Tópicos Especiais em Segurança da Informação

TP8 - Explorar vulnerabilidades de segurança em um sistema web

Arthur do Prado Labaki

12-07, 2022

GBC 235

Informações adicionais

Nem todos os exercícios estão com imagem aqui no relatório, mas todas as imagens integradas nesse relatório, quanto códigos, planilhas ou gifs de demonstração estão em meu repositório no GitHub abaixo.
Link do meu GitHub

Observação

Esse laboratório foi realizado em sala de aula, sendo uma competição de *Capture The Flag*. Os três alunos que conseguirem maior pontuação ganharam uma barra de chocolate (eu fiquei em segundo). Sendo assim, esse relatório será dividido em duas partes.

Configuração e reconhecimento do sistema

Nessa primeira etapa da competição, todos os participantes tiveram que montar uma aplicação web em nuvem, utilizando a plataforma de nuvem Heroku. A aplicação a ser subida é o OWASP Juice Shop, que é uma aplicação web criada para treinamentos de segurança, demonstrações de conscientização, *CTFs* e como cobaia para ferramentas de segurança.

Ainda nessa primeira etapa, foi dado um tempo para observação do sistema, usado para identificar funcionalidades, layout da aplicação, além de seguir o "caminho feliz"que é um guia do próprio Juice Shop para demonstrar os pontos do sistema web.
(*https://pwning.owasp-juice.shop/part1/happy-path.html*).

O site que eu criei no Heroku para realizar essa competição está o link abaixo. Vale lembrar que a pontuação é atrelada aos *cookies* do navegador, então a pontuação do site será diferente para cada máquina.
Link para o meu Juice Shop

Explorar o sistema e identificar vulnerabilidades

Essa segunda parte consiste em realizar os exercícios da competição. Foram listados 12 exercícios comuns e 3 exercícios bônus que deveriam ser realizados antes do tempo acabar. Eu não consegui fazer o 5 e o último dos bônus em aula, porém aqui vou fazer todos para obter um melhor entendimento de vulnerabilidades.

Exercício 1)

Encontrar a página que mostra o placar dos desafios (é altamente recomendável que vocês comecem por aqui – o *score board* permitirá que vocês visualizem a lista de problemas e também visualizem dicas para resolvê-los).

Para encontrar essa página basta colocar */score-board* na *url* do site.
(*https://segurancactf.herokuapp.com//score-board*)

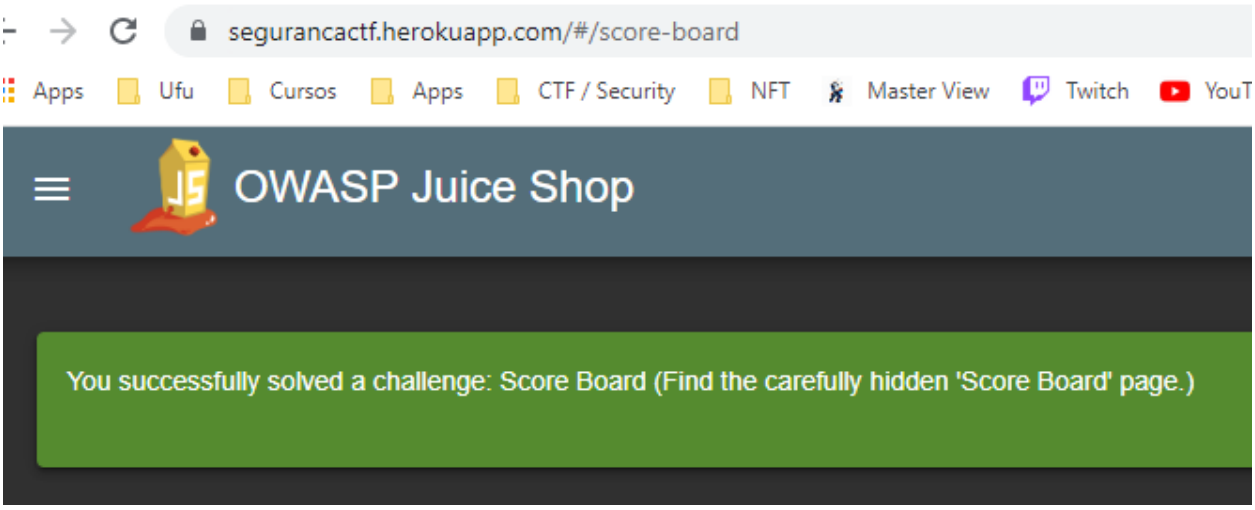

Figura 1: Resolução do exercício 1

Exercício 2)

Acessar um documento confidencial.

