

Warszawa, 13.06.2021

Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych

Bezpieczeństwo medycznych systemów informacyjnych
(BEMSI)

Projekt: Aplikacja web/mobile – szyfrowany notatnik z
przechowywaniem w chmurze

Dokumentacja końcowa

Prowadzący: dr inż. Robert Kurjata

Wykonawcy:

Adamiuk Zuzanna 300444, Depko Kinga 300452

Krakowiak Aleksandra 290292, Rancew Joanna 300465

Spis treści

Cel projektu.....	3
Projekt rozwiązania.....	3
Dostęp dla użytkownika.....	3
Bezpieczeństwo.....	3
Realizacja projektu.....	3
Wykorzystane technologie.....	3
Środowisko pracy.....	4
Wygląd aplikacji	4
Strona startowa.....	4
Rejestracja.....	4
Logowanie.....	6
Strona użytkownika	7
Implementacja rozwiązania	10
Baza danych	10
Analiza ryzyka i przyjęte modele zabezpieczeń	12
Logowanie/Rejestracja użytkownika - hasło	12
Szyfrowanie notatek.....	13
Włamanie się do systemu.....	14
Instrukcja kompilacji, uruchamiania i korzystania	14
Rozwój aplikacji	14
Bibliografia	15

Cel projektu

Głównym celem projektu było stworzenie szyfrowanego notatnika, który będzie umożliwiał bezpieczne przechowywanie danych w chmurze. Operacje wykonywane w aplikacji zapewniają poufność, za równo w trakcie przechowywania, jak i przesyłania notatek. Na każdym etapie pracy z danymi, stosowane są odpowiednie zabezpieczenia, by nie były one dostępne dla osób trzecich.

Projekt rozwiązania

Dostęp dla użytkownika

Zaproponowane rozwiązanie wykorzystuje technologię webową, dlatego też użytkownik, aby korzystać z informacji musi mieć dostęp do przeglądarki internetowej. Po uruchomieniu programu, użytkownik musi się zalogować (stosując login i hasło) lub jeśli nie posiada konta – zarejestrować się. Po wprowadzaniu poprawnych danych zostaje przyznany dostęp do przeglądania, usuwania oraz dodawania notatek – tylko tych przypisanych do zalogowanego autora.

Bezpieczeństwo

W celu zapewnienia bezpieczeństwa, część informacji przechowywanych w bazie danych jest zaszyfrowana funkcją haszującą z ziarnem. Zastosowanie ziarna wprowadza dodatkowe zabezpieczenie, ponieważ, w przypadku wycieku danych, nie zostanie ujawniona informacja o długości zaszyfrowanych danych oraz o ewentualnych powtarzających się hasłach w obrębie bazy. Nie przewidziano możliwości pobierania odszyfrowanych notatek – podgląd jest możliwy jedynie po uprzednim zalogowaniu się, w przeglądarce, co zabezpiecza aplikację także przed nierozsądnymi działaniami użytkowników i nieodpowiednim przechowywaniem notatek na ich urządzeniach. Ponadto, sesja użytkownika zabezpieczona limitem czasu bezczynności - po 3 minutach dochodzi do automatycznego wylogowania użytkownika, który nie wykonywał żadnych operacji. Dodatkowo kod umieszczony na repozytorium posiadał minimum wymaganych informacji, również plik `.gitignore` – nie ułatwiał ataku na naszą aplikację.

By zapewnić dodatkowe bezpieczeństwo podczas korzystania, aplikacja:

- nie umożliwia w żadnym stopniu korzystania bez logowania się,
- ma intuicyjny interfejs, co zmniejsza ryzyko błędów ludzkich,
- przy wybranych funkcjach zapewnia poufność, uwierzytelnienie,
- poprawność przesyłania danych w oparciu o stosowane modele,

Realizacja projektu

Wykorzystane technologie

Backend aplikacji został napisany w JavaScriptcie z wykorzystaniem Node.js i bibliotek:

- Express.js – framework do tworzenia aplikacji webowych
- Express EJS Layouts – ułatwienie generacji szablonów EJS, które pełnią rolę frontendu
- EJS – tworzenie intuicyjnych interfejsów

- Express Session – umożliwia operacje na sesjach, niezbędne do poprawnego logowania
- Mongoose, MongoDB – pozwalają na generację modeli z bazy MongoDB oraz komunikację z bazą
- Passport, Passport local – uwierzytelnianie użytkowników
- Connect-flash, popups – potrzebne do komunikacji, przechowywania wiadomości w sesji
- Bcrypt – niezbędna do prawidłowego szyfrowania treści, zawiera szereg funkcji szyfrujących
- Crypto, buffer – biblioteki używane do poprawnego szyfrowania notatek

Do przechowywania danych wykorzystano nierelacyjną bazę danych MongoDB. Stworzono dwie kolekcję users - przechowującą dane potrzebne do logowania (login, zaszyfrowane hasło) oraz notes - przechowującą notatki (login, title, date, zaszyfrowane description), a także keys – osobną kolekcję przechowującą klucze i wektory inicjujące.

