowych cech ASP.NET jest zorientowanie na zdarzenia podejście do programowania, które umożliwia efektywne reagowanie na interakcje użytkownika poprzez event handlers, czyli specjalne procedury obsługi zdarzeń. Dzięki temu programiści mogą łatwo

implementować funkcjonalności takie jak przetwarzanie kliknięć czy wprowadzanie danych przez użytkowników.

Platforma ta oferuje również bogaty zestaw gotowych kontrolerek serwerowych, które automatyzują tworzenie HTML, JavaScript oraz CSS, znacząco przyspieszając proces budowania interfejsów użytkownika. Kontrolki te, dzięki swojej modularności, pozwalają na szybkie i efektywne tworzenie zaawansowanych funkcjonalności bez konieczności pisania obszernego kodu od podstaw.

Zarządzanie stanem sesji jest kolejnym istotnym aspektem, który ASP.NET obsługuje z wyjątkową efektywnością. Mechanizmy te są niezbędne w aplikacjach, które wymagają utrzymywania stanu użytkownika pomiędzy różnymi żądaniami, co jest krytyczne dla aplikacji wymagających ciągłej autoryzacji lub utrzymywania ciągłości interakcji użytkownika.

Wspieranie wzorca projektowego Model-View-Controller (MVC) jest jedną z nowszych ewolucji w ASP.NET, co pozwala na jeszcze lepsze oddzielenie logiki biznesowej od warstwy prezentacji. MVC jest szczególnie cenione w środowisku programistycznym za promowanie czystego kodu, który jest łatwiejszy w utrzymaniu i rozbudowie.

Z punktu widzenia wydajności, ASP.NET zapewnia kompilację kodu źródłowego do języka pośredniego (Intermediate Language, IL), który następnie jest kompilowany do kodu maszynowego w momencie wykonania, co zapewnia wysoką wydajność działania aplikacji. Platforma ta oferuje również zaawansowane funkcje związane z bezpieczeństwem, takie jak uwierzytelnianie, autoryzacja oraz zabezpieczenia przed różnego rodzaju atakami, co jest kluczowe w kontekście tworzenia bezpiecznych aplikacji internetowych.

Skalowalność jest jednym z fundamentalnych aspektów ASP.NET, pozwalającym na elastyczne dostosowanie aplikacji do rosnących potrzeb użytkowników i zwiększającej się złożoności operacyjnej. Wspierana przez silną społeczność i wsparcie techniczne od Microsoft, platforma ta stanowi jedno z kluczowych rozwiązań w arsenale nowoczesnych technologii programistycznych, zdolnych sprostać wymaganiom zarówno małych, jak i dużych przedsięwzięć informatycznych.

Jedną z najbardziej użytecznych funkcji wybranego środowiska jest przeładowywanie na gorąco, znane również jako "hot reloading" jest to technika programistyczna używana głównie w procesie rozwoju oprogramowania, która pozwala na wprowadzanie zmian w kodzie aplikacji na żywo, bez potrzeby zatrzymywania i ponownego uruchamiania całego systemu lub aplikacji. Ta funkcja jest szczególnie cenna w środowiskach deweloperskich, gdzie szybkość iteracji i możliwość natychmiastowego zobaczenia efektów wprowadzonych zmian mogą znacząco przyspieszyć proces tworzenia oprogramowania. Przeładowywanie na gorąco działa

poprzez monitorowanie zmian w plikach źródłowych projektu. Gdy deweloper wprowadza zmiany w kodzie (na przykład w pliku JavaScript, CSS czy kodzie źródłowym aplikacji), mechanizm przeładowania na gorąco wykrywa te zmiany i automatycznie zastępuje lub aktualizuje uruchomione części aplikacji nowym kodem, bez konieczności restartowania aplikacji. W niektórych przypadkach, takich jak aplikacje webowe, może to dotyczyć jedynie określonych modułów lub komponentów.

Razor to silnik szablonów stworzony przez Microsoft, integralny dla platformy ASP.NET, który pozwala deweloperom na płynne integrowanie kodu C# z HTML, co jest szczególnie użyteczne przy tworzeniu dynamicznych stron internetowych i aplikacji webowych. Silnik ten jest zintegrowany z Visual Studio, co oferuje szerokie możliwości i wsparcie w procesie deweloperskim.

Razor charakteryzuje się prostą i intuicyjną składnią, która używa znacznika `@` do osadzania kodu serwerowego C# wewnątrz plików HTML. Taka integracja kodu z znacznikami umożliwia tworzenie dynamicznych treści, które są generowane serwerowo, a następnie renderowane jako HTML. Dzięki temu deweloperzy mogą łatwo manipulować zawartością strony, dostosowując ją do potrzeb użytkowników i kontekstu wykonania aplikacji.

Visual Studio, będące rozbudowanym środowiskiem programistycznym, oferuje zaawansowane narzędzia do pracy z Razor, takie jak podświetlanie składni, autouzupełnianie kodu oraz bezpośrednie podglądy renderowanej strony. Te funkcje znacząco przyspieszają i ułatwiają proces tworzenia kodu, pozwalając programistom na skupienie się na logice biznesowej, nie zaś na ręcznym zarządzaniu szczegółami implementacji.

5.2. Firebase

Firebase to platforma rozwoju aplikacji mobilnych i webowych zaprojektowana przez Google, która oferuje różnorodne narzędzia i usługi wspierające rozwój, zarządzanie oraz skalowanie aplikacji. Dwie kluczowe usługi oferowane przez Firebase, które mają znaczący wpływ na proces tworzenia i zarządzania aplikacjami, to Firebase Database oraz Firebase Authentication. Te komponenty wspierają odpowiednio przechowywanie danych i zarządzanie tożsamością użytkowników, co jest fundamentalne dla bezpieczeństwa, efektywności i skalowalności aplikacji.

Firestore Database, znany również jako Cloud Firestore, stanowi kluczowy komponent platformy Firebase, który znacznie rozszerza możliwości deweloperów w kontekście zarządzania danymi w aplikacjach mobilnych i webowych. Jako nowoczesna, skalowalna baza danych typu NoSQL oferowana przez Google, Firestore umożliwia efektywne

przechowywanie, synchronizację i zarządzanie danymi w strukturze dokumentów, co czyni go bardziej zaawansowanym niż jego poprzednik, Firebase Realtime Database.

Firestore przechowuje dane w formie dokumentów, które są zorganizowane w kolekcje. Każdy dokument może zawierać różne typy danych, takie jak tekst, liczby, daty, listy czy zagnieżdżone dokumenty, umożliwiając tworzenie złożonych i elastycznych struktur danych. Te dokumenty są automatycznie indeksowane, co zapewnia wysoką wydajność zapytań. Co więcej, kolekcje mogą zawierać inne kolekcje, tworząc hierarchiczną bazę danych, która jest zarówno potężna, jak i łatwa w zarządzaniu.

Jedną z najbardziej atrakcyjnych cech Firestore jest jego zdolność do synchronizacji danych w czasie rzeczywistym. Każda zmiana w dokumencie jest natychmiast rejestrowana i propagowana do wszystkich aktywnych klientów, umożliwiając użytkownikom na bieżąco obserwować aktualizacje bez konieczności odświeżania aplikacji. Ta funkcjonalność jest szczególnie przydatna w aplikacjach, które zależą od interakcji w czasie rzeczywistym, takich jak aplikacje do czatowania, gry wieloosobowe czy aplikacje współpracy.

