escola britânica de artes criativas & tecnologia

QUALIDADE DE SOFTWARE

Renan Yassumoto Ferreira

Relatório de Testes de Segurança
Juice Shop

Barueri
2026

Sumário

1. Introdução 3

2. Escopo e Ambiente 4

3. Vulnerabilidade 1 — Falha de validação de entrada (Injection) 5

3.1 Passo a passo da exploração 5

3.2 Evidências 5

3.3 Impacto 6

3.4 Recomendação 6

4. Vulnerabilidade 2 — Bypass de Autenticação (Login como Administrador) 7

4.1 Passo a passo da exploração 7

4.2 Evidências 7

4.3 Impacto 9

4.4 Recomendação 10

5. Vulnerabilidade 3 – Exposição de Informações Sensíveis por Falha de Controle de Acesso (IDOR / Broken Access Control) 11

5.1 Passo a passo da exploração 11

5.2 Evidências 11

5.3 Impacto 12

5.4 Recomendação 12

6. Resumo dos testes 13

7. Conclusão 13

1. Introdução

Este relatório tem como objetivo documentar a identificação e exploração de vulnerabilidades de segurança presentes na aplicação OWASP Juice Shop. A análise foi realizada com foco em simular o comportamento de um atacante, explorando falhas de validação e controle de entrada de dados, sem o uso de ferramentas automatizadas.

2. Escopo e Ambiente

Aplicação: OWASP Juice Shop

Ambiente:

- Versão hospedada (demo.owasp-juice.shop)
- Versão local via Docker (quando aplicável)

Ferramentas utilizadas:

- Navegador (Edge)
- DevTools do navegador (Console e Network)

Os testes foram realizados utilizando o navegador web com auxílio das ferramentas de desenvolvedor (DevTools), explorando funcionalidades expostas pela própria aplicação, sem uso de scanners ou ferramentas externas.

Metodologia

A metodologia adotada consistiu na análise manual da aplicação, identificando superfícies de ataque como campos de entrada de dados e parâmetros de requisição. Foram inseridos valores inesperados e payloads simples com o objetivo de observar comportamentos anômalos e identificar possíveis falhas de segurança.

3. Vulnerabilidade 1 — Falha de validação de entrada (Injection)

Foi identificado que o campo de busca da aplicação não realiza validação adequada da entrada do usuário, permitindo a inserção de caracteres especiais e expressões lógicas que são processadas pela aplicação sem bloqueio.

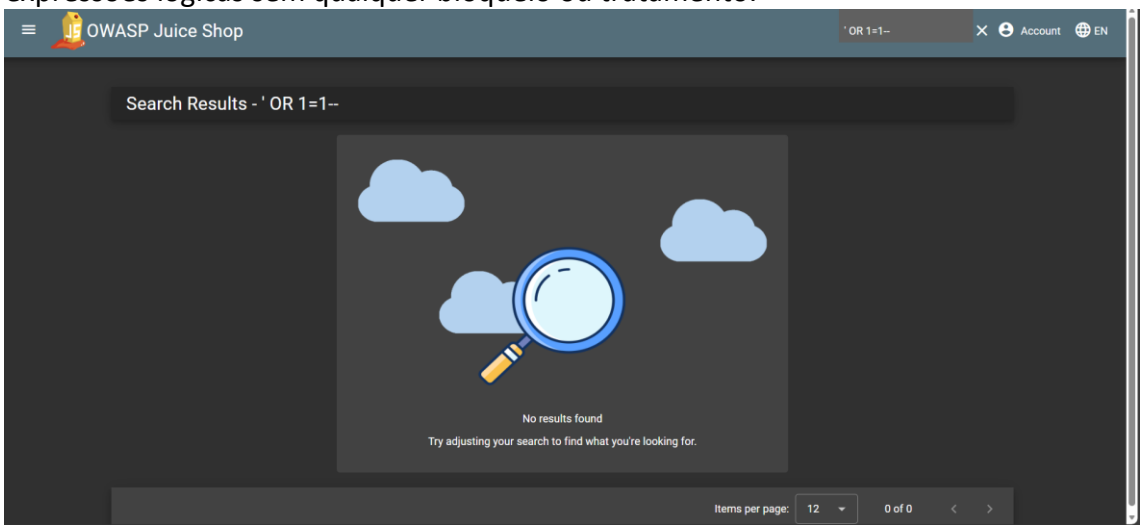
3.1 Passo a passo da exploração

1. Acessei a página inicial da aplicação.
2. Utilizei o campo de busca.
3. Inseri a busca apenas o caractere '.
4. Inseri a busca de ' OR 1=1--.
5. Observei comportamento inesperado na aplicação.