Środowisko pracy

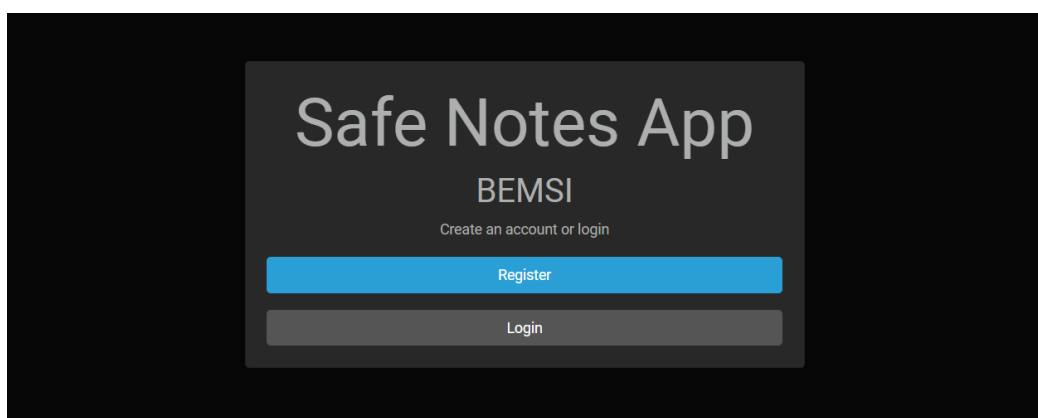
Kod źródłowy aplikacji został napisany w Visual Studio Code, zaś do obsługi bazy danych wykorzystano MongoDB Compass opierając się na dostępnej w chmurze bazie danych stworzonej przy użyciu MongoDB Atlas. Wszyscy wykonawcy korzystali z jednakowego środowiska, a spójność edytowanego kodu zapewniono przy systemie kontroli wersji Git.

Adres repozytorium projektu: https://gitlab-stud.elka.pw.edu.pl/jrancew/bemsi_notatnik

Wygląd aplikacji

Strona startowa

Po uruchomieniu serwera, wyświetlana jest strona główna aplikacji umożliwiająca logowanie i rejestrację.



Rysunek 1. Zrzut ekranu strony głównej aplikacji

Rejestracja

Po wybraniu opcji *Register*, użytkownik zostaje przekierowany do formularza rejestracyjnego, w którym jest proszony o wprowadzenia loginu oraz dwukrotne wprowadzenie hasła, aby uniknąć potencjalnych błędów.

Register

Please use new password with at least 10 characters.
Do not use password with words or any personal data

Remember: Do not share your password with anyone!

Login

Enter login

Password

Create Password

Confirm Password

Confirm Password

Register

Have An Account? [Login](#)

Rysunek 2. Zrzut ekranu strony do rejestracji

Bardzo ważnym elementem systemu bezpieczeństwa naszej aplikacji jest poprawne hasło. Z tego powodu użytkownik informowany jest o kilku zasadach tworzenia bezpiecznego hasła przy rejestracji.

Login użytkownika musi być wartością unikatową, dlatego po wprowadzenia wartości program weryfikuje, czy rekord o podanym loginie istnieje w bazie. Ponadto, formularz weryfikuje, czy wprowadzone hasło spełnia nałożone wymagania – ma co najmniej 10 znaków.

Register

Please use new password with at least 10 characters.
Do not use password with words or any personal data

Remember: Do not share your password with anyone!

Password should be at least 10 characters long ✕

Login

bemsi

Register

Please use new password with at least 10 characters.
Do not use password with words or any personal data

Remember: Do not share your password with anyone!

Login is already registered. Please log in or change the login ✕

Login

bemsi

Register

Please use new password with at least 10 characters.
Do not use password with words or any personal data

Remember: Do not share your password with anyone!

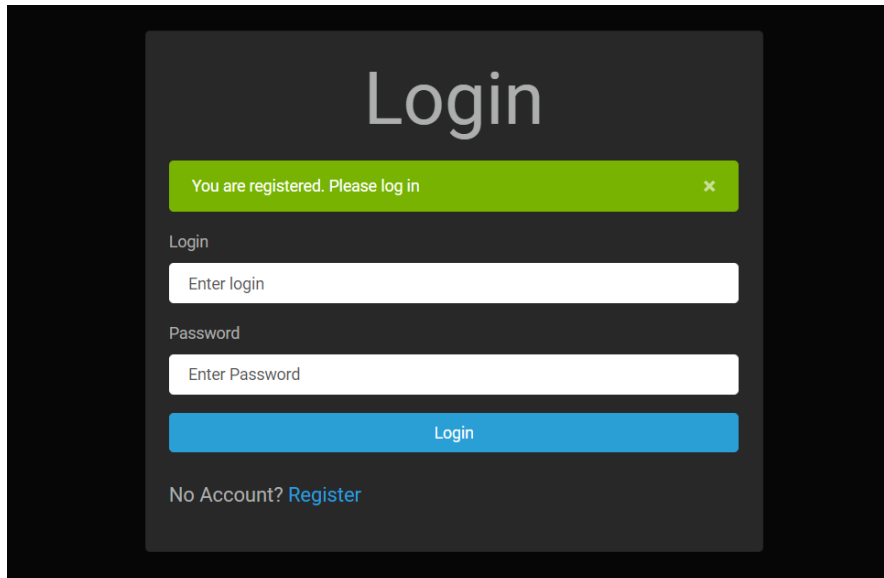
Passwords do not match ✕

Rysunek 3. Zrzuty ekranów komunikatów o błędach przy rejestracji

Po poprawnej rejestracji użytkownik zostaje przekierowany na stronę umożliwiającą logowanie. Wszelkie komunikaty na etapie rejestracji są przekazywane użytkownikowi w przystępny sposób, ze szczegółami.