Analisando a página *robots.txt*, que é uma pagina para informar aos robots visitantes quais diretórios de um site não devem ser vasculhados por eles, é possível verificar que existe somente um diretório não permitido, o */ftp*. Nele é possível ver uma especie de *Index of/*, mostrando alguns arquivos e pastas do sistema. Nele é possível ver o arquivo *acquisitions.md* que é o arquivo confidencial que estamos procurando.

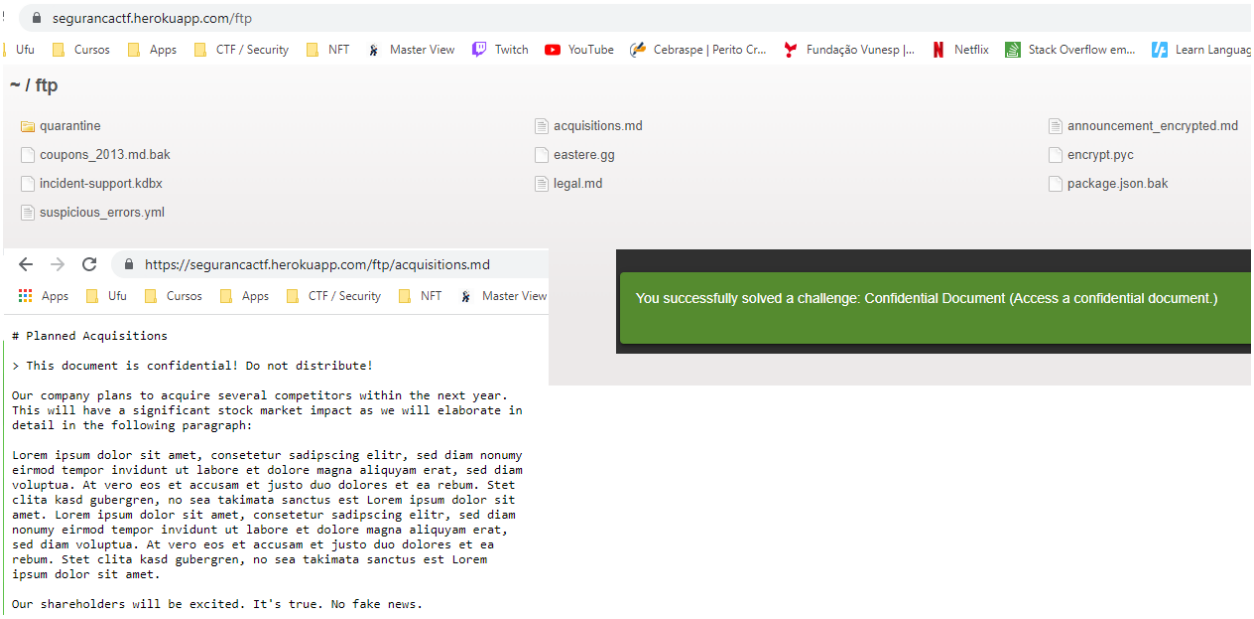

Figura 2: Resolução do exercício 2

Exercício 3)

Provocar um erro que não é devidamente tratado pela aplicação.

Como o sistema é propositalmente falho, deve ser possível gerar o erro utilizando strings não tratadas. Sabendo disso, basta tentar utilizar uma *string* de linguagem de banco de dados (como ') para conseguir esse erro.