3.2 Evidências

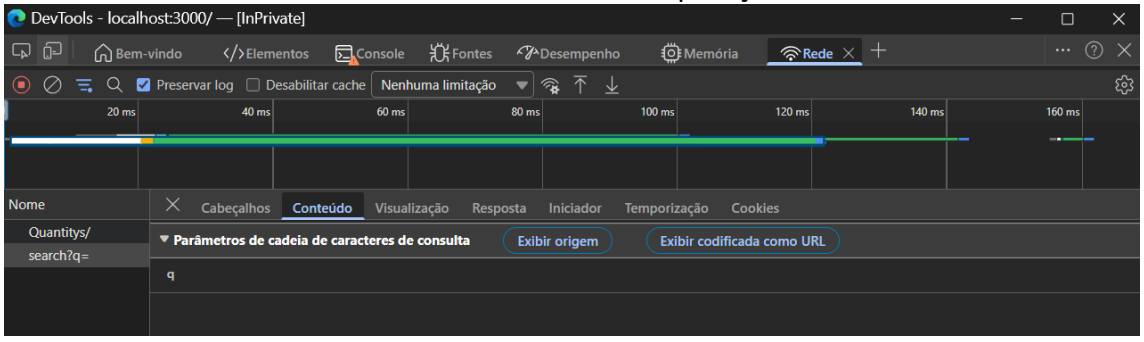
Evidência 1 — Evidência visual da falha no campo de busca

Resultado da funcionalidade de busca da aplicação OWASP Juice Shop exibindo o payload ' OR 1=1-- refletido diretamente na interface do usuário. Esse comportamento evidencia a ausência de validação e sanitização da entrada, permitindo a inserção de expressões lógicas sem qualquer bloqueio ou tratamento.



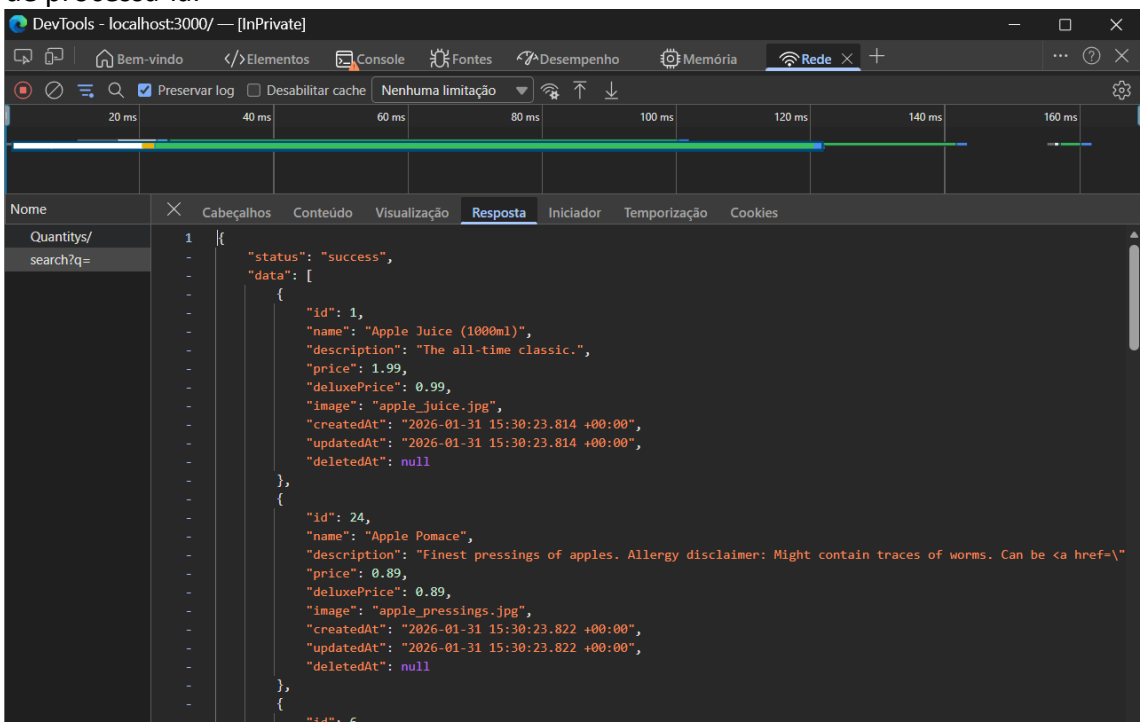
Evidência 2 — Evidência do envio do payload via requisição HTTP