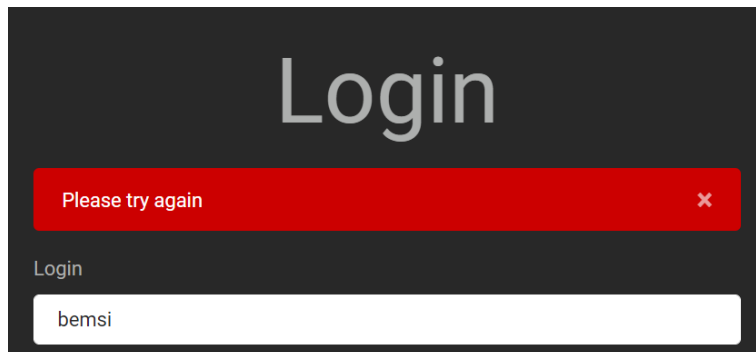
Logowanie

Panel logowania wymaga od użytkownika podania loginu oraz hasła.

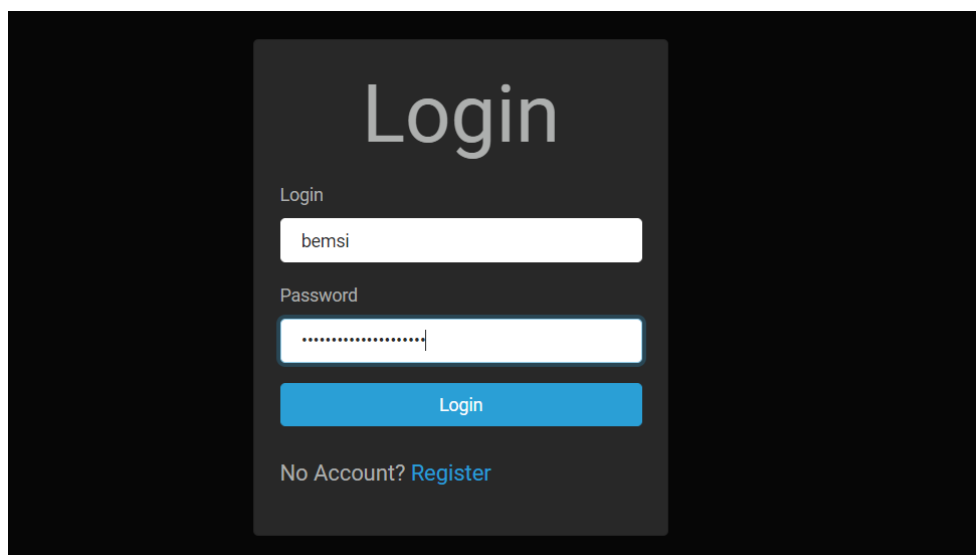
A screenshot of a web application's login page. The page has a dark gray background. At the top, the word "Login" is displayed in a large, white, sans-serif font. Below it, a green notification bar with a white 'x' icon on the right contains the text "You are registered. Please log in". Underneath the notification, there are two input fields: the first is labeled "Login" and contains the placeholder text "Enter login"; the second is labeled "Password" and contains the placeholder text "Enter Password". Below these fields is a blue button with the text "Login". At the bottom of the form, there is a link that says "No Account? Register".

Rysunek 4. Zrzut ekranu strony do logowania, po poprawnej rejestracji

W przypadku wprowadzenia błędnego hasła lub loginu wyświetla się stosowny komunikat – co ważne, nie podaje on informacji, które w pól zostało niepoprawnie uzupełnione, by utrudnić „brute-force attack”.

A screenshot of the same login page as in Figure 4, but with an error message. The green notification bar has been replaced by a red one with a white 'x' icon on the right, containing the text "Please try again". The "Login" input field now contains the text "bemsi". The "Password" field and the "Login" button remain unchanged.

Rysunek 5. Zrzut ekranu komunikatu o błędnie wprowadzonych danych logowania



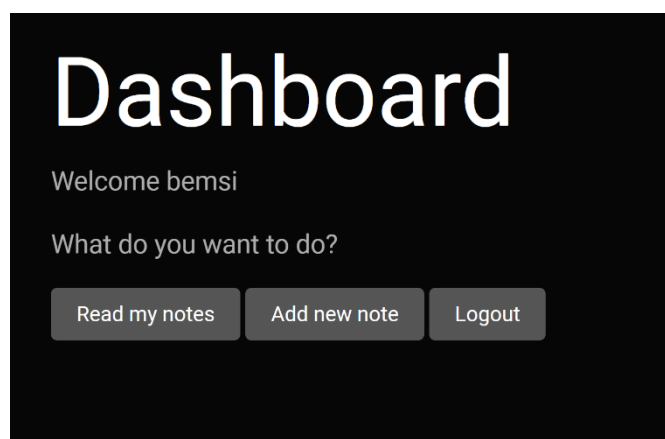
Rysunek 6. Zabezpieczenie widoczności hasła

Dodatkowym elementem zabezpieczeniem jest uniemożliwienie widoczności konkretnych znaków hasła podczas wprowadzania, co chroni użytkownika przed „Shoulder surfing”, czyli podglądaniem.

Po poprawnym wprowadzeniu danych, użytkownik zostaje przekierowany na dedykowaną stronę główną. Hasła porównywane są dzięki funkcji *compare* z biblioteki *bcrypt*, co zapewnia poufność także na tym etapie.