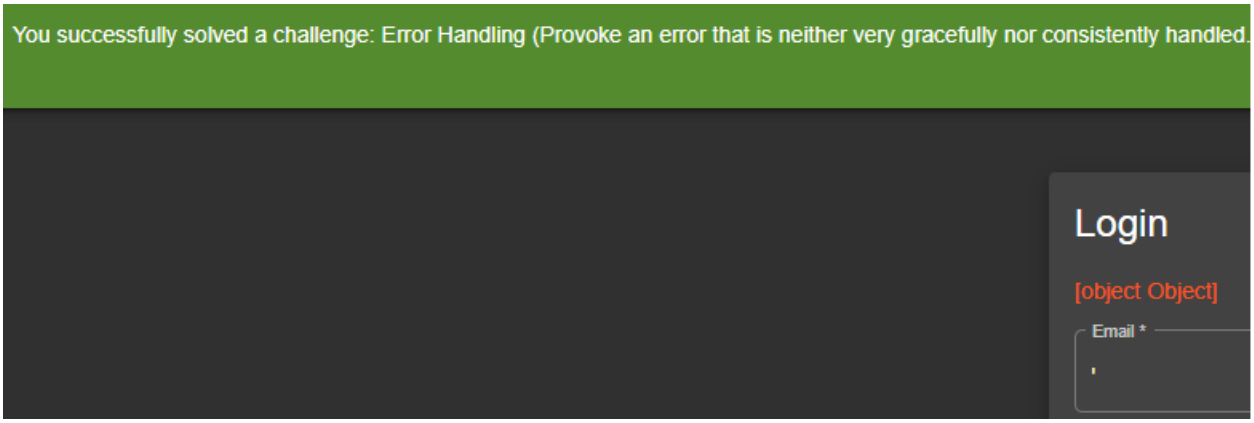

Figura 3: Resolução do exercício 3

Exercício 4)

Ler a política de privacidade/segurança da empresa.

Para isso basta logar em uma conta (ou criar uma e logar) e ir em *Account*, *Privacy Security* e *Privacy Policy*.

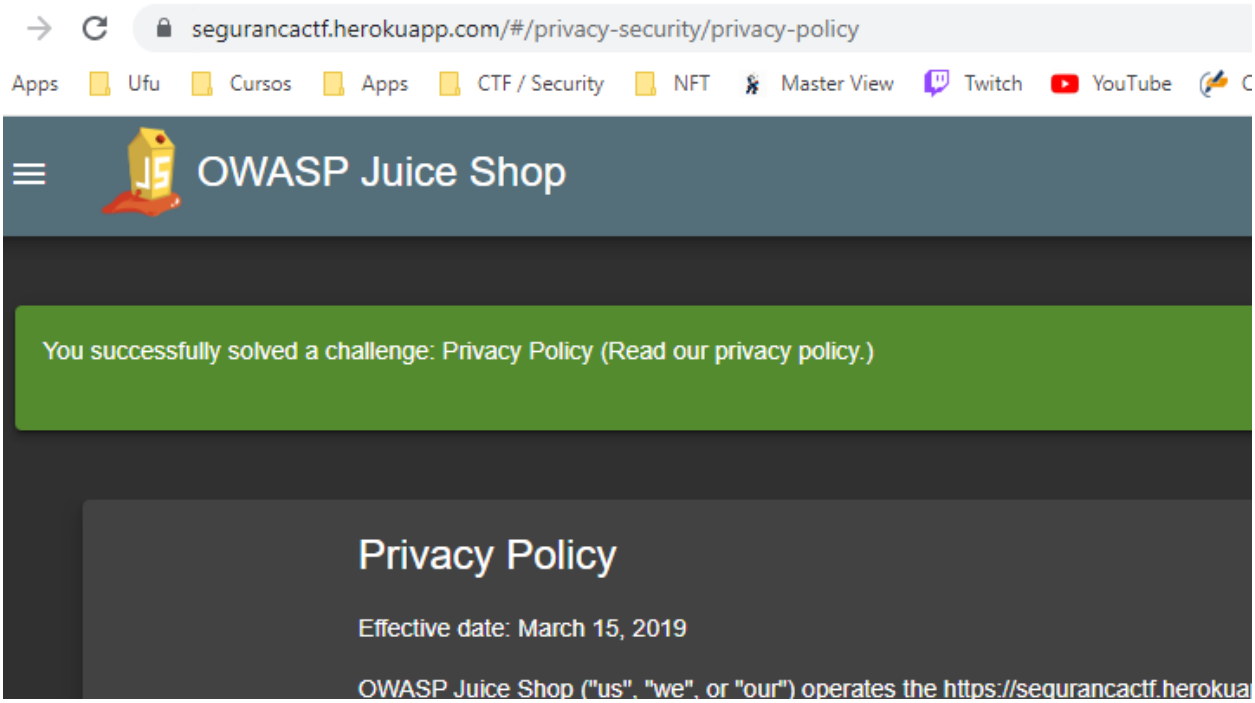

Figura 4: Resolução do exercício 4

Exercício 5)

Fazer o sistema redirecionar o usuário para um site de doação que está fora do ar.