Requisição HTTP capturada por meio das ferramentas de desenvolvedor do navegador (DevTools), demonstrando o envio do payload ' OR 1=1-- através do parâmetro de consulta ‘q’ na URL da funcionalidade de busca, confirmando que a entrada maliciosa foi encaminhada ao backend da aplicação.



Evidência 3 — Evidência da aceitação da entrada pelo backend

Resposta da API da aplicação retornando status de sucesso ("status": "success") e dados em formato JSON após o processamento da requisição contendo o payload malicioso, indicando que a aplicação não realizou validação adequada da entrada antes de processá-la.



A combinação das evidências apresentadas demonstra que a aplicação processa entradas não confiáveis fornecidas pelo usuário sem validação apropriada, caracterizando uma falha de segurança do tipo Injection / Falha de Validação de Entrada.

3.3 Impacto

A ausência de validação de entrada pode permitir ataques de Injection, possibilitando manipulação de consultas no backend, exposição de dados sensíveis e comprometimento da integridade da aplicação em cenários reais.

3.4 Recomendação

Recomenda-se a implementação de validação e sanitização de entradas do usuário, utilização de consultas parametrizadas e adoção de mecanismos de segurança no backend para prevenir ataques de Injection.

4. Vulnerabilidade 2 — Bypass de Autenticação (Login como Administrador)

Foi identificada uma falha no mecanismo de autenticação da aplicação OWASP Juice Shop que permite o bypass do processo de login. A aplicação não realiza validação adequada das entradas fornecidas durante a autenticação, possibilitando a manipulação da lógica de verificação de credenciais.

Como resultado, é possível obter acesso não autorizado a contas privilegiadas, incluindo a conta de administrador, sem o conhecimento de credenciais válidas. A falha identificada caracteriza uma vulnerabilidade crítica de autenticação, pois permite acesso administrativo sem credenciais válidas, representando um risco elevado à segurança da aplicação.