Strona użytkownika

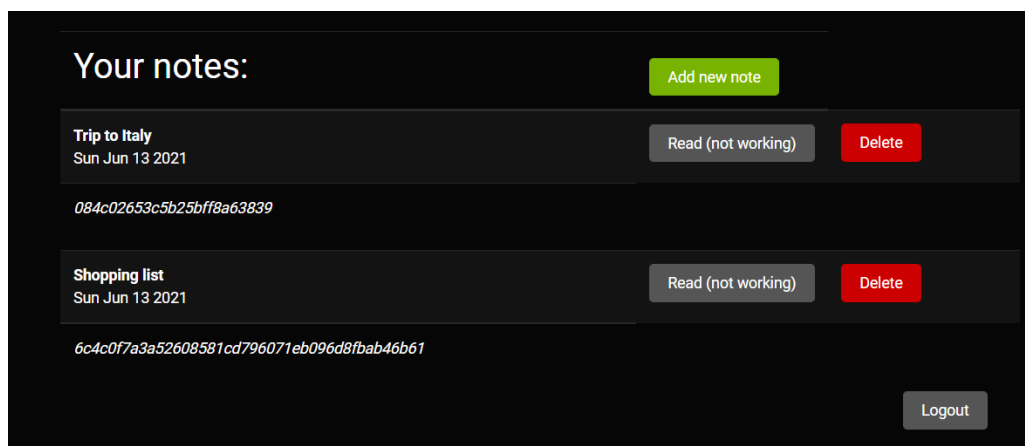
Po zalogowaniu rozpoczyna się uwierzytelniona sesja. Uwierzytelnienie zostało zapewnione poprzez logowanie, a z poziomu kodu: użycie bibliotek „passport” oraz odpowiednią konfigurację weryfikacji uwierzytelnienia na stronie głównej oraz stronach umożliwiających zarządzanie notatkami. Uwierzytelnienie jest widoczne także poprzez personalne powitanie użytkownika, zgodnie ze wzorem „*Welcome {login}*”.



Rysunek 7. Zrzut ekranu strony głównej użytkownika

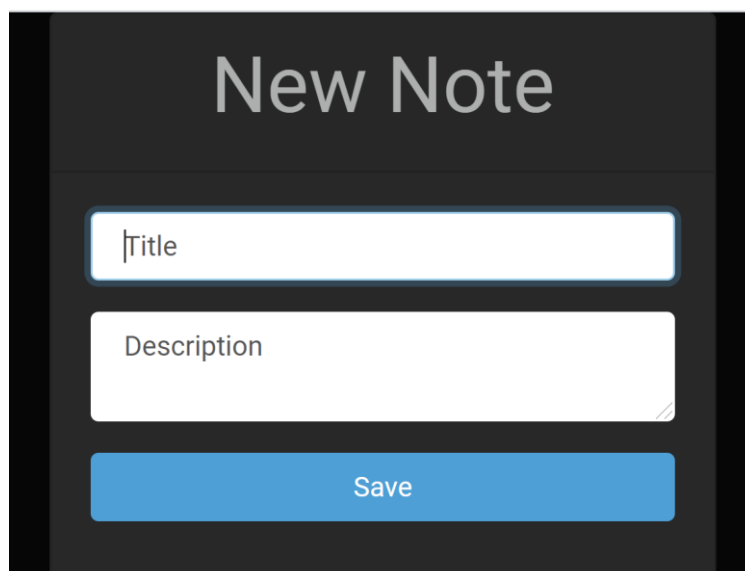
Z poziomu strony głównej (Rysunek 7.), użytkownik ma do wyboru kilka opcji: podgląd swoich notatek, dodanie nowej notatki lub wylogowanie. Opcja czytania notatek przekierowuje

na stronę, gdzie wyświetlane są wszystkie notatki danego użytkownika. Strona z notatkami nie umożliwia bezpośrednio podglądu notatki ze względu na bezpieczeństwo (widoczne są tytuł, data oraz zaszyfrowana treść). Dla każdej notatki stworzone zostały funkcje: podgląd oraz usunięcie. Dla ułatwienia obsługi umieszczony jest tu także również przycisk, który odpowiada za dodanie kolejnego wpisu oraz przycisk wylogowania.



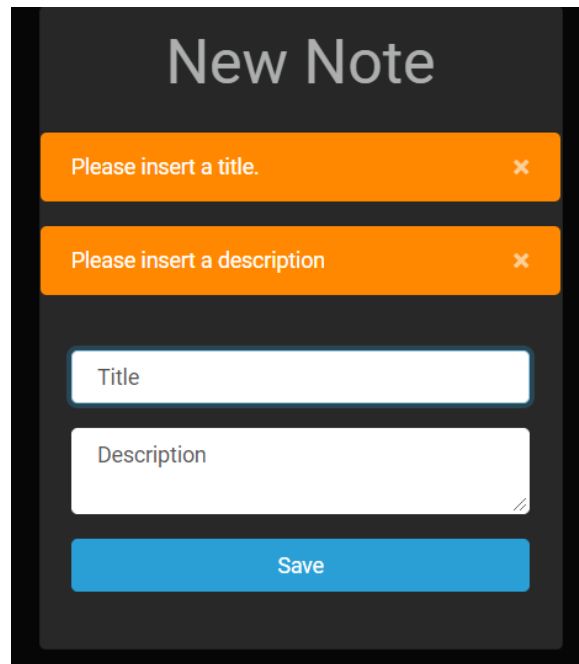
Rysunek 8. Zrzut ekranu strony ze wszystkimi notatkami użytkownika

Funkcja podglądu (Przycisk: „Read (not working)”) aktualnie, ze względu na niedopracowaną komunikację z bazą w zakresie przesyłania kluczy oraz wektorów inicjujących do bazy i z bazy, nie funkcjonuje poprawnie.