Inspecionando o *Source* do *main.js* do site, buscando pela palavra *redirect*, encontramos "*./redirect?to=https://blockchain.info/address/1AbKfgvw9psQ41NbLi8kufDQTezwG8DRZm*", que é o site de doação fora do ar. Basta acessa-lo para concluir o desafio.
(*segurancactf.herokuapp.com/redirect?to=https://.....*)

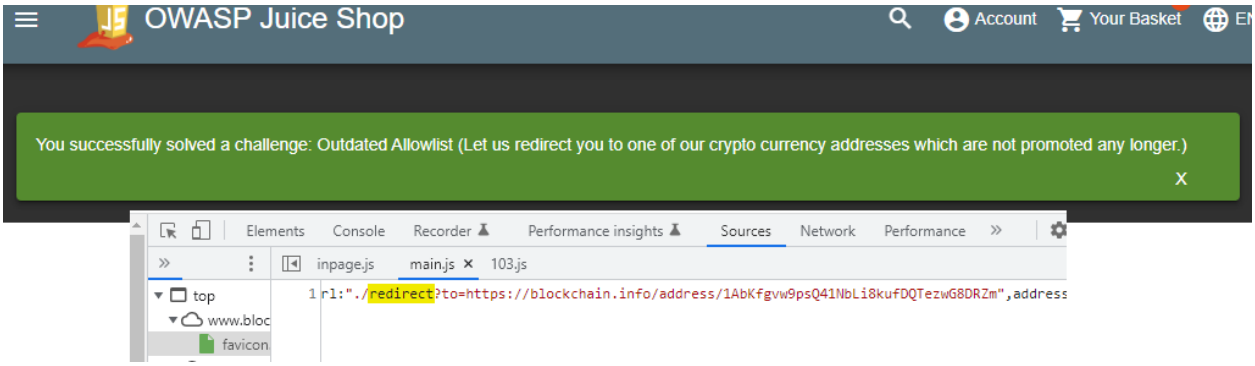

Figura 5: Resolução do exercício 5

Exercício 6)

Criar/registrar um usuário com senhas distintas nos campos "*Password*"e "*Repeat Password*".

Analisando a pagina criar conta, é possível perceber que somente o *"Repeat Password"* foi tratado. Então basca colocar a mesma senha nos dois campos e depois alterar o conteúdo do campo *"Password"*, criando a conta com as senhas diferentes.