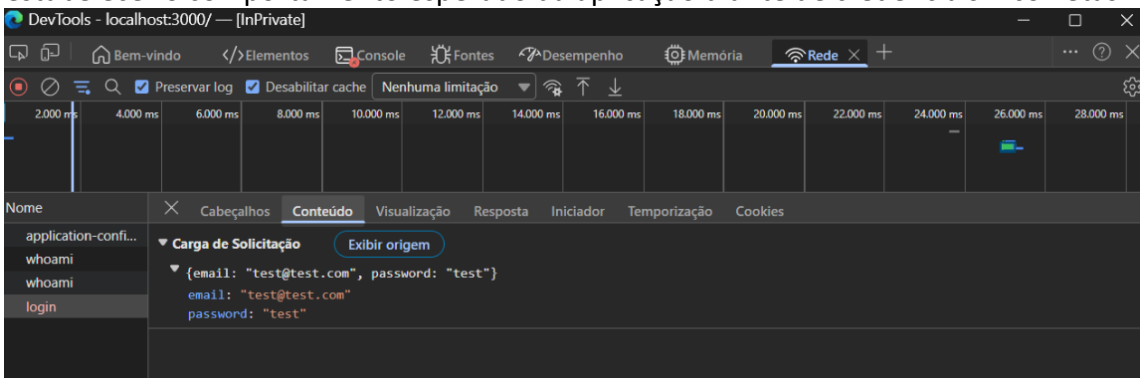
4.1 Passo a passo da exploração

1. Acessei a funcionalidade de login da aplicação.
2. Realizei uma tentativa de autenticação, utilizando credenciais inválidas, confirmando o comportamento esperado de bloqueio de acesso.
3. Em seguida, inseri um payload malicioso (' OR 1=1--) no campo de email, acompanhado de um valor arbitrário no campo de senha.
4. A requisição de autenticação foi enviada e capturada por meio das ferramentas de desenvolvedor do navegador (DevTools).
5. A aplicação retornou uma resposta de sucesso, fornecendo um token de autenticação válido.
6. Após a autenticação, foi confirmado o acesso à conta administrativa por meio da interface da aplicação e do endpoint ‘whoami’.

4.2 Evidências

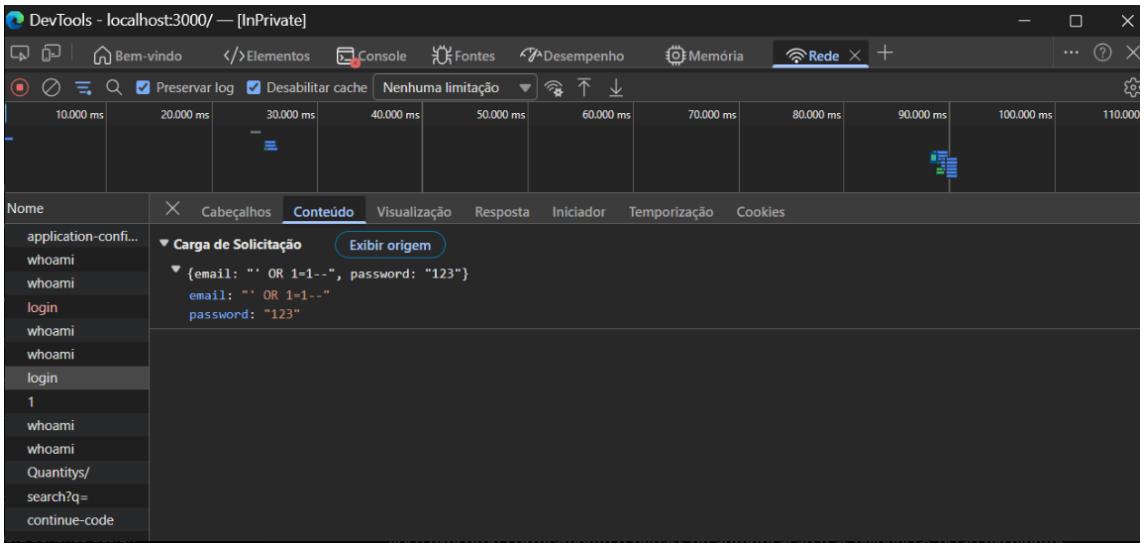
Evidência 1 — Tentativa inicial de login sem sucesso (baseline)

Tentativa de autenticação utilizando credenciais inválidas (test@test.com / test), resultando na resposta “Invalid email or password”. Essa etapa foi realizada para estabelecer o comportamento esperado da aplicação diante de credenciais incorretas.