Rysunek 9. Zrzut ekranu strony z możliwością dodanie nowej notatki

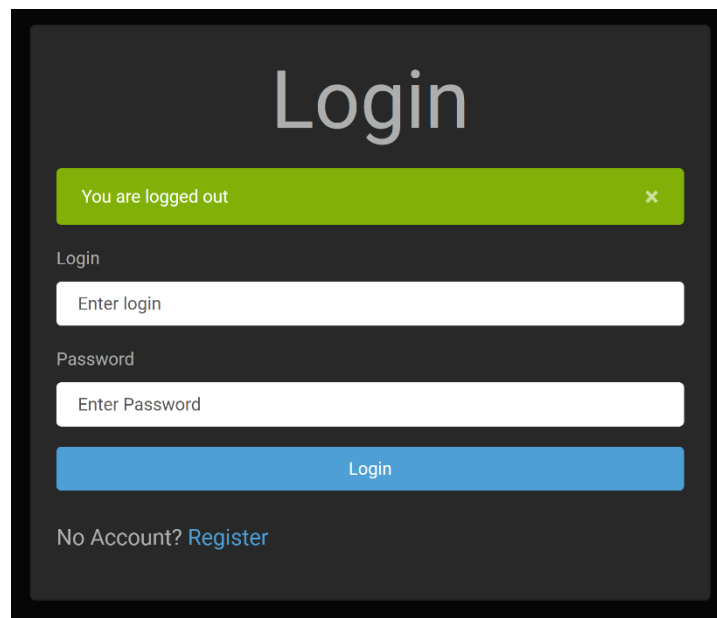
Tworząc nową notatkę użytkownik musi uzupełnić wszystkie pola inaczej możliwość dodania zostanie zablokowana (następuje odświeżenie strony, a użytkownik otrzymuje komunikat o błędzie).



The screenshot shows a 'New Note' form with a dark gray background. At the top, the title 'New Note' is displayed in white. Below the title, there are two orange error message boxes. The first box contains the text 'Please insert a title.' and the second box contains 'Please insert a description', both with a small 'x' icon to their right. Below these boxes are two white input fields: the first is labeled 'Title' and the second is labeled 'Description'. At the bottom of the form is a blue button labeled 'Save'.

Rysunek 10. Zrzut ekranu komunikatów o błędnym wprowadzeniu notatki

W przypadku poprawnego uzupełnienia użytkownik po dodaniu zostaje przekierowany do strony ze wszystkimi notatkami. Po zakończonej pracy użytkownik może wylogować się ze strony przy pomocy przycisku *Logout*, co potwierdza komunikat o poprawnym wylogowaniu (Rys. 10).

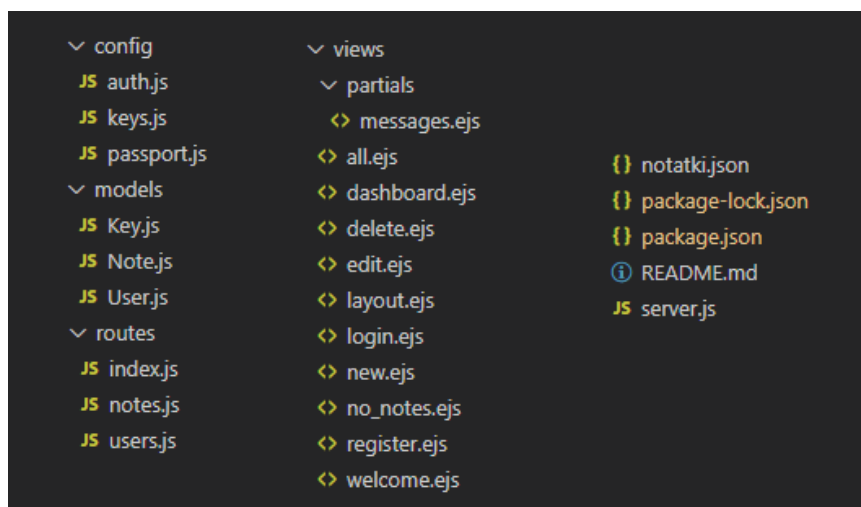


The screenshot shows a 'Login' form with a dark gray background. At the top, the title 'Login' is displayed in white. Below the title, there is a green message box containing the text 'You are logged out' and a small 'x' icon to its right. Below this box, the word 'Login' is written in small white text. There are two white input fields: the first is labeled 'Enter login' and the second is labeled 'Enter Password'. Below these fields is a blue button labeled 'Login'. At the bottom, the text 'No Account?' is followed by a blue link labeled 'Register'.

Rysunek 11. Zrzut ekranu po wylogowaniu

Implementacja rozwiązania

Podczas realizacji projektu utworzono katalogi z plikami, niezbędnymi do poprawnego działania programu.