Firebase Authentication zapewnia kompleksowe rozwiązanie do zarządzania tożsamością użytkowników w aplikacji. Usługa ta obsługuje zarówno standardowe metody logowania, takie jak e-mail i hasło, jak i logowanie przez zewnętrzne dostawców tożsamości, w tym Google, Facebook, Twitter i GitHub. Dodatkowo, Firebase Authentication wspiera logowanie anonimowe, które użytkownicy mogą wykorzystać do przetestowania aplikacji przed założeniem konta.

Zaawansowane funkcje Firebase Authentication obejmują integrację z innymi usługami Firebase, co umożliwia tworzenie spójnych reguł bezpieczeństwa opartych na tożsamości użytkownika. Na przykład, reguły dla Firestore Database mogą być skonfigurowane, aby zezwalać na dostęp tylko zautoryzowanym użytkownikom, co zwiększa bezpieczeństwo i prywatność danych.

Firebase Authentication również oferuje funkcje takie jak weryfikacja dwuetapowa i resetowanie hasła, co dodatkowo wzmacnia bezpieczeństwo aplikacji. Ponadto, usługa ta zapewnia skalowalne rozwiązanie, które może obsługiwać duże ilości użytkowników i transakcji, co jest kluczowe dla rosnących aplikacji.

Podsumowując platforma Firebase, wraz z usługami takimi jak Firestore Database oraz Firebase Authentication, dostarcza zestaw zaawansowanych narzędzi umożliwiających programistom projektowanie aplikacji, które są jednocześnie dynamiczne, bezpieczne i łatwe w skalowaniu. Zastosowanie tych technologii w kontekście szerszego ekosystemu Firebase oraz chmury Google Cloud Platform, umożliwia efektywne zarządzanie infrastrukturą

aplikacji, automatyzację procesów deweloperskich oraz implementację zaawansowanych funkcji analitycznych. Takie podejście pozwala na adaptację do ciągle zmieniających się warunków rynkowych i potrzeb użytkowników, czyniąc Firebase kluczowym komponentem w arsenale programistów aplikacji mobilnych i webowych. Dzięki swojej kompleksowości i modułowości, Firebase przyczynia się do znaczącego przyspieszenia procesu rozwoju aplikacji, co jest krytyczne w szybko ewoluującej przestrzeni technologicznej [7].

5.3. C#

Język programowania C# to nowoczesny język programowania rozwijany przez Microsoft od 2000 roku, stanowiący kluczowy element platformy .NET. Charakteryzuje się obiektywnym paradygmatem i jest szeroko ceniony za swą moc, elastyczność oraz wszechstronność. Znacząco przyczynił się do ewolucji tworzenia oprogramowania, oferując developerom efektywne narzędzia do budowy zarówno prostych, jak i złożonych aplikacji systemowych, webowych oraz mobilnych.

Język C# jest statycznie typowany, co przekłada się na większą kontrolę nad bezpieczeństwem typów oraz wydajność aplikacji. Jego składnia jest czysta i zrozumiała, co obniża próg wejścia dla nowych programistów oraz przyspiesza procesy deweloperskie. C# oferuje rozbudowane wsparcie dla wielu paradygmatów programowania, w tym programowania obiektowego, imperatywnego, a także elementy programowania funkcyjnego i zdarzeniowego, co czyni go niezwykle elastycznym.

Jako język stworzony z myślą o platformie .NET, C# doskonale integruje się z jej ekosystemem, umożliwiając wykorzystanie szerokiej gamy bibliotek i narzędzi, takich jak ASP.NET do tworzenia aplikacji webowych, Entity Framework do obsługi baz danych, czy Windows Presentation Foundation (WPF) dla aplikacji desktopowych. Platforma .NET dostarcza również rozbudowane środowisko czasu wykonania, Common Language Runtime (CLR), które zarządza wykonywaniem kodu, obsługą pamięci i wieloma innymi aspektami operacji systemowych, pozwalając developerom na skupienie się na logice aplikacji [15].

C# znajduje zastosowanie w wielu dziedzinach programowania. Jest powszechnie stosowany do tworzenia aplikacji dla systemu Windows, zarówno klasycznych aplikacji desktopowych, jak i nowoczesnych aplikacji uniwersalnych (UWP). Dodatkowo, dzięki narzędziom takim jak Xamarin, programiści mogą używać C# do tworzenia aplikacji na platformy mobilne, w tym Android i iOS. C# jest również popularnym wyborem dla programistów gier, głównie za sprawą silnika Unity, który pozwala na tworzenie gier na prawie każdą platformę.

Programowanie w C# wiąże się z szeregiem wyzwań, w tym zarządzanie pamięcią i zasobami, chociaż platforma .NET znacząco te aspekty ułatwia. Bezpieczeństwo aplikacji jest innym kluczowym zagadnieniem, a C# oferuje wiele mechanizmów bezpieczeństwa, w tym silne typowanie, bezpieczne zarządzanie pamięcią i szczegółowe API do zarządzania dostępem i autoryzacją.

Język ten jest ciągle rozwijany, na oficjalnej stronie Microsoft możemy znaleźć następującą deklarację. „Będziemy nadal rozwijać język C#, aby sprostać zmieniającym się potrzebom programistów i pozostać nowoczesnym językiem programowania. Będziemy z entuzjazmem i wprowadzać innowacje we współpracy z zespołami odpowiedzialnymi za biblioteki .NET, narzędzia dla programistów oraz wsparcie dla różnych zastosowań, jednocześnie starając się pozostać w duchu języka. Uznając różnorodność dziedzin, w których używany jest C#, będziemy preferować ulepszenia języka i wydajności, które przynoszą korzyści wszystkim lub większości programistów, zachowując przy tym wysokie zaangażowanie w kompatybilność wsteczną” [2].

C# utrzymuje swoją pozycję jako jeden z wiodących języków programowania dzięki ciągłym ulepszeniom, adaptacji do nowych technologii i solidnej integracji z platformą .NET. Jego wszechstronność i moc sprawiają, że jest idealnym wyborem dla szerokiego zakresu zastosowań, od aplikacji korporacyjnych, przez rozbudowane systemy backendowe, po zaawansowane gry komputerowe.

5.4. Javascript

JavaScript jest dynamicznym językiem programowania, który odgrywa kluczową rolę w ekosystemie tworzenia stron internetowych. Od momentu jego pojawienia się w 1995 roku, JavaScript zyskał na znaczeniu jako fundamentalny element technologiczny w projektowaniu interaktywnych i dynamicznych witryn internetowych. Jego uniwersalność, wszechstronność i nieustanne rozwijanie się przez społeczność deweloperów sprawiają, że JavaScript jest obecnie niezbędnym narzędziem w arsenale każdego front-end dewelopera.

JavaScript początkowo był używany głównie do dodawania prostych efektów interaktywnych na stronach internetowych. Jednakże z biegiem lat jego możliwości znacznie się rozszerzyły. Obecnie JavaScript umożliwia tworzenie skomplikowanych aplikacji internetowych (web apps), które działają płynnie i oferują użytkownikowi doświadczenie porównywalne z aplikacjami natywnymi.

Jedno z bardzo przydatnych funkcji jest `alert()`, pozwala ona na łatwe i czytelne przekazywanie informacji do użytkownika. Ta funkcjonalność nie należy do rdzenia języka (nie

jest zawarta w specyfikacji ECMA), ale można z niej korzystać w środowisku przeglądarki. Umożliwia wyświetlanie komunikatów w okienku dialogowym. Okienko dialogowe blokuje wątek przeglądarki, co oznacza, że żaden inny kod nie zostanie wykonany, dopóki użytkownik nie kliknie OK [3].

