

Vulnerabilities Analysis Report

Universidade de Aveiro(*UA*)

Guilherme Mendonça Claro,
Eduardo Lopes Ferreira,
João Afonso Pereira Ferreira,
Tiago Figueira Mostardinha



VERSAO 1

Project 1 - eHealth Corp

Security of Information and Organizations
2022/2023

João Paulo Barraca & André Zúquete

DETI - Departamento de Electrónica, Telecomunicações
e Informática

Mestrado Integrado em Engenharia de Computadores e
Telemática (MIECT) &
Licenciatura em Engenharia de Computadores e
Informática (LECI)

Guilherme Mendonça Claro,
Eduardo Lopes Ferreira,
João Afonso Pereira Ferreira,
Tiago Figueira Mostardinha

(98432) gui@ua.pt,
,(102648) edu.fernandes@ua.pt,
(103037) ferreiraafonsojoao@ua.pt,
,(103944) tiago.mostardinhas@ua.pt

13/11/2022

Índice

1	Introdução ao eHealth Corp	1
2	Objetivos	2
3	Common Weakness Enumeration(Common Weakness Enumeration (CWE))	4
3.1	[1] CWE's utilizadas:	4
3.1.1	CWE-79	4
3.1.2	CWE-89	4
3.1.3	CWE-256	4
3.2	Exploração das Vulnerabilidades	5
3.2.1	CWE-79	5
3.2.2	CWE-89	5
3.2.3	CWE-256	5
3.3	Como compor as vulnerabilidades	7
3.3.1	CWE-79	7
3.3.2	CWE-89	7
3.3.3	CWE-256	7
3.4	Outras Considerações sobre a nossa plataforma	7
4	Work Distribution	10

Lista de Figuras

1.1	eHealth Corp's WebSite	1
2.1	Docker Containers	3
3.1	<i>Login Page</i>	6
3.2	<i>Register Page</i>	6
3.3	<i>Make an Appointment being Logged in</i>	8
3.4	<i>Appointment's Code Confirmation being Logged in</i>	8
3.5	<i>Make an Appointment without being Logged in</i>	9
3.6	<i>Log Out Confirmation</i>	9

Capítulo 1

Introdução ao eHealth Corp

Nos dias de hoje, qualquer pessoa enquanto navegadora na *internet*, reconhece os perigos a ela associados, pelo que, existe a dificuldade de manter os dados de cada utilizador seguros, assim como existem bastantes vulnerabilidades nos diversos WebSites.

Desta forma, neste projeto implementou-se um simples WebSite, denominado por "*eHealth Corp*" de forma a evitar eventuais vulnerabilidades e não comprometer o nosso site e os dados dos nossos utilizadores. Dentro deste site, é possível o registo de novos utilizadores assim como o login dos já existentes na nossa plataforma. Ainda é admissível a marcação de uma consulta.

Para podermos denotar o efeito das vulnerabilidades, desenvolveu-se dois WebSites, um seguro e outro defeituoso, por outras palavras, um contra vulnerabilidades e outro sem qualquer tipo de proteção.



Figura 1.1: eHealth Corp's WebSite

Capítulo 2

Objetivos

- Desenvolver uma versão incorreta e outra correta da plataforma/WebSite;
- Entender o efeito fatal das vulnerabilidades num determinado WebSite;
- Corrigir e proteger ao máximo perante estas vulnerabilidades.