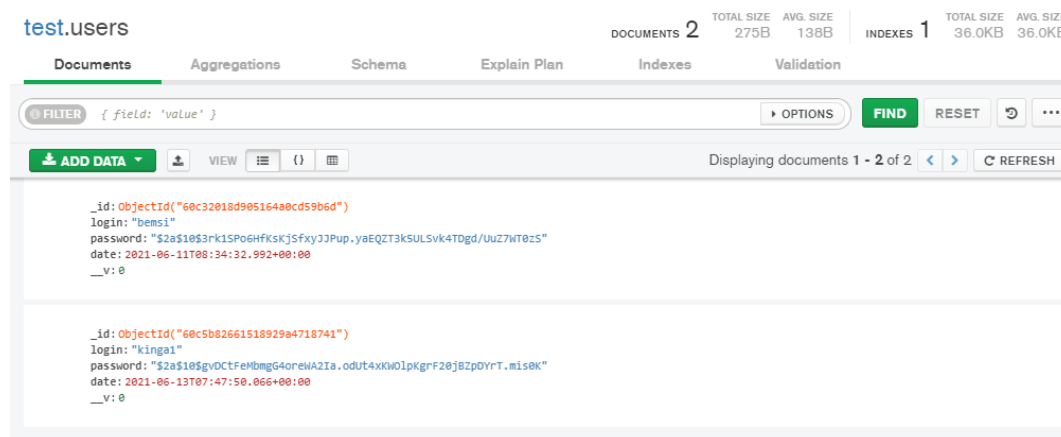


Rysunek 12. Utworzone pliki wraz z katalogami

Folder *config* odpowiada za pliki konfiguracyjne w projekcie. *Auth.js* to plik, w którym przechowywane są funkcje odnośnie autoryzacji. Następnie *keys.js*, który odpowiada za informacje potrzebne do połączenia się z bazą, natomiast *passport.js* – za konfigurację uwierzytelniania oraz weryfikacji podczas logowania. Kolejny folder *models* odpowiedzialny jest za przechowywanie informacji dotyczących struktury notatki, danych do rozszyfrowywania i użytkownika. Folder *routes* posiada pliki, w których opisane są poszczególne metody dla kolejnych podstron internetowych. Folder *views* zawiera pliki odpowiedzialne za widok poszczególnych stron. Plik *server.js* odpowiada za całe funkcjonowanie aplikacji.

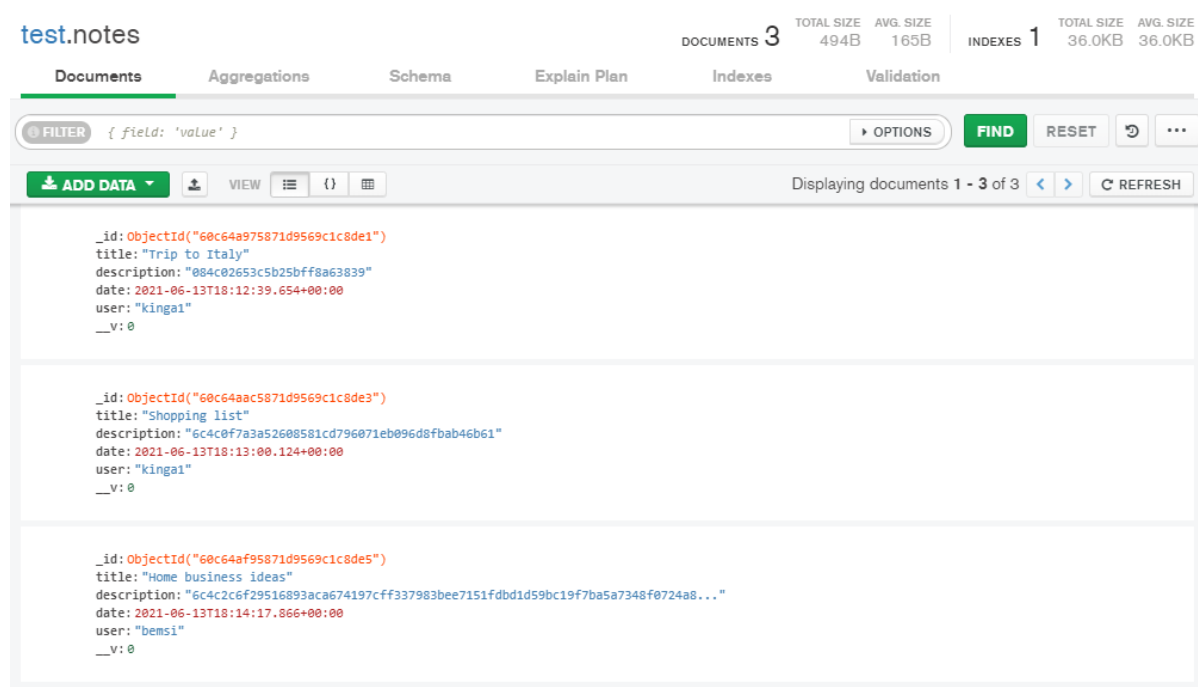
Baza danych

Aplikacja łączy się z bazą MongoDB, w której utworzono 3 kolekcje: *users*, *notes* i *keys*. Pierwsza z nich przechowuje dane niezbędne do logowania – loginy oraz zaszyfrowane hasła.