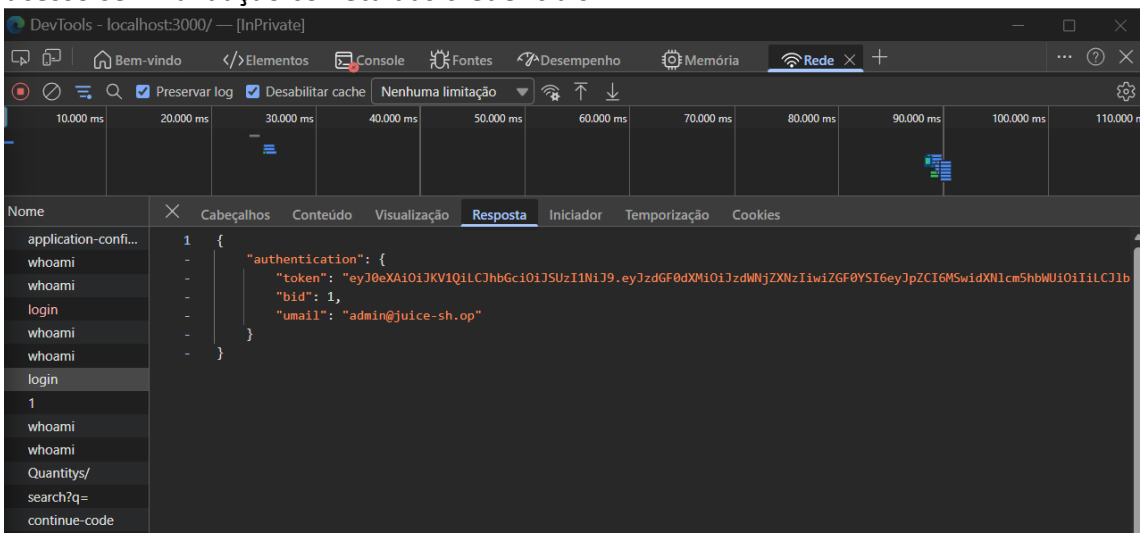
Evidência 2 — Envio de payload malicioso no campo de email

Requisição HTTP de autenticação capturada via DevTools, demonstrando o envio do payload ' OR 1=1-- no campo de email e um valor arbitrário no campo de senha, evidenciando a ausência de validação adequada dos dados enviados ao backend durante o processo de login.



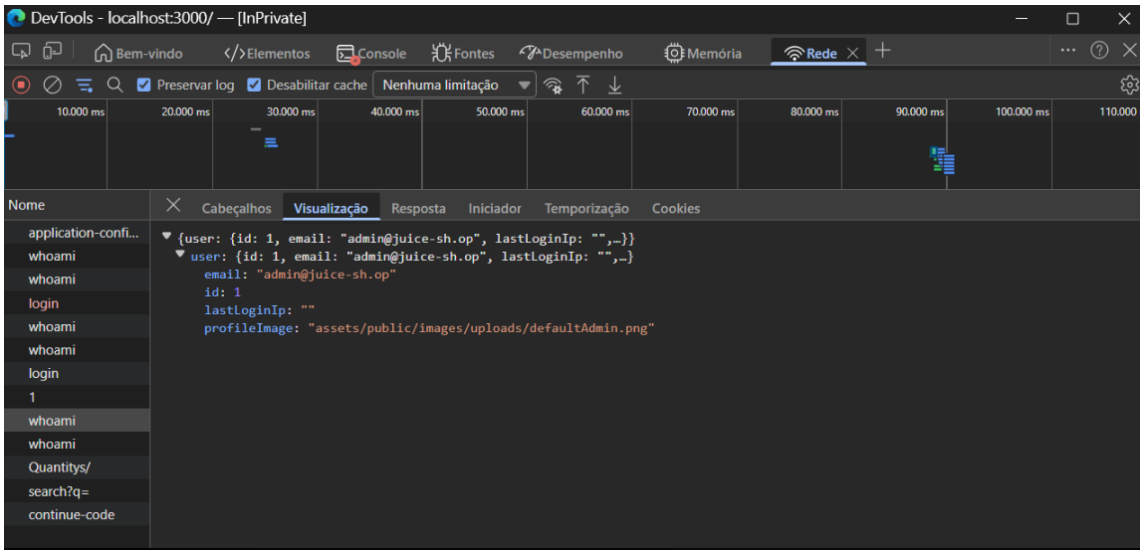
Evidência 3 — Resposta da API confirmando autenticação bem-sucedida

Resposta da API após o envio do payload malicioso, retornando objeto de autenticação contendo token JWT e identificação do usuário administrador (admin@juice-sh.op), confirmando que o backend aceitou a requisição e concedeu acesso sem validação correta das credenciais.