Containers Give feedback						
A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more						
<input type="checkbox"/>	NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
<input type="checkbox"/>	one-health	-	Running (2/2)			 
<input type="checkbox"/>	db-1 7e911fc83218	one-health-db:latest	Running	3306:3306	1 minute ago	 
<input type="checkbox"/>	api-1 1dc4731f71d1	one-health-api:latest	Running	63199:8080	1 minute ago	 

Figura 2.1: *Docker Containers*

Capítulo 3

Common Weakness Enumeration(CWE)

3.1 [1] CWE's utilizadas:

3.1.1 CWE-79

CWE-79(ou 'Cross Site Scripting') é uma vulnerabilidade em que o software neutraliza incorretamente o input na geração da pagina web. Esta vulnerabilidade aproveita-se do Javascript embutido do conteudo que a web page da display, podendo correr involuntariamente o codigo do lado do utilizador

3.1.2 CWE-89

CWE-89(ou 'SQL-Injection') é uma vulnerabilidade em que é neutralizado incorretamente pelo software os caracteres especiais usados em comandos SQL. Esta vulnerabilidade é bastante usada para que o atacante consiga aplicar código indesejado nas queries da base de dados, logo permite mostrar informação que era suposto estar segura e tambem permite apagar conteudo relvante das tabelas e mudar a informação protegida.

3.1.3 CWE-256

O CWE-256 é uma vulnerabilidade que se refere ao save de passwords em plain text, isto sendo, a base de dados guarda passwords com nenhuma proteção,logo, se atacante conseguir aceder a base de dados, terá acesso a conta de todos os utilizadores.

3.2 Exploração das Vulnerabilidades

3.2.1 CWE-79

Durante a geração de uma página, a aplicação não tem como prevenir a data de conter conteúdo que pode ser executável a partir de um Web Browser, como JavaScript, HTML tags and attributes, entre outros. Uma vez que o *script* prevém de uma página Web que foi enviada pelo servidor, o *Web Browser* de quem está a ser atacado executa o script malicioso no contexto do domínio do servidor da web.

E isto é vantajoso para o atacante, uma vez que, este é capaz de correr, pelo lado do servidor, código automaticamente ou até mesmo enganar o utilizador a inserir informação que noutro contexto este não partilharia

3.2.2 CWE-89

Esta vulnerabilidade tem como identidade o uso de pelicas ('), aspas ("), combinação de caractéres(-//) para comentar a query. O local de ato destas vulnerabilidades é na página de login(Fig. 3.1), onde se vai fazer as ditas *Sql Injections*. Ao inserir por exemplo -> username--// no input do username, o atacante, mesmo não conhecendo a *password*, o atacante é capaz de aceder à pagina, visto que o input colocado pelo mesmo comentará o campo onde o sistema iria buscar a *password*. Outros Exemplos:

- user=’bob’ AND TEST – //;
- ’ or 1 in (SELECT @@version) – //’;
- ’ or 1 in (select password from users where username = ’admin’) – //’;
(irá retornar a *hashed password* do utilizador denominado de ’admin’. Qualquer campo ou utilizador pode ser obtido)
- ’ or 1 in (select username from users where id=1) – //. (É possível conhecer todos os nomes de usuário referindo-os ao seu id. Outros campos podem ser usados.), etc.

O que é o atacante ganha? Ao injectar certas linhas de código o atacante obtém informação frágil e assim perturba o funcionamento da clínica. Em particular, acesso direto ao login(por login) e indireto (por descobrir a *password*) a contas de users.

3.2.3 CWE-256

’Storing a password in plaintext may result in a system compromise.’
Esta vulnerabilidade permite visualizar em *plaintext* a *password* de um determinado utilizador, na medida em que utiliza *SQLInjection* em conjunto com a vulnerabilidade referida anteriormente (CWE-89). Para prevenir este acontecimento utilizou-se uma cifra, denominada por SHA-256, onde o atacante é impossibilitado de ver a