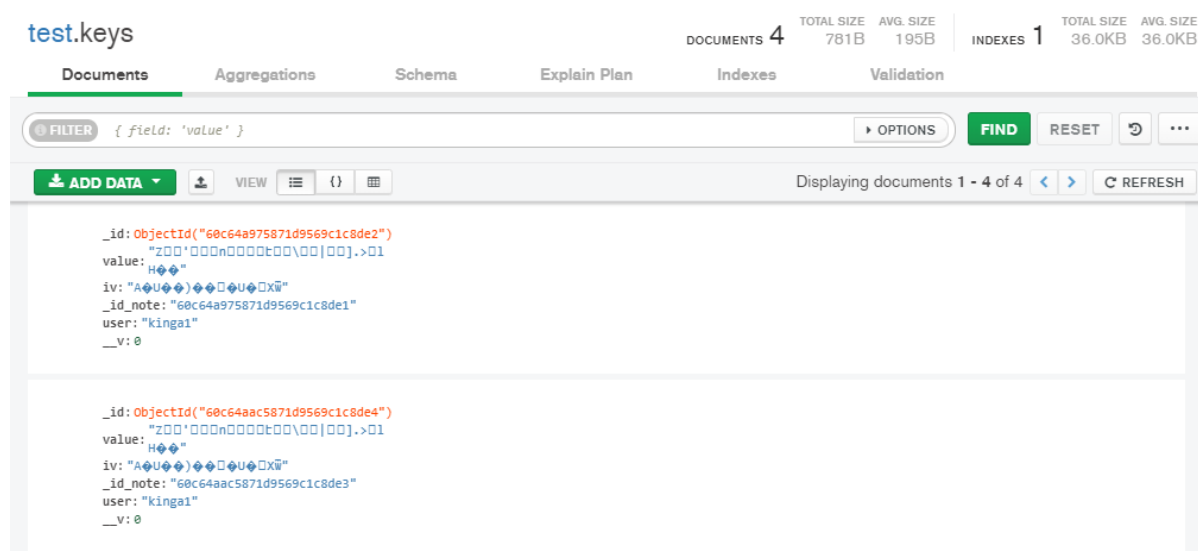
Rysunek 1313. Zrzut ekranu przykładowych danych w kolekcji Users

Kolekcja *notes* zawiera dane dotyczące notatek wszystkich użytkowników – tytuły, zaszyfrowane treści, daty utworzenia notatek oraz ich autorów.



Rysunek 1414. Zrzut ekranu przykładowych danych w kolekcji Notes

W kolekcji *keys* przechowywane są klucze szyfrujące, wektory inicjalizujące, id szyfrowanych notatek oraz loginy autorów.



Rysunek 151516. Zrzut ekranu przykładowych danych w kolekcji Keys

Analiza ryzyka i przyjęte modele zabezpieczeń

Logowanie/Rejestracja użytkownika - hasło

Analiza ryzyka:

- Posiadane zasoby:
loginy i hasła w nierelacyjnej bazie danych
- Określenie potencjalnych naruszeń bezpieczeństwa:
aspekt fizyczny: uszkodzenie nierelacyjnej bazy danych, brak możliwości odczytania danych, brak tworzenia kopii zapasowych
aspekt osobowy: podszywanie się pod użytkownika, błędy ludzkie: wprowadzenie błędnych danych, udostępnienie hasła
aspekt programowy: przechwycenie danych w trakcie wprowadzania loginu i hasła
- Określenie stopnia zagrożenia:
aspekt fizyczny: średnie zagrożenie
aspekt osobowy: duże zagrożenie
aspekt programowy: duże zagrożenie

W czasie realizacji projektu przyjęto, że podczas rejestracji tworzone hasło będzie zaszyfrowane przy pomocy Hash z ziarnem i wysyłane jako ciąg znaków do nierelacyjnej bazy danych. Nazwa użytkownika (login) podczas rejestracji będzie informacją jawną również wysyłaną do nierelacyjnej bazy danych. Hash z ziarnem umożliwia tworzenie takich samych haseł dla innych loginów. Kolejnym atutem wykorzystania tej szyfryzacji jest fakt, że podczas wycieku danych ujawnienie haseł będzie trudniejsze niż w przypadku samego użycia Hash. Użytkownik podczas tworzenia konta *MUSI* zapamiętać swoje hasło, ponieważ nie można skorzystać z opcji *Przypomnij hasło*.

Podczas logowania użytkownika do swojej strony podaje on hasło. Podawane hasło podczas logowania jest poddawane funkcji haszującej i porównywane z wynikiem w bazie danych.

Czy zachowane są cele kryptografii?

Poufność – zapewnienie, że nikt nie posiadający odpowiedniej wiedzy nie będzie mógł uzyskać dostępu do informacji. W naszym wypadku jest to zapewnione, ponieważ podczas rejestracji tworzy się ciąg znaków, który jest trudny do obliczenia.

Integralność – zapewnienie odbiorcy, że odebrana informacja nie została zmodyfikowana od momentu nadania przez nadawcę. Ten przypadek rozumiemy, jako że podczas logowania danego użytkownika hasło nie zostało zmienione przez kogoś innego (użytkownik jako nadawca i odbiorca). W naszej aplikacji nie ma możliwości zmiany hasła, tym samym możliwość dokonania zmiany przez osoby trzecie jest mała.

Uwierzytelnienie – zapewnienie, że strona jest tą, za którą się podaje. Tu sprawdzamy, czy dany użytkownik to prawdziwy użytkownik. Uwierzytelnianie odbywa się poprzez podanie

hasła i loginu. W przypadku podania złego login/hasła wyskakuje powiadomienie o błędnych danych.