Mimo licznych zalet, JavaScript wiąże się także z wyzwaniem w kontekście bezpieczeństwa. Skrypty mogą być wykorzystywane do przeprowadzania ataków, takich jak cross-site scripting (XSS) czy clickjacking. Dlatego też istotne jest stosowanie najlepszych praktyk programistycznych oraz regularne aktualizowanie używanych bibliotek i frameworków, aby zabezpieczyć aplikacje przed potencjalnymi zagrożeniami.

Podsumowując, JavaScript, będąc w czołówce technologii webowych, stanowi oś współczesnego rozwoju stron internetowych. Jego rola w tworzeniu dynamicznych, interaktywnych i bezpiecznych aplikacji jest niepodważalna. Dzięki ciągłym innowacjom i rosnącej społeczności deweloperów, JavaScript pozostaje na froncie ewolucji cyfrowej, umożliwiając tworzenie coraz to nowszych i lepszych rozwiązań w przestrzeni internetowej.

5.5. HTML oraz CSS

HTML (HyperText Markup Language) i CSS (Cascading Style Sheets) stanowią podstawę tworzenia stron internetowych, definiując strukturę i prezentację treści w sieci internetowej. Te dwa języki, choć fundamentalnie różne w swoich funkcjach i celach, współpracują ze sobą, aby umożliwić tworzenie estetycznych i funkcjonalnych witryn, które są dostępne na różnych urządzeniach i platformach. Niniejszy artykuł ma na celu przedstawienie ewolucji, zastosowań oraz kluczowych cech HTML i CSS w kontekście rozwoju stron internetowych.

HTML jest standardem w zakresie definiowania struktury i organizacji treści na stronie internetowej. Pierwsza specyfikacja HTML została opracowana w 1991 roku i od tego czasu przeszła wiele rewolucji, z HTML5 jako najnowszą i najbardziej zaawansowaną wersją. HTML5 wprowadziło wiele usprawnień, w tym wsparcie dla multimediów (audio i wideo), nowe semantyczne elementy (takie jak `<article>`, `<section>`, `<nav>`, `<header>`, `<footer>`), a także zaawansowane formularze i elementy interaktywne, co znacząco poszerzyło możliwości tworzenia bogatych aplikacji internetowych [8].

CSS został pierwotnie zaprojektowany jako środek do oddzielenia treści dokumentu od detali prezentacji, takich jak kolory, czcionki i układ. Rozwój CSS przebiegał równolegle do HTML, a jego najnowsza wersja, CSS3, przynosi rozwiązania takie jak animacje, gradienty, przejścia, transformacje, a także nowe modele układu jak Flexbox i Grid. Te narzędzia

pozwalają deweloperom na tworzenie responsywnych i atrakcyjnych wizualnie stron internetowych, które dostosowują się do rozmiaru i orientacji ekranu użytkownika [5].

HTML i CSS są niezależne, ale ich prawdziwa moc ujawnia się w integracji. HTML definiuje strukturę strony, podczas gdy CSS zarządza jej wyglądem i formatowaniem. Razem umożliwiają one tworzenie responsywnych, dostępnych i funkcjonalnych witryn internetowych. Przykładowo, semantyczne elementy HTML mogą być stylizowane za pomocą CSS w sposób, który wspiera dostępność i SEO, a także poprawia ogólną użyteczność strony.[9]

5.6. Bootstrap

Bootstrap jest otwartoźródłowym frameworkiem front-endowym, który znacząco ułatwia i przyspiesza proces tworzenia responsywnych i atrakcyjnych stron internetowych oraz aplikacji webowych. Został zaprojektowany i rozwijany początkowo przez Marka Otto i Jacoba Thorntona w Twitterze, a pierwsza wersja została wydana w 2011 roku. Od tego czasu Bootstrap zyskał na popularności w społeczności deweloperskiej jako narzędzie, które zapewnia spójność, efektywność oraz łatwość w dostosowywaniu designu.

Bootstrap oferuje bogaty zestaw gotowych komponentów, takich jak przyciski, formularze, karty, nawigacje i wiele innych, które są dostępne w postaci predefiniowanych klas CSS. Co więcej, Bootstrap zapewnia system siatki (grid system), który umożliwia elastyczne i intuicyjne tworzenie układów strony, dostosowując się do różnych rozdzielczości i urządzeń. Dzięki temu deweloperzy mogą tworzyć responsywne strony internetowe, które prawidłowo wyświetlają się zarówno na dużych monitorach, jak i na ekranach smartfonów.

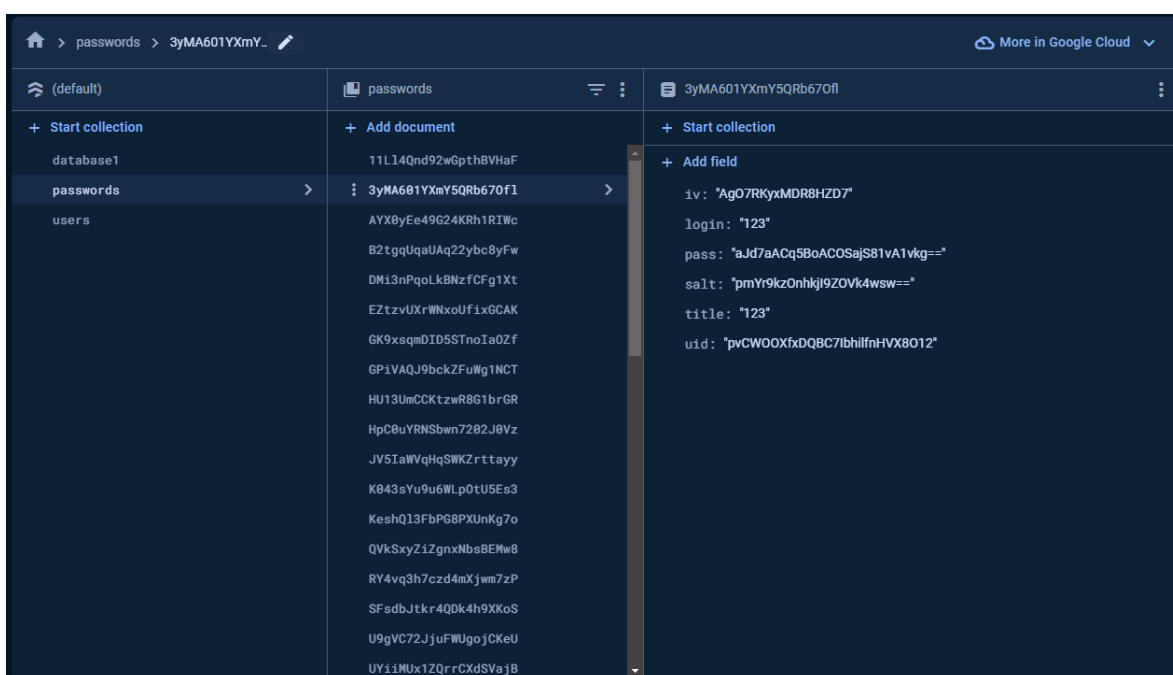
Jedną z kluczowych zalet Bootstrapa jest jego modularność. Deweloperzy mogą wybierać, które komponenty chcą zaimportować do swojego projektu, co pozwala na optymalizację rozmiaru plików końcowych.