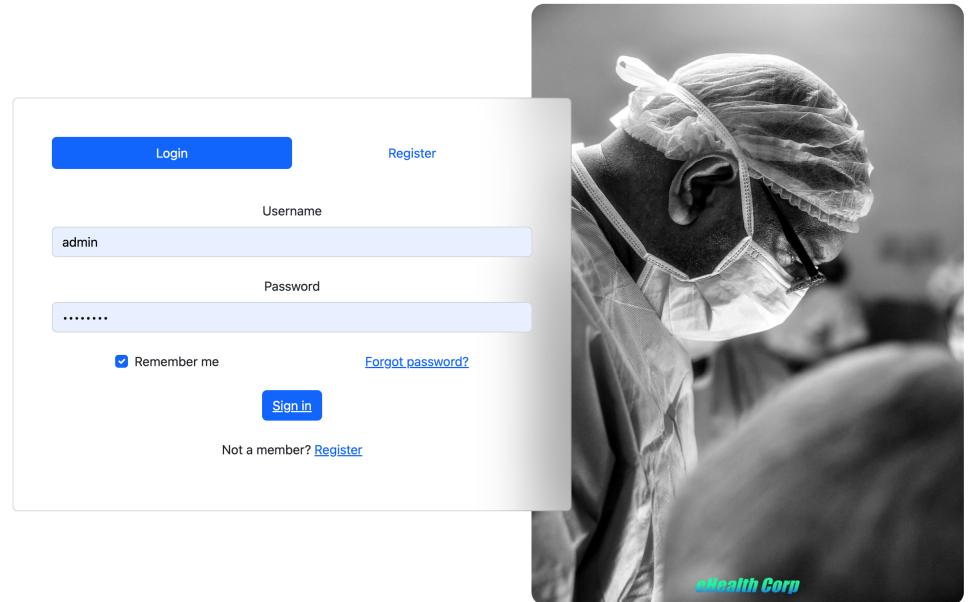


Figura 3.1: *Login Page*

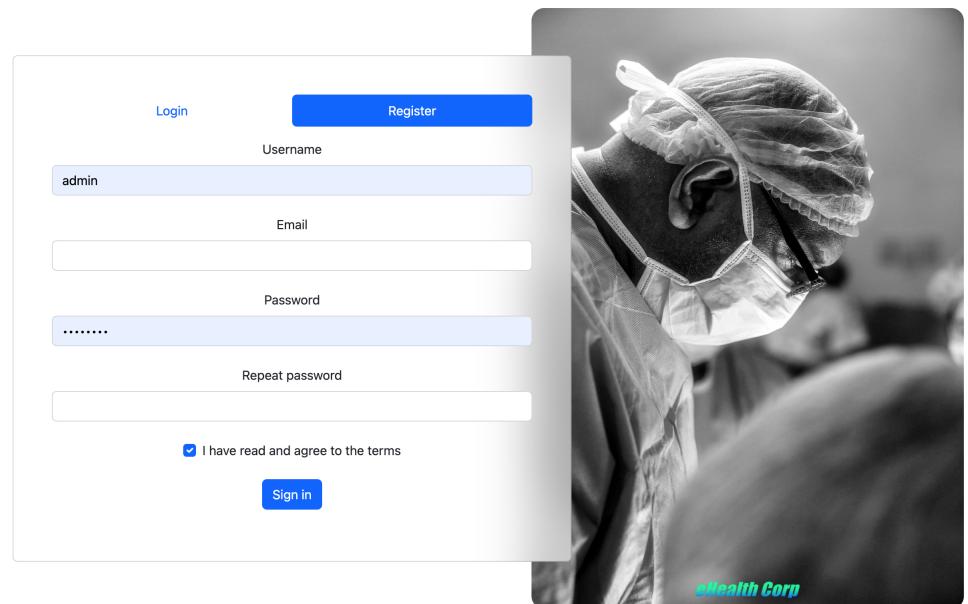


Figura 3.2: *Register Page*

password, por esta estar cifrada.

Tal acontece em todos os campos em que se é possível fazer uma *SQLInjection*. Caso o atacante pretenda visualizar as *passwords* na base de dados, o site seguro não o deve permitir porque ninguém deve poder aceder as *passwords* de terceiros. O objetivo é garantir que **ninguém** consegue visualizar as *passwords*.

