



Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação
Verificação, Validação e Teste de Software (CK0241)

TB03 - Relatório de Análise Estática

GoShop

Italo Sousa de Alencar - 495638
José Portela Soares Neto - 377181
Luís Antônio da Silva Maia - 493458

Histórico de versões

Versão	Data	Autor	Descrição
1.0	20/10/2023	Portela	Criação da versão inicial do relatório de análise estática
1.1	20/10/2023	Italo	Adição da introdução e seus subtópicos
1.2	20/10/2023	Luís	Adição dos problemas na seção 2.1
1.4	23/10/2023	Portela	Criação da Discussão
1.4	24/10/2023	Portela	Criação da Conclusão
1.4	24/10/2023	Portela	Atualização do Glossário
1.5	24/10/2023	Luís	Ajuste na seção 2.1 e revisão geral das sessões
1.6	24/10/2023	Portela	Ajuste na seção 1.2 e revisão geral das sessões
1.7	24/10/2023	Italo	Revisão e formatação das referências bibliográficas e glossário.
2.0	24/10/2023	Italo, Portela e Luis	Revisão e finalização do documento.

Sumário

1. Introdução	4
1.1. Aplicação e código fonte	4
1.2. Descrição da(s) ferramenta(s) de Análise Estática	4
2. Resultados gerais	4
2.1. Lista de problemas analisados	4
3. Discussão	5
4. Conclusão	5
5. Referências	5
Glossário	6

1. Introdução

Este relatório de análise estática tem como objetivo fornecer uma avaliação abrangente e detalhada das características, qualidade e potenciais áreas de melhoria na aplicação *GoShop* [1]. A análise estática é uma metodologia crítica na engenharia de software que se concentra na inspeção de código, estrutura e documentação sem a necessidade de execução do programa.

1.1. Aplicação e código fonte

O sistema *GoShop* [1] trata-se de uma pequena plataforma de e-commerce eletrônico desenvolvida para atender as necessidades de uma loja online tais como gestão de compras e venda de produtos. O sistema conta um módulos de criação de produtos para que o *administrador* possa gerir seus produtos e o módulo de compras para que o *cliente* possa adicionar itens a um carrinho de compras e comprar os produtos dentre outros.

A infraestrutura do *backend* da aplicação foi desenvolvido utilizando *Node.js* [5] e *Express* [3]. No que diz respeito a interface do usuário (*frontend*), foram utilizados *JavaScript* [6], *HTML* [12], *CSS* [13] e *ReactJs* [4]. Quanto ao armazenamento de dados, o sistema faz uso do banco de dados *MySQL* [11]. O sistema está disponível através do link [1]. Além disso, uma cópia do código fonte do sistema foi replicada em um repositório local [2] e foi executada com sucesso.

1.2. Descrição da(s) ferramenta(s) de Análise Estática

O *SonarQube* [10] é uma ferramenta de análise estática de código-fonte de código aberto amplamente utilizada por desenvolvedores de software e equipes de desenvolvimento para melhorar a qualidade do código, identificar problemas de código e garantir a conformidade com padrões de codificação. A ferramenta oferece uma ampla gama de recursos que ajudam a automatizar a revisão de código, detectar vulnerabilidades de segurança, medir a qualidade do código e fornecer informações valiosas para melhorar o processo de desenvolvimento de software.

O *SonarQube* se baseia em regras pré-definidas para analisar o código. Uma regra é uma boa prática e cada linguagem possui um grupo de regras relacionadas.

Toda vez que um código quebra uma regra, uma *Issue* é gerada. As *Issues* estão divididas nas seguintes categorias: *code smells*, *bugs* e vulnerabilidades, Security Hotspot, Coverage, Duplications.