5.7. Baza danych NoSQL

Bazy danych NoSQL, czyli "Not Only SQL", to kluczowy element w zakresie zarządzania danymi, który zyskał na znaczeniu dzięki swojej zdolności do efektywnego zarządzania bardzo dużymi zbiorami danych, jak również ze względu na elastyczność i skalowalność, które są niezbędne w nowoczesnych aplikacjach internetowych i mobilnych. Systemy NoSQL różnią się od tradycyjnych relacyjnych baz danych swoją strukturą i sposobami modelowania danych, co jest odpowiedzią na ograniczenia tych ostatnich w kontekście rosnących wymagań aplikacji o dużym natężeniu danych [10].

Bazy dokumentowe, takie jak Firestore Database, przechowują dane w formie dokumentów, które zazwyczaj są formatowane jako JSON, BSON, lub XML. Każdy dokument w bazie jest niezależną jednostką danych zawierającą różne klucze i wartości. Dokumenty te nie muszą posiadać tego samego schematu, co umożliwia dużą elastyczność w zarządzaniu strukturą danych.

Kolekcja to grupa dokumentów. Dokumenty w ramach kolekcji są zazwyczaj spokrewnione z tą samą encją obiektu, taką jak profile użytkowników czy przechowywane hasła. Możliwe jest przechowywanie w kolekcji niezwiązanych ze sobą dokumentów, ale nie jest to zalecane [4].



Rys. 5.1. Widok panelu bazy danych

Document ID	iv	login	pass	salt	title	uid
11L14Qnd92wGpthBVHaF	"blEdnrxwB01yq50ay"	"asdasd"	"yLFPv2u1DkiBxbCgtrtNIEHJABbgbPQ="	"tWdHqDKonVPRj2FzLfw4Rw=="	"asdddd"	"jqPJsRrGuHV"
3yMA601YXmY5QRb67OfI	"Ag07RKyxMDR8HZD7"	"123"	"aJd7aACq5BoACOSajS81vA1vkg=="	"pmYr9kzOnhkjI9ZOVk4wsw=="	"123"	"pvCW00XfxD"
AYX0yEe49G24KRh1RIWc	"WgTZ4PF8xeP7LSUm"	"asd"	"tOpPGY2xTj4m4M1AdYIN+KgB+Q=="	"uow5xpqS1H008slCNjPDYQ=="	"asd"	"prieF8UxolZc"
B2tggUqaUAq22ybc8yFw	"1/GsdEiSPGxou92p"	"asd"	"hJhn1U7fklam+kd+KvR7BVlwA=="	"ProOrJeYUUX2Ma1BvKzPHg=="	"asd"	"prieF8UxolZc"
CSIo3EOLz8l2C7XjJbzb	"k87Oc/QD6V0UWYsr"	"eryk123"	"FvHLKNeo0frp+hCRYFo11vNX2wREqukvWsFhQRzupQ=="	"PHmri2ixbZYOTRRgJOB7IA=="	"facebook"	"U4Dy8l4K3UI"
DMi3nPqoLkBNzfCFg1Xt	"k/XJ3UuAmvy1GHcY"	"asdddasd"	"+CYySxcuJzRxN2ZbK40pineBYm5uMVQ="	"DkzaxgzOVOONTD/WNMtIFA=="	"asdasdad"	"jqPJsRrGuHV"
EztzvUXrWNxoUfIXGCAK	"FT3hkb+h+xpSatp"	"123"	"6zWzntbEg6STuMlegQnlyjZ95Q=="	"6LeQcIPNWxjBzImxszVnlw=="	"123"	"pvCW00XfxD"
GK9xsqmDID5STnolaOZf	"0AwGck3tDYqC71P"	"facebokk"	"UrOuBzn8HM1yv0VMZ0RijQ9+izzvZc5"	"UJCeSgwHUQwW7uWom0834Q=="	"facebokk"	"pvCW00XfxD"

Rys. 5.2. Widok panelu bazy danych w postaci tabeli

Tab. 5.1 Schemat bazy danych

Nazwa Kolumny	Typ danych
Document ID	string
iv	string
login	string
Pass	string
Salt	string
title	string
uid	string

Bazy danych NoSQL są znane z elastyczności schematów, co oznacza, że nie zawsze wymagają ściśle określonych typów danych dla każdej kolumny. Wiele systemów NoSQL, takich jak Firestore Database, domyślnie traktuje dane jako ciągi znaków, chyba że wyraźnie zaznaczono inaczej.

Identifier	Providers	Created ↓	Signed In	User UID
test13@log.pl		Apr 11, 2024	May 6, 2024	U4Dv8l4K3UMoHrgAFmBjlrFjf...
test12@log.pl		Apr 8, 2024	Apr 11, 2024	ftkorglkEDW22OSnqC0sTHVc...
test11@log.pl		Apr 8, 2024	Apr 8, 2024	zYE5v6ZtzkfEcFpKTbNTTq6F...
test10@wp.pl		Apr 5, 2024	Apr 5, 2024	Ti0PC5D0F4ezE4VDQdsrq5Ar...
test7@wp.pl		Mar 25, 2024	Mar 25, 2024	QshloLNnbDYcbQqkOZQkAsa...
test6@wp.pl		Mar 25, 2024	Apr 5, 2024	prieF8UxolZq5eHxzgjxmglg1...
test5@wp.pl		Mar 25, 2024	Mar 25, 2024	pvCW00XfxDQBC7lbhlfHnHVX...
test3@wp.pl		Mar 25, 2024	Mar 25, 2024	IUAWxpJ2j7OaymtMsWKX31f...

Rys. 5.3. Widok zarejestrowanych użytkowników

W kontekście rosnącej świadomości społecznej na temat prywatności oraz wzrastających regulacji prawnych dotyczących ochrony danych osobowych, projektowanie baz danych z minimalnym zbieraniem danych staje się kluczowym aspektem w rozwijaniu aplikacji zapewniających zarówno bezpieczeństwo, jak i komfort użytkowników. Minimalizacja danych polega na świadomym ograniczaniu ilości i rodzaju gromadzonych informacji do tego, co jest absolutnie niezbędne do realizacji określonych funkcji aplikacji czy usługi. Ten podejście jest zgodne z zasadą minimalizacji danych, która jest jednym z filarów ogólnego rozporządzenia o ochronie danych (GDPR) obowiązującego w Unii Europejskiej, a także innych podobnych regulacji na całym świecie.

Głównym celem minimalizacji danych jest zwiększenie prywatności użytkowników poprzez ograniczenie potencjalnego ryzyka związanego z naruszeniem danych. Mniejsza ilość przechowywanych informacji oznacza mniejsze ryzyko w przypadku potencjalnych incydentów bezpieczeństwa. Ponadto, minimalizacja danych pomaga budować zaufanie użytkowników, którzy stają się coraz bardziej świadomi i wymagający w kwestii ochrony swoich danych osobowych.

6. Opis funkcjonalności aplikacji internetowej

6.1. Rejestracja i logowanie

Rejestracja użytkowników na stronie internetowej jest kluczowym elementem wielu aplikacji webowych, szczególnie tych wymagających personalizacji i ochrony danych użytkowników. W projekcie proces rejestracji użytkownika został zrealizowany przy pomocy Firebase - jednej z najpopularniejszych platform backendowych oferowanej przez Google. Firebase zapewnia wiele usług, takich jak baza danych w czasie rzeczywistym, hosting, uwierzytelnianie użytkowników i wiele innych.