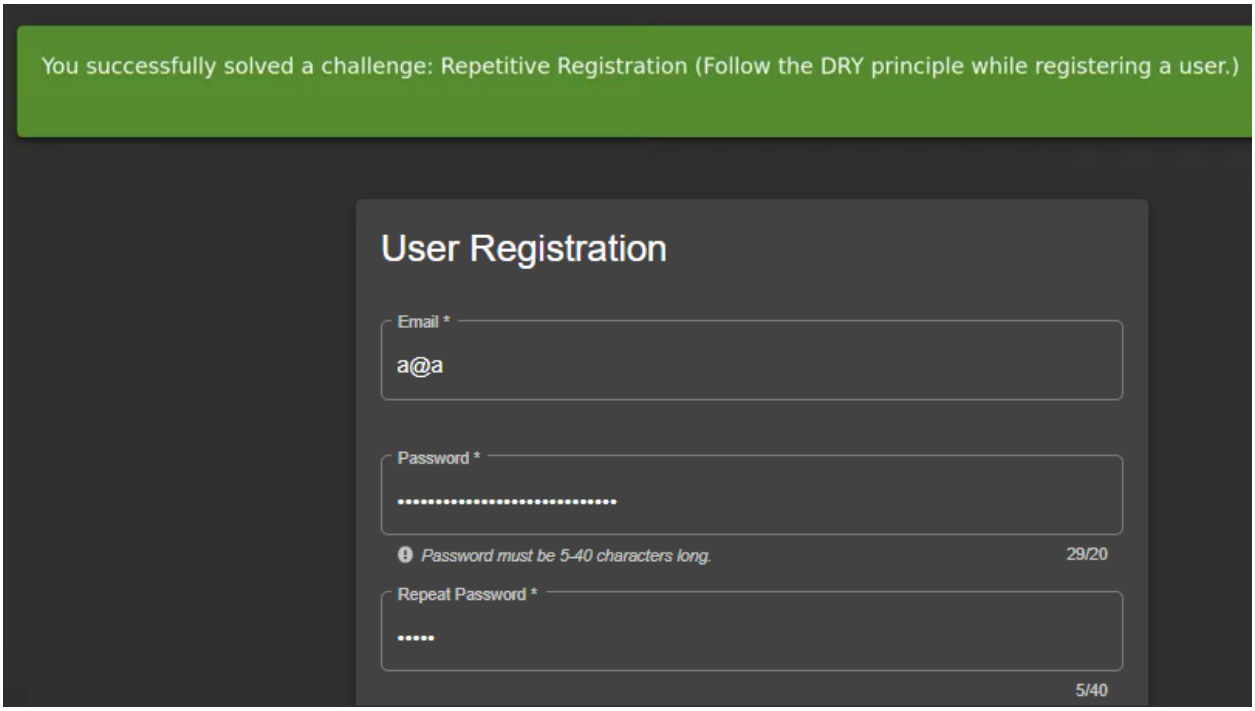

Figura 6: Resolução do exercício 6

Exercício 7)

Verifique os pontos onde o sistema é vulnerável a *XSS*. Use a seguinte *string*: *iframe src=”javascript:alert(‘xss‘)”*.

XSS ou *Cross-Site Scripting* é um tipo de vulnerabilidade do sistema de segurança de um computador, encontrado normalmente em aplicações web que ativam ataques maliciosos ao injetarem *client-side script* dentro de páginas web vistas por outros usuários. Nesse caso é um ataque *XSS* baseado em *DOM*, que é geralmente um ataque do lado do cliente, e a carga maliciosa nunca é enviada ao servidor. Isso torna ainda mais difícil a detecção de *Web Application Firewalls* e engenheiros de segurança que analisam os *logs* do servidor porque nunca veem o ataque.

No caso desse comando, ele só ira mandar um alerta na tela caso esse tipo de vulnerabilidade não foi tratada. Para isso, no campo de busca do site sera injetado essa *string* maliciosa, resolvendo o desafio.

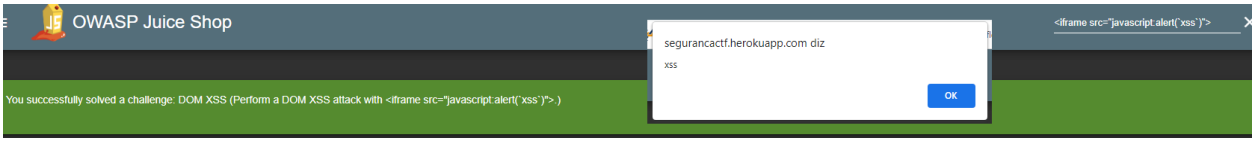

Figura 7: Resolução do exercício 7

Exercício 8)

Forneça uma nota zero na seção de *feedback* da loja.

Na página de *feedback*, analisando o código fonte dela é possível notar que existe diversos impedimentos para notas negativas, como mínimo sendo 1. Porém analisando o botão de *submit*, existe uma função que registra a nota (originalmente esta *disabled*), e que pode ser alterada, não enviando valor a nota atribuindo automaticamente um 0.

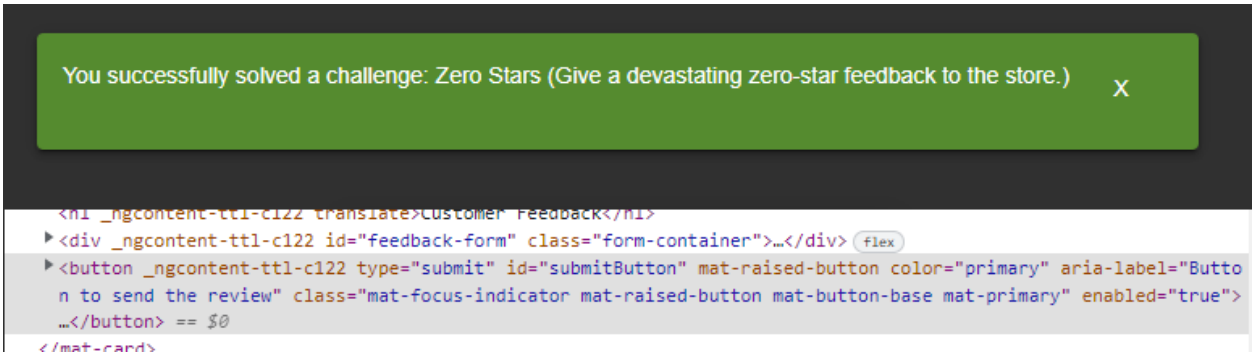

Figura 8: Resolução do exercício 8

Exercício 9)

Faça o login com a conta de administrador.