Obviamente que, com este poder, o atacante poderá fazer-se passar pelo utilizador e aceder à sua conta, permitindo então marcar consultas e obter resultados da consulta desse mesmo paciente.

3.3 Como compor as vulnerabilidades

3.3.1 CWE-79

Para não permitir o *Cross-Site-Scripting* de ser feito pelo o atacante teremos desabilitar o uso de certos caracteres tornando-os invalidos.

3.3.2 CWE-89

Para garantir que o atacante não consiga realizar a *SQL Injection*, temos de "sanitizar" o input, isto sendo, certos carateres crtivos como -//,' e ",têm de ser substituidos por outros carateres que não permitem o ataque. Tambem tendo de ajustar a forma das queries.

3.3.3 CWE-256

Para dar a volta a esta vulnerabilidade tivemos que cifrar as *passwords* no momento de criação da conta, portanto a password enviada para a *database* já está cifrada, logo se algum atacante conseguir o acesso a base de dados, ele não vai saber quais são as *passwords* dos users

3.4 Outras Considerações sobre a nossa plataforma

Se o utilizador tiver feito anteriormente o Login na nossa plataforma, não irá ter problemas a marcar a sua consulta, contudo, caso não o tenha feito, não vai conseguir marcar a sua consulta, uma vez que a este não lhe será atribuído o código de confirmação (Fig. 3.4). Como podemos identificar na imagem seguinte (Fig. 3.3), a partir do canto superior direito é possível observar que o utilizador denominado "admin" tem acesso a todas as funcionalidades do WebSite.



Figura 3.3: *Make an Appointment* being Logged in

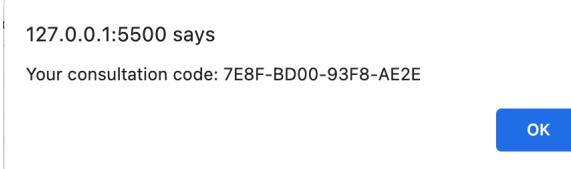


Figura 3.4: *Appointment's Code Confirmation* being Logged in

Assim que se submete o pedido da marcação da consulta é-se fornecido um código de confirmação da marcação da consulta (Fig. 3.4). No caso do utilizador não estiver Logged in na nossa plataforma, como é possível observar-se no canto superior direito é apresentado o botão de "Login/Register", confirmando que este não tem acesso às finalidades do WebSite (Fig. 3.5). Uma vez que não tem acesso, não será possível marcar a consulta pretendida pelo mesmo, pelo que a página irá fazer *reload* e não terá acesso ao código de confirmação.

Se o utilizador pretender fazer *Logout* da sua conta, basta 'carregar' no botão superior direito vermelho e será imprimida, em forma de popUp, uma mensagem de confirmação (Fig. 3.6)

The screenshot shows a web application interface for making an appointment. At the top, there is a navigation bar with the eHealthCorp logo, Home, About Us, Use Code, and a Login / Register button. The main title is "Make an Appointment". Below the title, there is a form with the following fields:

- A dropdown menu set to "Dental".
- A text input field containing "910923144".
- A message area containing:

Good Morning Dr. Ferreira,
I would like to make a Dental's Appointment for the next week, if possible next Wednesday
(23/11/2022).
Cumpliments,
Olivia Rodrigo
- A date input field showing "23/11/2022" with a calendar icon.
- A green "Submit Request" button.

Figura 3.5: *Make an Appointment without being Logged in*



Figura 3.6: *Log Out Confirmation*

Capítulo 4

Work Distribution

- Eduardo Fernandes: 30%
- Guilherme Claro: 25%
- João Ferreira: 30%
- Tiago Mostardinha: 15%

Bibliografia

[1] DHS, online from <https://cwe.mitre.org/>, 2022.

Acrónimos

MIECT Mestrado Integrado em Engenharia de Computadores e Telemática

LECI Licenciatura em Engenharia de Computadores e Informática

CWE Common Weakness Enumeration