Ikony są pobierane z zewnętrznej biblioteki. Google Fonts to internetowa biblioteka czcionek, która umożliwia szybkie korzystanie z obrazów na stronach internetowych. Czcionek oraz ikony są przechowywane na serwerach Google i są być ładowane bezpośrednio na stronę za pomocą linku [12].



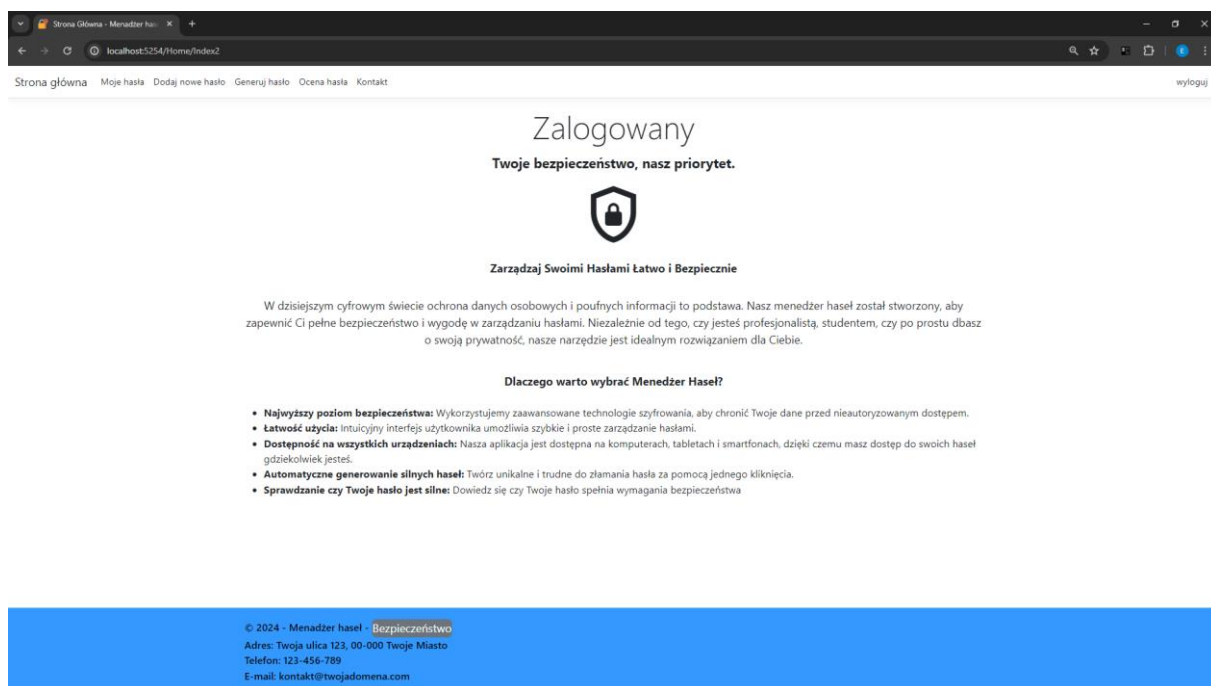
The image shows a registration form titled "Zarejestruj się" (Register). Below the title is a dark silhouette of a person with a plus sign to its right. The form contains two input fields: "Adres email" (Email address) with the value "test20@email.pl" and "Hasło" (Password) with the value "SIIneHasło123\$". Below the email field is a small text note: "Nie udostępniamy nikomu Twojego adresu email." (We do not share your email address with anyone). To the right of the password field is an eye icon for toggling visibility. At the bottom of the form is a green button labeled "Zarejestruj się".

Rys. 6.1. Widok panelu rejestracji

Firebase Authentication jest usługą, która ułatwia proces uwierzytelniania użytkowników. Główna metoda jest logowanie za pomocą adresu e-mail i hasła. Po pomyślej autoryzacji użytkownik dostaje dostęp do zasobów, które zostały mu przydzielone. Uwierzytelnianie jest kluczowym aspektem bezpieczeństwa każdej nowoczesnej aplikacji, a jego skuteczność bezpośrednio wpływa na ochronę danych użytkownika.

Autoryzacja to proces weryfikacji tożsamości użytkownika i przypisania mu odpowiednich uprawnień. W kontekście Firebase, autoryzacja odbywa się głównie za pomocą tokenów JWT (JSON Web Tokens), które są generowane po pomyślnym uwierzytelnieniu użytkownika. Proces ten rozpoczyna się od uwierzytelnienia, podczas którego użytkownik wprowadza swoje dane logowania, takie jak e-mail i hasło. Następnie Firebase generuje token JWT, który jest unikalny dla sesji użytkownika. Token ten jest przechowywany po stronie klienta, zazwyczaj w lokalnym magazynie przeglądarki lub w pamięci aplikacji mobilnej. Przy każdym żądaniu do serwera token jest przesyłany, a serwer weryfikuje jego ważność i uprawnienia użytkownika, co pozwala na autoryzację dostępu do chronionych zasobów.

Autoryzacja użytkownika jest kluczowym elementem każdej aplikacji webowej i mobilnej, zapewniającym, że dostęp do zasobów i funkcji aplikacji mają tylko uprawnione osoby.



Rys. 6.2. Widok strony głównej po zalogowaniu

6.2. Dodawanie, wyświetlanie, aktualizacja oraz usuwanie haseł

CRUD to akronim odnoszący się do czterech podstawowych operacji wykonywanych na danych w aplikacjach informatycznych, których celem jest zarządzanie danymi przechowywanymi w bazach danych lub innych magazynach danych. Skrót CRUD pochodzi od angielskich słów: Create, Read, Update, Delete, które oznaczają odpowiednio:

Create (Tworzenie) rysunek 6.3 – Dodawanie nowych danych do bazy. Operacja ta polega na tworzeniu nowych rekordów w bazie danych. W kontekście interfejsów użytkownika, często wiąże się to z formularzami, które pozwalają użytkownikom na wprowadzenie i zapisanie nowych danych.

Dodaj nowe hasło



Tytuł:

facebook

Login:

Eryk123

Hasło:

haslofacebook



Klucz szyfrujący:

Silnehaslo



Każde hasło musi być zabezpieczone **tym samym kluczem**.

Ten Klucz to jedyny sposób, aby wyświetlić hasła.

Zapamiętaj lub zapisz go!

Zapisz

Rys. 6.3. Widok panelu dodawania nowego hasła

Read (Odczyt) rysunek 6.4 oraz 6.5 – Ta operacja umożliwia przeglądanie istniejących zaszyfrowanych danych. Może to być realizowane poprzez wyświetlenie tylko wybranego hasła lub wyświetlenie wszystkich przechowywanych haseł. Dane można odczytać tylko po wprowadzaniu poprawnego klucza szyfrującego.

Oto lista haseł.

Klucz szyfrujący:

Deszyfruj wybrane hasło

Nazwa: facebook

Login: Eryk

Hasło: haslofacebook

Usuń

Aktualizuj

Rys. 6.4. Widok panelu wyświetlania hasła

Oto lista wszystkich twoich haseł.