Niezaprzeczalność – brak możliwości wyparcia przez stronę udziału w procesie. Podczas tworzenia konta nie jesteśmy w stanie zweryfikować, czy osoba korzystająca z konta nie podszywa się pod nikogo. Nie jest to w pełni zachowane.

Szyfrowanie notatek

Analiza ryzyka

- Posiadane zasoby:
loginy, tytuły, data utworzenia, treść
- Określenie potencjalnych naruszeń bezpieczeństwa:
 - aspekt fizyczny*: uszkodzenie nierelacyjnej bazy danych, brak możliwości odczytania danych
 - aspekt osobowy*: podszywanie się pod użytkownika, błędy ludzkie: wprowadzenie błędnych danych, udostępnienie hasła, co daje dostęp do funkcji odszyfrowującej notatki
 - aspekt programowy*: przechwycenie transmisji, niewystarczająco zabezpieczone dane w bazie, błędy przy szyfrowaniu lub deszyfrowaniu notatek
- Określenie stopnia zagrożenia:
 - aspekt fizyczny*: małe zagrożenie
 - aspekt osobowy*: duże zagrożenie
 - aspekt programowy*: średnie zagrożenie

Do szyfrowania notatek użyty został szyfr blokowy CTR – *counter*. Wybór szyfru jest definiowany przez nas w aplikacji i w razie potrzeby stosowania innego zabezpieczenia – może zostać zmieniony. Szyfrowanie przeprowadzane jest w oparciu o funkcję biblioteki *crypto*, która umożliwia poprawne dwukierunkowe działanie: szyfrowanie i rozszyfrowywanie z użyciem wektora inicjującego oraz klucza. Notatka jest szyfrowana zanim zostanie przekazana do bazy, co zabezpiecza treść przed wyciekiem podczas ewentualnych ataków w czasie transmisji. Poprawne deszyfrowanie zaprezentowane jest w kodzie jedynie dla testów i na rzecz projektów, w docelowej aplikacji rozszyfrowywanie notatki następuje pojedynczo, po kliknięciu przycisku „Read” i w żadnym innym momencie. Podstawą bezpieczeństwa tego systemu jest także przechowywanie kluczy i wektorów inicjujących w odpowiedniej kolekcji. Nasze rozwiązanie wymaga jeszcze rozwoju w tym zakresie.

Czy zachowane są cele kryptografii?

Poufność – została zapewniona poprzez bezpieczne wyświetlanie notatek (jedynie po bezpośredniej prośbie o wyświetlenie danej notatki, ze zmniejszonym ryzykiem podglądania).

Integralność – nie została przez nas zapewniona. Dopilnowaliśmy jednak, by komunikacja z bazą była poprawna i na drodze baza-przeglądarka internetowa nie występowały błędy programowe.

Uwierzytelnienie – użytkownik tworząc notatkę musi być zalogowany, jest uwierzytelniany poprzez login i hasło. Login zostaje zapisany w bazie danych wraz z notatką. Nie istnieje jednak żadna inna forma uwierzytelnienia tworzenia notatki.

Niezaprzeczalność - podczas korzystania z konta nie jesteśmy w stanie zweryfikować, czy to właściciel konta korzysta z niego, ale tworzenie notatki na danym profilu jest uwierzytelnianie loginem użytkownika. Nie jest to w pełni zachowane.

Włamanie się do systemu

W przedstawionej powyżej analizie ryzyka, przedstawiłyśmy słabe punkty naszego systemu zabezpieczeń, które można byłoby wykorzystać do włamania się na konto użytkownika.

Instrukcja kompilacji, uruchamiania i korzystania

W pierwszej kolejności należy pobrać repozytorium projektu (link wyżej). Zaleca się otworzyć kod źródłowy projektu w Visual Studio Code. Kolejnym krokiem jest zainstalowanie odpowiednich pakietów (instrukcje do instalacji podane w pliku README.md). Następnie należy uruchomić aplikację poleceniem `npm run start`. Podczas uruchamiania pojawi się komunikat:

```
> app@1.0.0 start C:\Users\aleks\Desktop\Inżynieria Biomedyczna_sem6\BEMSI\projekt\bemsi_notatnik
> nodemon server.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
Example app listening at http://localhost:3000
MongoDB Connected
```

Rysunek 16. Komunikat przy uruchamianiu aplikacji

Link <http://localhost:3000> należy kliknąć i następnie w przeglądarce uruchomi się strona główna aplikacji. Korzystanie z aplikacji zostało opisane w rozdziale **Realizacja projektu**.

Rozwój aplikacji

Przede wszystkim, należałoby zadbać o poprawną deszyfrację notatek – jest to niezbędna funkcjonalność, która, niestety, nie działa poprawnie w zaproponowanym rozwiązaniu. Notatki, są poprawnie deszyfrowane przez dedykowane klucze, lecz niemożliwe jest wygenerowanie odszyfrowanych wartości do szablonu po przesłaniu kluczy do bazy.

W celu zabezpieczenia przed atakami typu brute-force, należałoby również wprowadzić ograniczoną liczbę prób logowania np. z pewnym, wydłużającym się z każdą niepoprawną próbą, czasem przerwy do kolejnej próby.

Obecnie, aplikacja nie umożliwia modyfikowania wprowadzonych wcześniej notatek, co bywa bardzo praktyczną funkcją, którą zdecydowanie warto by zaimplementować.

Bibliografia

- R.Kurjata, Bezpieczeństwo Medycznych Systemów Informacyjnych, wykłady (2021)
- Dokumentacje użytych bibliotek