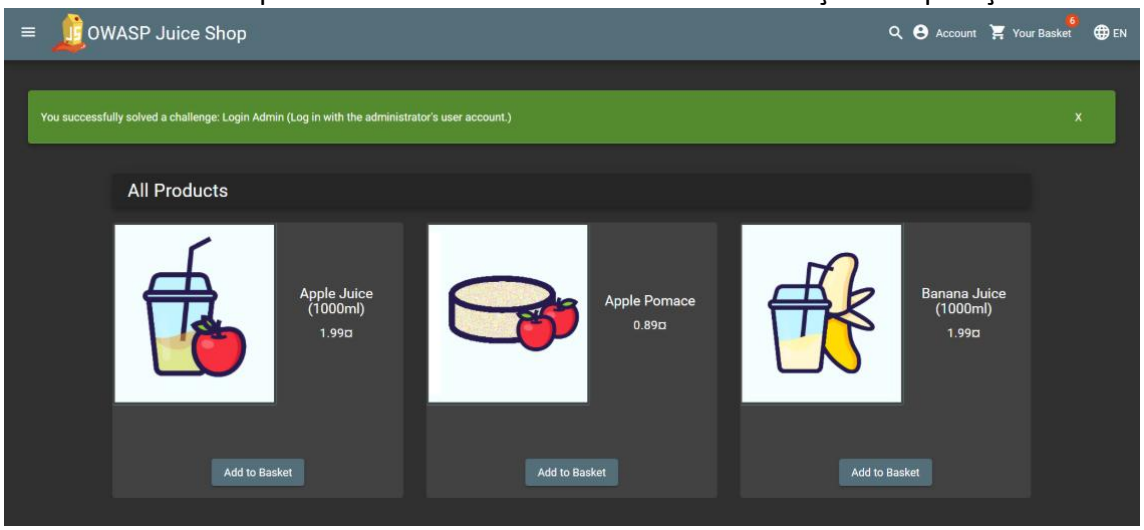
Evidência 4 — Confirmação de sessão autenticada (endpoint whoami)

Requisição ao endpoint whoami após a autenticação, retornando informações do usuário administrador, confirmando que a sessão ativa pertence a uma conta com privilégios elevados.



Evidência 5 — Evidência visual de login bem-sucedido como administrador

Interface da aplicação exibindo o usuário autenticado com permissões administrativas, acompanhada da notificação de sucesso do desafio “Login Admin”, confirmando o comprometimento do mecanismo de autenticação da aplicação.



As evidências demonstram que a aplicação não realiza validação adequada das entradas fornecidas durante o processo de autenticação, permitindo a manipulação da lógica de verificação de credenciais. Essa falha possibilita o bypass completo do mecanismo de login e o acesso não autorizado a contas privilegiadas, representando um risco crítico à segurança da aplicação.

4.3 Impacto

Essa vulnerabilidade permite que um atacante contorne completamente o mecanismo de autenticação da aplicação, obtendo acesso não autorizado a contas com privilégios elevados. O impacto inclui comprometimento da confidencialidade, integridade e disponibilidade do sistema, possibilitando a visualização, modificação ou exclusão de dados sensíveis, além da execução de ações administrativas sem autorização legítima.

4.4 Recomendação

Recomenda-se a implementação de validação rigorosa das entradas fornecidas durante o processo de autenticação, bem como o uso de consultas parametrizadas no backend para evitar manipulação da lógica de verificação de credenciais. Adicionalmente, devem ser adotadas boas práticas de segurança, como tratamento adequado de erros, mecanismos de proteção contra ataques de Injection e testes regulares de segurança nos fluxos de autenticação.

5. Vulnerabilidade 3 – Exposição de Informações Sensíveis por Falha de Controle de Acesso (IDOR / Broken Access Control)

Foi identificada uma falha de controle de acesso inadequado na aplicação, onde endpoints da API retornam informações sensíveis do usuário autenticado sem a devida validação de privilégios ou restrição de acesso.

Após a autenticação, a aplicação permite o acesso a recursos internos, como informações de sessão do usuário (whoami) e dados relacionados a avaliações (reviews), sem aplicar controles adicionais que limitem a exposição dessas informações conforme o nível de permissão do usuário.