Existem duas maneiras diferentes de realizar esse exercício. Em sala eu utilizei *SQL Injection* utilizando o e-mail do administrador do site, que foi encontrado em uma *review* de um produto, colocando o código "*admin@juice-sh.op'- -* "no campo de e-mail e qualquer senha. Porém, estudando um pouco mais, é possível utilizar essa vulnerabilidade sem precisar de

qualquer e-mail, utilizando o código ” *’ or 1=1 - -* ”.

Explicando melhor, no primeiro comando, o apóstrofo serve para comunicar com o banco de dados e os dois hifens para ignorar o próximo campo (a senha), permitindo o acesso caso exista essa vulnerabilidade. Já no segundo comando, é alterado a busca do banco de dados, retornando o primeiro e-mail valido (*or 1=1* significando que sempre vai ser verdade) e ignorando o campo senha também.

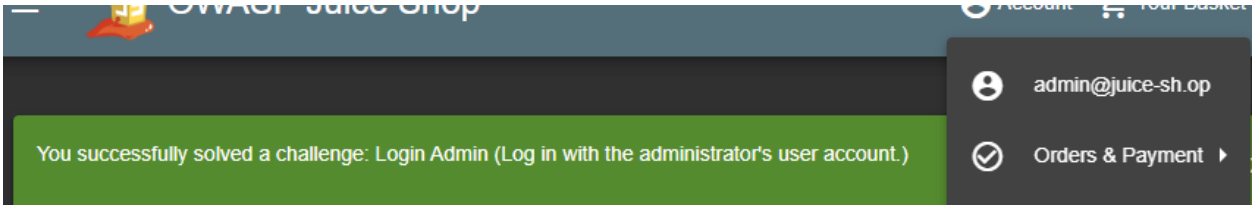

Figura 9: Resolução do exercício 9

Exercício 10)

Acesse a seção de administração da loja.

Inspecionando o *Source* da pagina *Main.js*, e pesquisando por *Path* é possível saber os caminhos que a *url* do site possui. Analisando bem, é possível encontrar um chamado *path:”administration”*, que é a seção de administração da loja. Vale lembrar que é preciso estar logado como administrador do site para poder acessar.

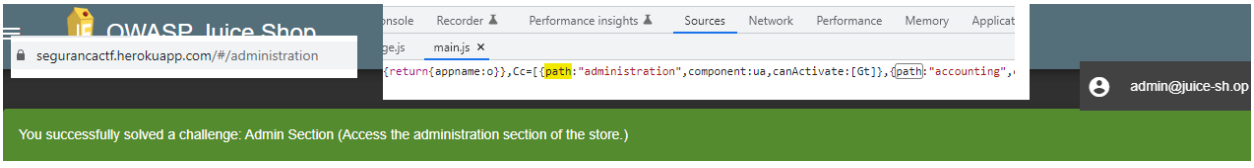

Figura 10: Resolução do exercício 10

Exercício 11)

Apague todos os *feedbacks* com 5 estrelas.

Quase como uma sequencia do exercício anterior, acessando a seção de administração da loja logado como administrador, é possível ver e excluir todos os *feedbacks*, resolvendo o desafio.

