Pracownia specjalistyczna z przedmiotu Cyberbezpieczeństwo (PS1)

Studia niestacjonarne II stopnia, kierunek Systemy Inteligentne

Autorzy: Tomasz Cekało, Adrian Kwiatkowski, Łukasz Gąsowski, Mateusz Siebiatyński

# 1. Zaimplementować prosty formularz logowania. Wykorzystać tabelę users. Hasło przechowywane w postaci jawnej.

Powstały projekt został stworzony z użyciem ASP.NET MVC. Jako bazę danych użyto SQLite.

A screenshot of a login page

AI-generated content may be incorrect.

Grupa zaimplementowała prosty formularz rejestracji i logowania. Jako dostawcę tożsamości wybraliśmy Microsoft.AspNetCore.Identity. Użyta paczka z jednej strony nie pozwala na przechowywanie haseł w postaci jawnej. Z drugiej strony hasła nie powinny być przetrzymywane w ten sposób, a raczej jako hash którego nie da się odtworzyć. Oto dlaczego:

* Jeśli atakujący zdobędzie dostęp do bazy danych, wtedy może odczytać je wszystkie jednocześnie.
* Wiele regulacji (takich jak RODO, HIPAA itd.) wymaga przechowywania haseł w sposób bezpieczny.
* Nawet jeśli uzyskanie dostępu do bazy danych z zewnątrz jest mało prawdopodobny, to złośliwi insiderzy (lub nieostrożni developerzy) mogą doprowadzić do wycieku haseł.
* Ochrona przed atakami „credential stuffing” - użytkownicy często używają tych samych haseł na różnych stronach.

2. Zaimplementować formularz wyświetlający wiadomości zapisane przez użytkowników - tabela messages.  
Stworzony został widok listujący wiadomości.

A screenshot of a computer

AI-generated content may be incorrect.

Wiadomości są przechowywane w tabeli Messages. Każda wiadomość ma Id, Text i Autora.

A screenshot of a computer program

AI-generated content may be incorrect.

# 3. Zaimplementować formularz umożliwiający dodawanie wiadomości przez zalogowanych użytkowników.

Zaimplementowany system pozwala na dodawanie nowych wiadomości. Jest on dostępny pod przyciskiem „Create New”:

A close up of a contact us

AI-generated content may be incorrect.

Jedynym polem które można w tym formularzu wypełnić jest tekst nowej wiadomości.

A screenshot of a computer

AI-generated content may be incorrect.

Pole autora, mimo że widoczne, jest nadpisywane przez backend, ponieważ potencjalny haker mógłby je nadpisać za pomocą developer tools przeglądarki.

# 4. Zaimplementować formularz umożliwiający usuwanie wiadomości poprzez ich twórców.

Został dodany formularz umożliwiający usuwanie wiadomości, dostępny pod tym przyciskiem:  
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Przycisk jest widoczny tylko dla autorów wiadomości:

A close up of text

AI-generated content may be incorrect.

# 5. Zaimplementować formularz umożliwiający nadawanie bądź odbieranie uprawnień do modyfikacji innym użytkownikom przez właściciela wiadomości.

Formularz do nadawania uprawnień znajduje się na liście wiadomości i jest dostępny tylko dla jej twórcy: A screenshot of a computer

AI-generated content may be incorrect.

W formularzu podaje się email użytkownika, który ma zostać autorem:  
A screenshot of a computer

AI-generated content may be incorrect.  
Po wejściu w szczegóły wiadomości możemy podejrzeć listę jej edytorów, a jeśli jesteśmy jej autorem, to widoczne będą dla nas też przyciski umożliwiające usunięcie edytora:

A screenshot of a computer

AI-generated content may be incorrect.

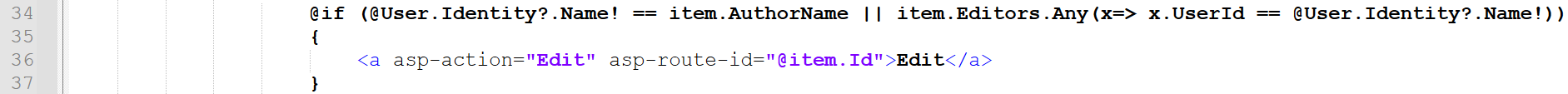
# 6. Zaimplementować formularz umożliwiający modyfikację wiadomości poprzez ich twórców bądź też użytkowników którzy mają do tego uprawnienia.

Formularz, który umożliwia edycję wiadomości jest identyczny jak formularz jej tworzenia:

A screenshot of a computer

AI-generated content may be incorrect.

przycisk do edycji jest widoczny tylko dla autorów i edytorów wiadomości:



7. Czy zaimplementowany mechanizm jest bezpieczny. Czy ustawienie wartości przycisków na disable można uznać za bezpieczne rozwiązanie?

Zaimplementowany przez nas mechanizm wydaje się bezpieczny. Mimo ograniczania widoczności przycisków (przedstawione na zrzutach ekranu), zespół postanowił dodać sprawdzanie uprawnień dostępowych do danych wiadomości także w backendzie, ponieważ potencjalny haker mógłby mimo wszystko uzyskać dostęp do tych formularzy modyfikując adresy URL przeglądarki. Na zrzucie ekranu dodawanie edytora do wiadomości: A screenshot of a computer program

AI-generated content may be incorrect.

User.Identity.Name to nazwa aktualnie zalogowanego użytkownika i . Podobne sprawdzanie uprawnień można znaleźć w wielu miejscach w kodzie.

Ustawianie przycisków na disable miałoby podobny efekt – można je zmodyfikować narzędziami deweloperskimi.

Problemem byłoby gdyby potencjalny atakujący mógł zmodyfikować swoją nazwę użytkownika, na przykład w ciasteczkach przeglądarki, jednak są one zaszyfrowane przez użytą bibliotekę:

A screenshot of a computer

AI-generated content may be incorrect.