5.1 Passo a passo da exploração

1. Realizei a autenticação na aplicação utilizando credenciais obtidas por outra vulnerabilidade.
2. Após o login bem-sucedido, monitorei as requisições realizadas pela aplicação por meio das ferramentas de desenvolvedor do navegador (aba Network / Rede).
3. Observei que a aplicação realiza chamadas ao endpoint ‘whoami’, que retorna informações detalhadas do usuário autenticado, como:

- ID do usuário
- E-mail
- Caminho da imagem de perfil

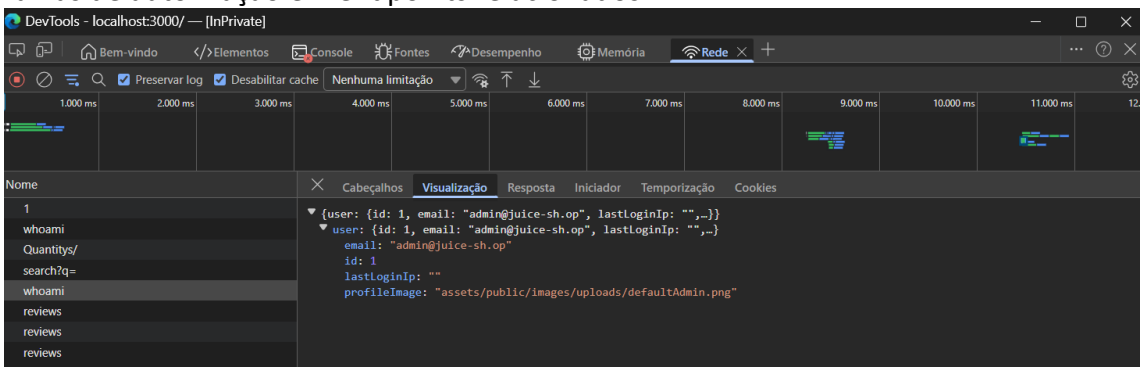
4. Em seguida, acessei diretamente o endpoint ‘reviews’, que retorna dados sem que haja validação adicional de autorização.
5. A API responde com status de sucesso (status: success) e fornece os dados solicitados, demonstrando ausência de verificação de permissões adequadas

5.2 Evidências

Evidência 1 – Enumeração e exposição de dados do usuário autenticado via endpoint /whoami

Nesta evidência, observa-se a resposta do endpoint /whoami, que retorna informações detalhadas do usuário autenticado, incluindo identificador interno (id), e-mail administrativo, IP do último login e caminho da imagem de perfil. A API responde com sucesso (status: success) e fornece dados sensíveis sem camadas adicionais de proteção ou minimização de resposta.

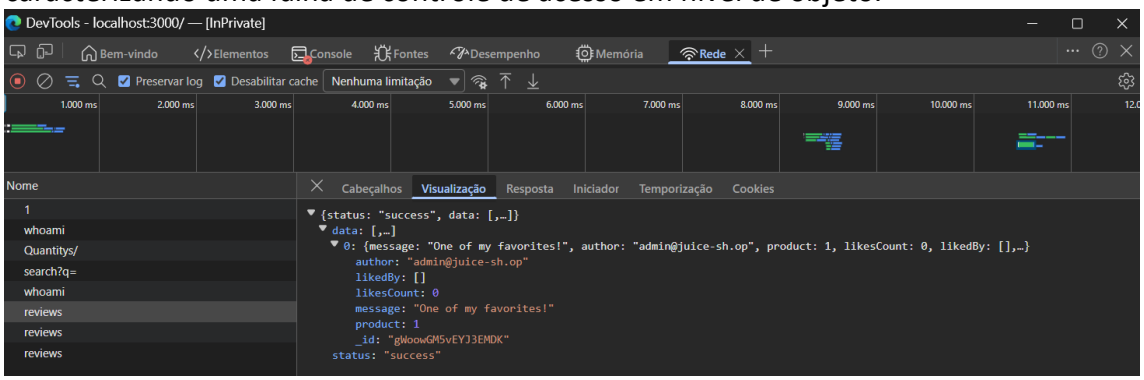
Embora o usuário esteja autenticado, a exposição excessiva de informações internas amplia a superfície de ataque e pode ser explorada em conjunto com outras vulnerabilidades, como enumeração de usuários, engenharia social ou exploração de falhas de autorização em endpoints relacionados.