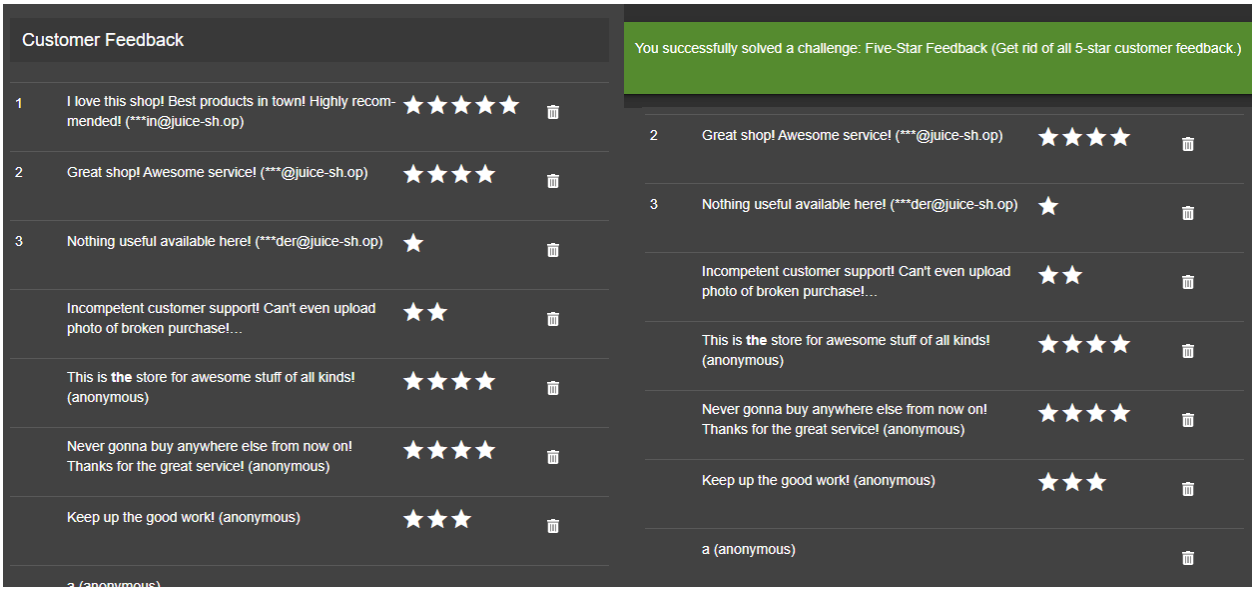

Figura 11: Resolução do exercício 11

Exercício 12)

Faça o login com a conta do usuário *MC Safesearch*.

Pesquisando no *Search* por 'MC' conseguimos encontrar o e-mail dele em uma *review* (*mc.safesearch@juice-sh.op*). Nesse desafio, não podemos utilizar *SQL Injection*, e ele é mais uma piadinha. Pesquisando o nome do usuário no Google, encontramos uma musica do *MC Safesearch* chamada *Protect Ya Passwordz* e nela ele afirma que a senha de seu login é o nome do seu cachorro (*Mr. Noodles*) com 0 no lugar do o.

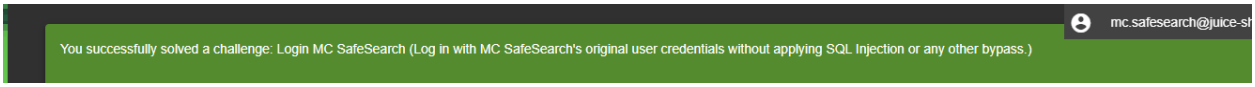

Figura 12: Resolução do exercício 12

Exercício Bônus 1)

Visualizar produtos no carrinho de compra de outros usuários.

Inspecionando o carrinho de compra, na aba *Alpication e Session Storage*, é possível ver o *bid* do usuário, que deve ser um identificador do usuário com o carrinho de compra dele. Alterando esse valor, conseguimos visualizar o carrinho de compra de outro usuário.