Klucz szyfrujący:

Deszyfruj wszystkie hasła

Pokaż/Ukryj hasła

Nazwa: onet

Login: piotr123

Hasło: BLO7O1-An7va6-dcCFPI

Usuń **Aktualizuj**

Nazwa: facebook

Login: Eryk

Hasło: haslofacebook

Usuń **Aktualizuj**

Rys. 6.5. Widok panelu wyświetlania wszystkich haseł

Update (Aktualizacja) rysunek 6.6 – Operacja ta pozwala na zmianę już istniejących rekordów w bazie danych. W praktyce oznacza to modyfikacje istniejących danych do logowania.

Aktualizuj dane.

Nazwa:

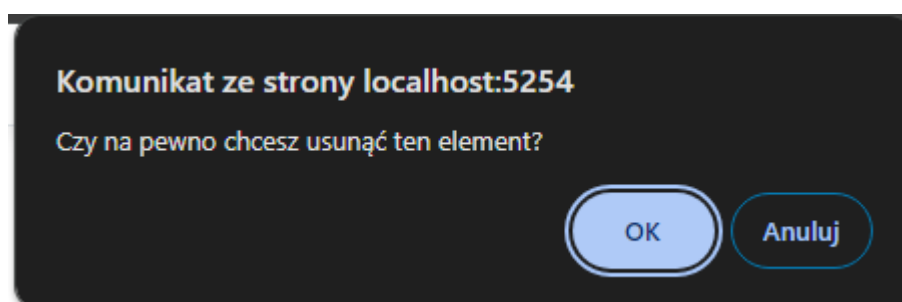
Login:

Hasło:

Klucz:

Rys. 6.6. Widok panelu edycji hasła

Delete (Usuwanie) – rysunek 5.7 Ta funkcjonalność umożliwia usuwanie rekordów z bazy danych. Jest to krytyczna operacja, która jest zabezpieczona dodatkowym komunikatem przed przypadkowym usunięciem danych. Operacji usuń nie można cofnąć, dane zostają trwale utracone.



Rys. 6.7. Komunikat przed usunięciem hasła

6.3. Generowanie i ocena haseł

Generator losowych haseł zaimplementowany w aplikacji jest jedną z bardziej użytecznych funkcjonalności dla użytkownika końcowego. Generowanie losowych haseł jest techniką stosowaną do tworzenia silnych i bezpiecznych haseł, które są niemal niemożliwe do odgadnięcia dla ludzi i maszyn. Losowe oraz silne hasła są kluczowym elementem zabezpieczania kont online i ochrony danych osobowych przed nieautoryzowanym dostępem.

Na rysunku 6.8 pokazano działanie poniższego kodu JavaScript, który generuje 3 grupy po 6 losowych znaków. Takie podejście uznaje się za bardzo bezpieczne i takie hasła są niemal niemożliwe do złamania w rozsądnym czasie.

```
Func<string> generatePassword = () =>
{
    var random = new Random();
    var characters =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    Func<int, string> generatePart = length =>
        new string(Enumerable.Repeat(characters, length)
            .Select(s => s[random.Next(s.Length)]).ToArray());

    return $"{generatePart(6)}-{generatePart(6)}-{generatePart(6)}";
};
```

Generowanie haseł



Wygeneruj bezpieczne hasło:

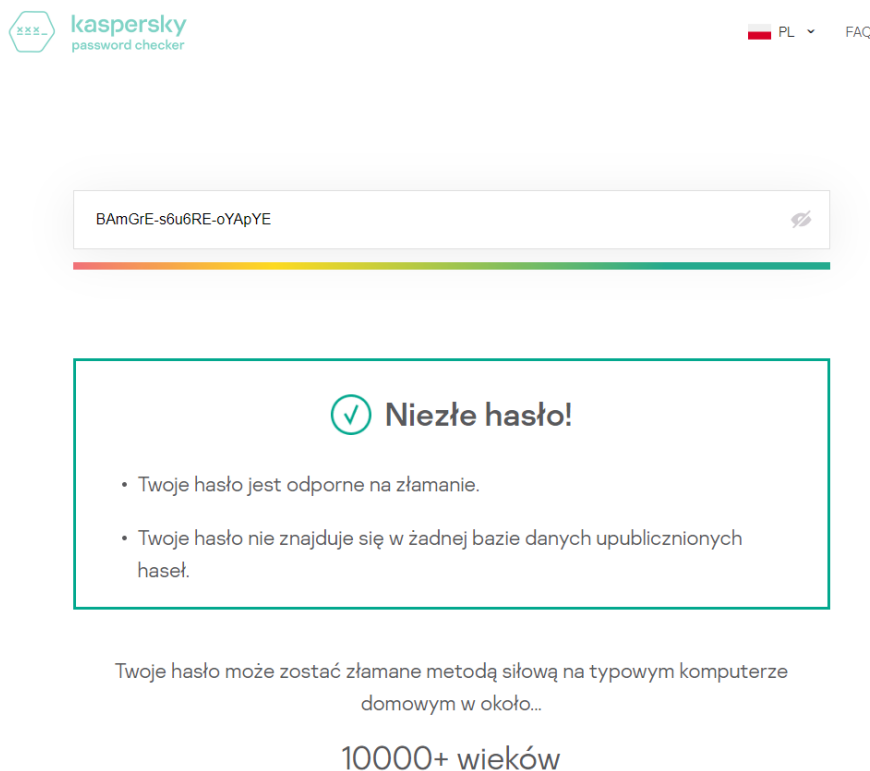
BAmGrE-s6u6RE-oYApYE

Generuj nowe hasło

Kopiuj hasło do schowka

Rys. 6.8. Generowanie bezpiecznych haseł

Według narzędzia udostępnionego przez producenta programów antywirusowych rysunek 6.9, powyższe hasło wygenerowane poprzez menadżer haseł może zostać złamane metoda siłową na domowym komputerze w około 1000000 (jeden milion) lat.




Rys. 6.9. Wynik testu wygenerowanego hasła przy pomocy zewnętrznego narzędzia
Źródło: <https://password.kaspersky.com/pl/>

Ocena siły haseł, to proces analizowania haseł pod kątem ich odporności na próby złamania. Jest to ważny element w utrzymaniu bezpieczeństwa cyfrowego, ponieważ silne hasła są podstawową linią obrony przed nieautoryzowanym dostępem do systemów i danych. Głównymi cechami, które powinno zawierać hasło są:

- co najmniej 8 znaków
- małe i duże litery
- co najmniej jedna cyfra
- co najmniej jeden znak specjalny

Na rysunkach 6.10 oraz 6.11 przedstawione jest działanie wbudowanej w aplikację funkcji sprawdzania haseł użytkownika. Hasło jest sprawdzane lokalnie i nie jest nigdzie przechowywane. Rezultatem działania jest komunikat pozytywny lub negatywny.

Ocena haseł




Wprowadź hasło:

Sprawdź siłę hasła

Hasło nie jest wystarczająco silne. Upewnij się, że zawiera co najmniej 8 znaków, w tym duże litery, małe litery, cyfry i znaki specjalne.

Rys. 6.10. Widok funkcji oceny haseł (negatywny)

Ocena haseł



Wprowadź hasło:

Sprawdź siłę hasła

Hasło jest silne.