Evidência 2 – Acesso a avaliações (reviews) associadas a outro usuário

A imagem demonstra que, após a autenticação como usuário administrador, foi possível acessar o endpoint de avaliações (/reviews) e obter dados sensíveis relacionados a comentários do sistema, incluindo mensagem, autor (admin@juice-sh.op), identificador interno do review e referência direta ao produto associado. A resposta da API foi retornada com status success, sem aplicação de filtros adicionais ou validação de escopo de acesso.

Essa evidência confirma que o backend expõe informações internas de usuários e conteúdos sem a devida restrição baseada em contexto ou privilégio específico, caracterizando uma falha de controle de acesso em nível de objeto.



5.3 Impacto

A exploração dessa vulnerabilidade pode resultar em:

- Exposição indevida de informações internas da aplicação.
- Enumeração de dados de usuários e recursos sensíveis.
- Facilitação de ataques encadeados, como: escalonamento de privilégios Engenharia social, mapeamento da estrutura interna da API.
- Quebra dos princípios de confidencialidade e minimização de dados.

Em cenários reais, essa falha pode comprometer a segurança geral da aplicação e dos usuários.

5.4 Recomendação

Recomenda-se a implementação das seguintes medidas corretivas:

- Aplicar validação rigorosa de autorização em todos os endpoints da API, garantindo que cada requisição seja verificada conforme o perfil e privilégio do usuário autenticado.
- Restringir a exposição de informações sensíveis apenas ao estritamente necessário.
- Implementar controles de acesso baseados em função (RBAC – Role-Based Access Control).
- Revisar endpoints que retornam dados de usuário (whoami, reviews, entre outros) para assegurar que não exponham informações além do necessário.
- Realizar testes de segurança focados em Broken Access Control, conforme as diretrizes do OWASP.

6. Resumo dos testes

Este relatório apresenta a análise de segurança realizada na aplicação OWASP Juice Shop, com foco na identificação e exploração controlada de vulnerabilidades presentes em seus mecanismos de autenticação, autorização e exposição de dados via API. A aplicação foi executada em ambiente local, permitindo a observação detalhada do comportamento do frontend e das requisições realizadas ao backend por meio das ferramentas de desenvolvedor do navegador.

Durante os testes, foram identificadas vulnerabilidades relacionadas à falha de validação de entradas**,** controle de acesso inadequado e exposição excessiva de informações sensíveis, possibilitando ações como autenticação indevida e acesso não restrito a dados internos da aplicação. Todas as vulnerabilidades encontradas foram devidamente documentadas, acompanhadas de evidências técnicas, descrição do impacto e recomendações de mitigação, seguindo boas práticas alinhadas ao OWASP Top 10.

O objetivo deste trabalho foi demonstrar, de forma prática, como falhas comuns em aplicações web podem ser exploradas e reforçar a importância da implementação de controles de segurança adequados desde as fases iniciais do desenvolvimento.

7. Conclusão

A análise realizada evidencia que aplicações web que não implementam mecanismos robustos de validação de entradas, controle de acesso e proteção de dados estão suscetíveis a ataques que podem comprometer significativamente a confidencialidade, integridade e disponibilidade das informações. As vulnerabilidades identificadas neste relatório demonstram como falhas aparentemente simples podem ser exploradas para obtenção de acesso indevido e exposição de dados sensíveis.

O uso do OWASP Juice Shop mostrou-se eficaz como ambiente de aprendizado, permitindo compreender, na prática, técnicas de exploração e suas consequências, além de reforçar conceitos fundamentais de segurança da informação. A documentação das vulnerabilidades, acompanhada de evidências técnicas claras, contribui para o entendimento do risco envolvido e para a definição de medidas corretivas adequadas.

Por fim, conclui-se que a adoção de boas práticas de segurança, como validação rigorosa de entradas, aplicação do princípio do menor privilégio e limitação da exposição de dados em APIs, é essencial para reduzir a superfície de ataque e aumentar a maturidade de segurança das aplicações web.