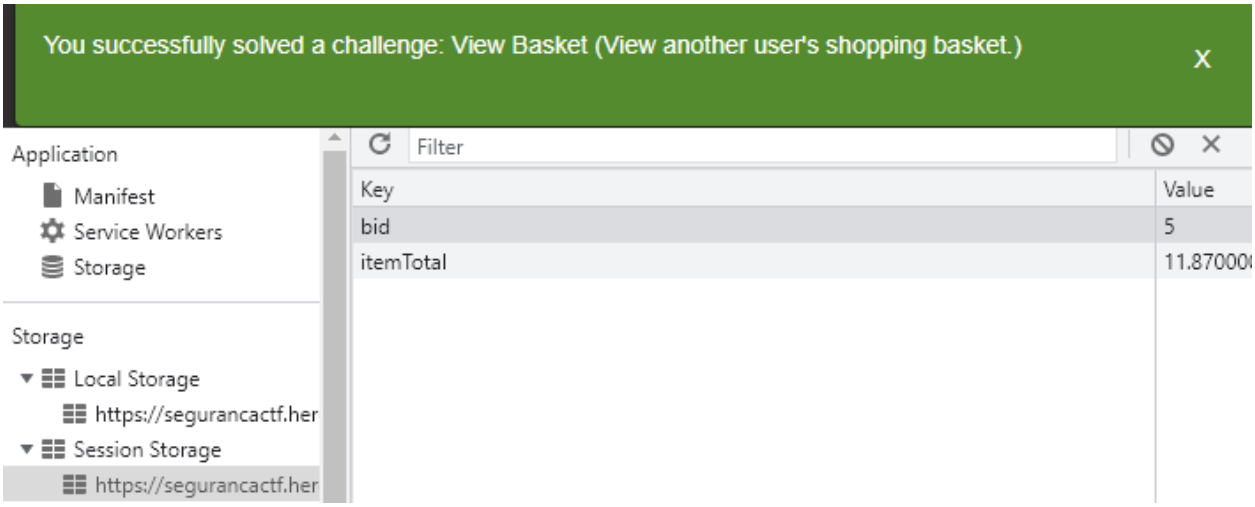

Figura 13: Resolução do exercício bônus 1

Exercício Bônus 2)

Postar *feedback* usando outro nome de usuário.

Na área de *feedbacks*, podemos ver que existe o campo e-mail e ele esta bloqueado. Para desbloquear, basta procurar esse campo no inspecionar do navegador e excluir a palavre *hidden*, deixando disponível para modificação.

Figura 14: Resolução do exercício bônus 2

Exercício Bônus 3)

Faça um pedido que te deixe rico! Ou seja, um pedido com valores negativos.

Analisando os pedidos em *Network* do navegador, é possível ver que, ao colocar e tirar elementos do carrinho, ele envia uma solicitação ao site informando a nova quantidade. Basta criar uma nova requisição, alterando essa quantidade para um valor negativo e solicitar ao site. Depois basta comprar o item normalmente para concluir esse desafio.

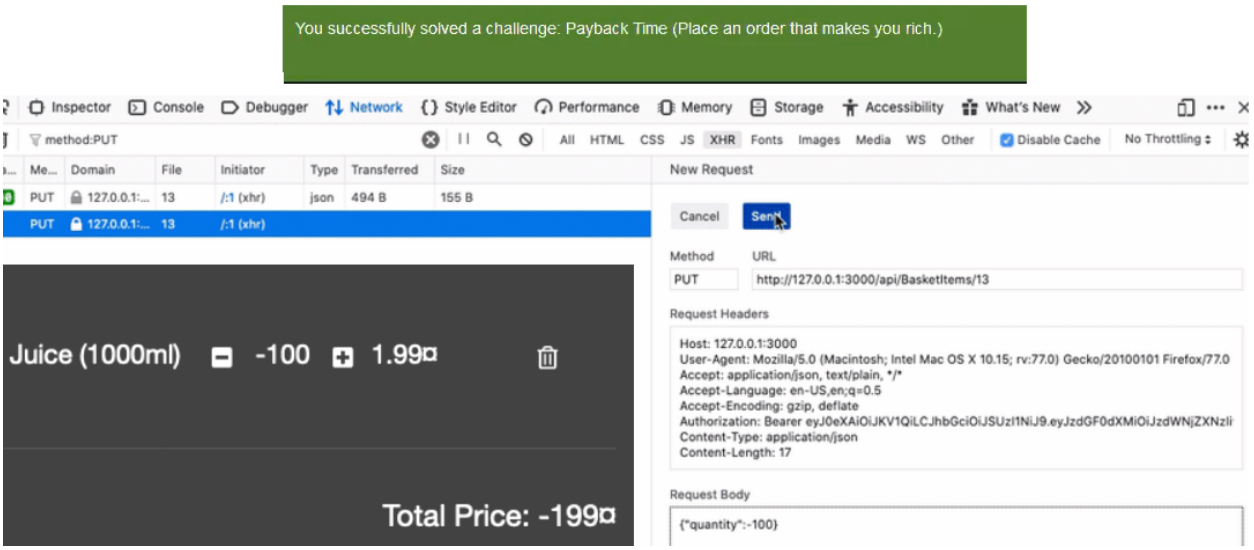

Figura 15: Resolução do exercício bônus 3