Rys. 6.11. Widok funkcji oceny haseł (pozytywny)

Przestawiona funkcjonalność oceniania haseł jest realizowana za pomocą poniższego kodu C#. Podane hasło jest sprawdzane 5 warunkami.

```
public static class PasswordStrengthChecker
{
    public static bool IsPasswordStrong(string password)
    {
        // Sprawdzenie długości hasła
        if (password == null || password.Length < 8)
            return false;

        // Sprawdzenie czy hasło zawiera co najmniej jedną dużą literę
        if (!password.Any(char.IsUpper))
            return false;

        // Sprawdzenie czy hasło zawiera co najmniej jedną małą literę
        if (!password.Any(char.IsLower))
            return false;

        // Sprawdzenie czy hasło zawiera co najmniej jedną cyfrę
        if (!password.Any(char.IsDigit))
            return false;

        // Sprawdzenie czy hasło zawiera co najmniej jeden znak specjalny
        if (!password.Any(ch => !char.IsLetterOrDigit(ch)))
            return false;

        return true;
    }
}
```

Formularz kontaktowy przedstawiony na rysunku 6.12 jest niezbędną funkcją w aplikacji menedżer haseł, które spełnia ważną rolę w zapewnieniu efektywnej i skutecznej komunikacji pomiędzy użytkownikami, a dostawcą usługi. W ramach takiego narzędzia, użytkownicy mogą łatwo zgłaszać wszelkie problemy, pytać o funkcjonalności, czy wyrażać swoje opinie na temat aplikacji, co jest kluczowe dla ciągłego udoskonalania jakości usług.

W przypadku menedżera haseł, gdzie bezpieczeństwo i zaufanie są kluczowymi aspektami, formularz kontaktowy staje się punktem pierwszego kontaktu dla użytkowników potrzebujących wsparcia. Może to dotyczyć kwestii technicznych, takich jak problemy z logowaniem, odzyskiwanie dostępu do konta, czy pytania dotyczące zabezpieczeń haseł. Umożliwienie użytkownikom szybkiego zgłaszania takich problemów przyczynia się do podniesienia ich zaufania do narzędzia, które ma chronić ich najważniejsze dane. Administrator aplikacji dostaje maila w postaci pokazanej na rysunku 6.13.

Kontakt



Masz pytanie? Napisz do nas!

Imię:

Adres e-mail:

Wiadomość:

zapytanie

Wyślij

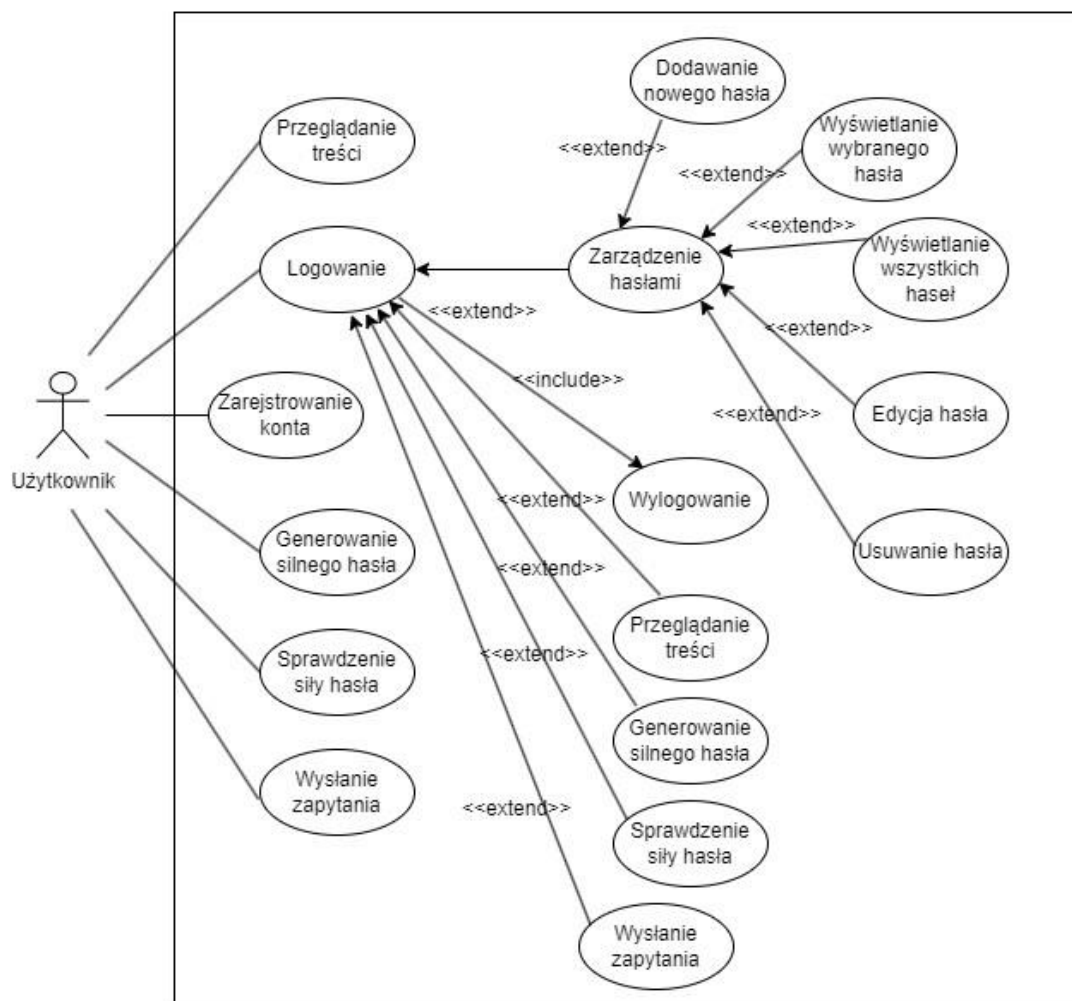
Rys. 6.12. Widok formularza kontaktowego



Rys. 6.13. Wygląd maila, który został wysłany z aplikacji

6.4. Diagram przypadków użycia

Diagram przypadków użycia przedstawiony na rysunku 6.14 to narzędzie modelowania używane w inżynierii oprogramowania, które służy do identyfikacji funkcjonalności systemu z punktu widzenia jego końcowych użytkowników. Diagramy te są kluczowym elementem UML (Unified Modeling Language) i odgrywają fundamentalną rolę w fazie projektowania i analizy systemów informatycznych.



Rys. 6.14. Diagram Przypadków użycia

Na przedstawionym diagramie przypadków użycia, można wyróżnić następujące elementy i ich role w systemie zarządzania hasłami:

1) Logowanie

Główna funkcja, która pozwala użytkownikowi uzyskać dostęp do systemu po pomyślnym wprowadzeniu danych uwierzytelniających.

2) Rejestracja konta

Umożliwia nowym użytkownikom utworzenie konta w systemie, co jest zwykle pierwszym krokiem do korzystania z usług.

3) Dodawanie nowego hasła

Umożliwia użytkownikowi dodanie nowego hasła do bazy danych. Funkcja ta jest rozszerzana między innymi przez "Wyświetlanie wybranego hasła" i "Wyświetlanie wszystkich haseł".

4) Wyświetlanie wybranego hasła

Pozwala użytkownikowi zobaczyć szczegóły konkretnego hasła, które jest już zapisane w systemie.

5) Wyświetlanie wszystkich haseł

Umożliwia użytkownikowi przegląd wszystkich zapisanych haseł w systemie.

6) Edycja hasła

Daje możliwość modyfikacji istniejących danych logowania.

7) Usuwanie hasła

Umożliwia użytkownikowi nieodwracalne usunięcie hasła z bazy danych.

8) Generowanie silnego hasła

Funkcja dostępna jest również dla niezalogowanych użytkowników. Pomaga użytkownikowi w utworzeniu mocnego i trudnego do złamania hasła.

9) Sprawdzenie siły hasła

Funkcja dostępna jest również dla niezalogowanych użytkowników. Umożliwia użytkownikowi sprawdzenie, jak silne jest jego hasło, co może wpłynąć na decyzje dotyczące jego zmiany lub ulepszenia.

10) Wylogowanie

Pozwala użytkownikowi bezpiecznie opuścić system, zamykając sesję użytkownika.

11) Wysyłanie zapytania

Funkcja, która umożliwia użytkownikowi kontakt z administratorem systemu w celu rozwiązania problemów lub uzyskania dodatkowych informacji. Funkcja dostępna jest również dla niezalogowanych użytkowników.

Każda z tych powyższych funkcji jest połączona z aktorem "Użytkownik", co oznacza, że wszystkie te działania są dostępne dla osób zalogowanych w systemie. Natomiast niezalogowany użytkownik nie ma możliwości skorzystać z funkcjonalności dotyczących przechowywania i zarządzania hasłami. Diagram również pokazuje relacje rozszerzenia (extend) i zawierania (include), które określają, jak interakcje między różnymi funkcjami istnieją w systemie.

7. Podsumowanie

Projekt inżynierski zakładający stworzenie zaawansowanego menedżera haseł został pomyślnie ukończony. Celem inicjatywy było opracowanie aplikacji, która nie tylko skutecznie zwiększa bezpieczeństwo cyfrowe użytkowników poprzez zaawansowane mechanizmy szyfrowania, ale również oferuje wygodę i prostotę obsługi w codziennym użytkowaniu.

Menadżer haseł, będący odpowiedzią na rosnące wymagania związane z zarządzaniem licznymi kont cyfrowych, został wyposażony w szereg funkcji co czyni go kompleksową aplikacją internetową. Dzięki implementacji technologii szyfrowania end-to-end, wszystkie dane są bezpieczne przed dostępem osób trzecich, nawet w przypadku ewentualnego przejęcia fizycznego urządzenia czy włamania na serwer przechowujący silnie zaszyfrowane hasła. Menadżer haseł został zaprojektowany tak, aby funkcjonować jako samodzielna aplikacja, niezależna od ekosystemów dużych korporacji technologicznych. To zapewnia użytkownikom większą kontrolę nad ich danymi oraz minimalizuje ryzyko nadużyć i wycieków informacji.

Projekt posiada również intuicyjny i łatwy w użyciu interfejs użytkownika, zaprojektowany z myślą o osobach o różnym stopniu zaawansowania technologicznego. Menadżer haseł jest kompatybilny z różnymi systemami operacyjnymi, co zapewnia łatwość integracji i użytkowania.

Dodatkowo, aplikacja oferuje funkcje takie jak generowanie silnych losowych haseł, sprawdzanie siły już istniejących haseł oraz możliwość zarządzania hasłami. Menadżer haseł również umożliwia bezpieczne przechowywanie innych wrażliwych informacji oprócz haseł, co czyni go wszechstronnym narzędziem do zarządzania prywatnością online.

Osiągnięcie celów przedsięwzięcia związanych ze stworzeniem zaawansowanego menadżera haseł, przy jednoczesnym zachowaniu zgodności z założeniami projektowymi, wymagało interdyscyplinarnego podejścia, które łączyło wiedzę z różnych dziedzin nauki oraz kreatywne myślenie. Rozwój aplikacji zintegrował aspekty z zakresu projektowania stron internetowych, bezpieczeństwa komputerowego, kryptografii, programowania oraz interakcji człowiek-komputer, co było kluczowe dla zapewnienia wysokiego poziomu bezpieczeństwa oraz użyteczności produktu. Zakończenie projektu przyniosło nowe, efektywne narzędzie, które nie tylko podnosi poziom bezpieczeństwa cyfrowego użytkowników, ale także znacząco usprawnia zarządzanie ich poświadczeniami. Menadżer haseł stanowi wartościowy dodatek do cyfrowego ekosystemu, oferując stabilne, bezpieczne i łatwe w obsłudze rozwiązanie dla każdego, kto ceni sobie ochronę swoich danych w internecie.

Literatura

- [1] National Institute of Standards and Technology, Advanced Encryption Standard (AES), 2023. Dostęp: <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf>
- [2] <https://learn.microsoft.com/en-us/dotnet/fundamentals/languages>
- [3] Stoyan Stefanov, JavaScript. Programowanie obiektowe, Helion, Gliwice, 2013.
- [4] Dan Sullivan „NoSQL przyjazny przewodnik”, Helion, Gliwice, 2016.
- [5] David Sawyer McFarland, CSS. Nieoficjalny podręcznik, Helion, Gliwice, 2017
- [6] W. Stallings, „Kryptografia i bezpieczeństwo sieci komputerowych. Matematyka szyfrów i techniki kryptografii” Helion, 2011.
- [7] <https://www.youtube.com/watch?v=igGIht2EByc&t=4s>
- [8] M. MacDonald HTML5. Nieoficjalny podręcznik. Wydanie II,. Helion, Gliwice, 2014.
- [9] <http://www.w3schools.com/>
- [10] M. Muraszkiewicz, H. Rybiński, Bazy Danych, AOW, Warszawa, 2003.
- [11] I. Sommerville Software Engineering 9th Edition, Addison Wesley, 2010
- [12] <https://fonts.google.com/>
- [13] Stallings William, Ochrona danych w sieci i intersieci, Wydawnictwo Naukowo Techniczne, 1997.
- [14] B. Frain, Projektowanie elastycznych witryn w HTML5 i CSS3, Helion, Gliwice, 2021.
- [15] <https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/>

STRESZCZENIE PROJEKTU INŻYNIERSKIEGO

Menadżer haseł

Autor: Eryk Delikat, 168139: EF-ZI

Opiekun: dr inż. Grzegorz Drałus

Słowa kluczowe: Menadżer haseł, Aplikacja internetowa, Szyfrowanie, ASP.NET.

Rezultatem tego projektu jest zaawansowany menadżer haseł, który jest niezależną aplikacją, która zwiększa bezpieczeństwo cyfrowe użytkowników poprzez efektywne i odpowiedzialne zarządzanie ich poświadczeniami. Główną funkcjonalnością jest przechowywanie i zarządzanie hasłami użytkownika, ale dodatkowo zostały zaimplementowane dodatkowe funkcjonalności takiej jak generowanie bezpiecznych i losowych haseł oraz ocena siły haseł podanych przez użytkownika. Menadżer haseł wyposażono w zaawansowane funkcje szyfrowania end-to-end przy pomocy algorytmu AES-GCM. Wyłącznie użytkownik posiada klucz deszyfrujący, co jest bardzo bezpiecznym podejściem, gwarantującym poufność danych.

DIPLOMA THESIS (BS) ABSTRACT

Password manager

Author: Eryk Delikat, code: EF-ZI

Supervisor: dr inż. Grzegorz Drałus

Key words: Password manager, Web application, Encryption, ASP.NET.

The result of this project is an advanced password manager, which is an independent application that enhances users' digital security through effective and responsible management of their credentials. The main functionality is the storage and management of user passwords, but additional features such as generating secure and random passwords and assessing the strength of passwords provided by the user have also been implemented. The password manager is equipped with advanced end-to-end encryption features using the AES-GCM algorithm. Only the user possesses the decryption key, which is a very secure approach, ensuring data confidentiality.