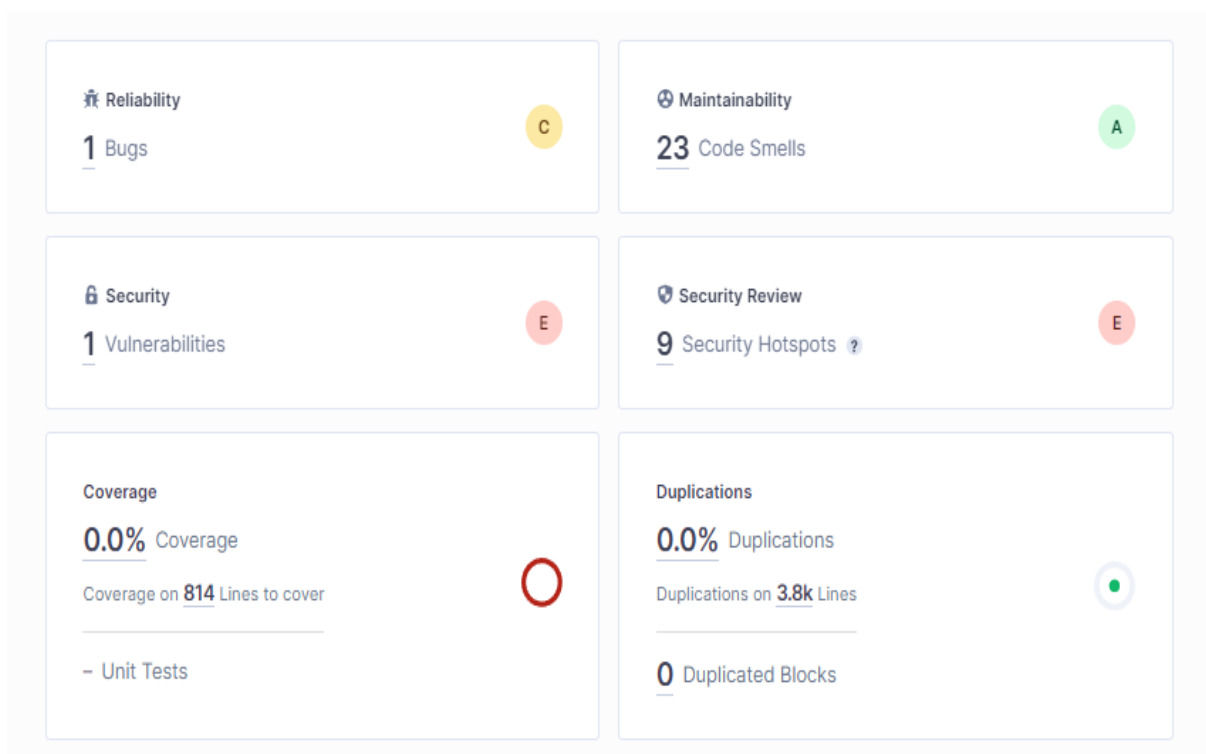
- **Code Smells:** problemas relacionados à manutenção do código. Deixar isso como está significa que, na melhor das hipóteses, os desenvolvedores terão mais dificuldade do que deveriam para fazer alterações no código. Na pior das hipóteses, eles ficarão tão confusos

com o estado do código que introduzirão mais erros à medida que fizerem alterações.

- **Bugs:** itens que representam erros no código que, se ainda não aconteceram em produção, provavelmente acontecerão, e no pior momento possível. Isso precisa ser corrigido.
- **Vulnerabilities (Vulnerabilidades):** são fraquezas ou brechas de segurança na aplicação.
- **Security Hotspot:** é um trecho de código sensível à segurança que é destacado, mas não afeta necessariamente a segurança geral do aplicativo.
- **Coverage:** Os relatórios de cobertura de teste informam qual porcentagem do seu código é coberta pelos casos de teste.
- **Duplications:** Informa que quando existem partes do código contém lógica duplicada.

2. Resultados gerais

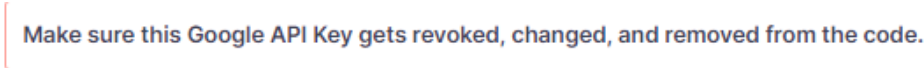
SonarQube[10]:



2.1. Lista de problemas analisados

Identificador	P01
Ferramentas envolvidas	SonarQube
Categoria	Bug
Localização	frontend/src/utils/getStripe.ts
Mensagem do erro	<div>Expected non-Promise value in a boolean conditional.</div>
Trecho do código	<p>As promessas precisam ser resolvidas ou aguardadas para retornar o valor esperado, caso contrário, retorna o objeto da promessa.</p> <p>Usar uma promessa em vez de seu valor resolvido pode ter resultados inesperados, levando a bugs.</p> <ul style="list-style-type: none">• Em condicional, sempre retornará um valor verdadeiro.• Em locais onde o tipo esperado é nulo, retornar uma promessa costuma ser um erro. <pre>goshop > frontend/src/utils/getStripe.ts 1 luisma... import { Stripe, loadStripe } from "@stripe/stripe-js"; 2 3 let stripePromise: Promise<Stripe null>; 4 export const getStripe = () => { 5 if (!stripePromise) {</pre>
Proposta de solução	<p>Inserir um “await” para esperar que a Promise seja resolvida, e somente depois usar o valor do resultado da Promise na condição</p> <pre>(async function foo() { const result = await bar(); // work with result })();</pre>
Comentários	

Identificador	P02
---------------	-----

Ferramentas envolvidas	SonarQube
Categoria	Vulnerability
Localização	frontend/src/app/firebase.ts
Mensagem do erro	
Trecho do código	<p>Uma chave de API irrestrita do Google divulgada em um código-fonte público seria usada por agentes mal-intencionados para consumir APIs do Google em nome do aplicativo. Isto teria um impacto financeiro, pois a organização seria cobrada pelos dados consumidos pelo agente malicioso.</p> <pre> 1 luisma... import { initializeApp } from "firebase/app"; 2 import { GoogleAuthProvider, getAuth } from "firebase/auth"; 3 import { getFirestore } from "firebase/firestore"; 4 5 const firebaseConfig = { 6 apiKey: "AIzaSyDhlcIOSp87k5kpdPgyKcegZKsinJ8OqPQ", </pre>
Proposta de solução	<p>No NodeJs existem as chamadas Variáveis Ambiente. Variáveis de ambiente recebem suporte diretamente do Node e são acessíveis por meio do objeto env (uma propriedade do objeto global process).</p> <p>As variáveis são guardadas em um arquivo .env. Ex.: CHAVE_API_GOOGLE = "chave" E acessadas no código dessa forma: process.env.CHAVE_API_GOOGLE. É subentendido que só acessa a produção quem tem as devidas credenciais, então é preciso ter segurança no acesso do servidor em si.</p> <p>Solução: Cria-se uma variavel ambiente CHAVE_API_GOOGLE = "AlzaSyDhlcIOSp87k5kpdPgyKcegZKsinJ8OqPQ" no arquivo .env, e no trecho do código (mostrado acima) na linha 6, substituir a chave por : process.env.CHAVE_API_GOOGLE.</p>
Comentários	

Identificador	P03
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/src/middleware/multerMiddleware.ts

Mensagem do erro	<div>Make sure the content length limit is safe here.</div>
Trecho do código	<pre> 1 import multer, { diskStorage } from "multer"; 2 3 export const multerUpload = multer({ storage: diskStorage({}) }); </pre>
Proposta de solução	<pre> let diskUpload = multer({ storage: diskStorage, limits: { fileSize: 8000000 // Compliant: 8MB } }); </pre>
Comentários	

Identificador	P04
Ferramentas envolvidas	SonarQube
Categoria	Code Smell
Localização	frontend/src/components/Form/CategorySelectBox.tsx
Mensagem do erro	<div>Prefer using nullish coalescing operator ('??') instead of a logical or (' '), as it is a safer operator.</div>
Trecho do código	<pre> 17 luisma_ return (18 <div className="relative w-full sm:w-32 xl:w-48"> 19 <select 20 id="category" 21 className="cursor-pointer outline-none appear 22 text-secondary text-sm value={props.selectedCategory -1} </pre>
Proposta de solução	<p>Compliant solution</p> <pre> function either(x: number undefined, y: number) { return x ?? y; } </pre>
Comentários	

Identificador	P05
Ferramentas envolvidas	SonarQube
Categoria	Code Smell
Localização	frontend/src/components/Form/ProductQuantitySelectBox.tsx
Mensagem do erro	Do not use Array index in keys
Trecho do código	<pre> 19 luisma_ 20 21 22 23 24 value={props.quantity} onChange={props.handleQuantityChange} > {selectOptions.map((option: number, index: number) => { return (<option key={index} value={option}> </pre>
Proposta de solução	<p>Para corrigir isso, usa-se uma string ou um número que identifique exclusivamente o item da lista. A chave deve ser única entre seus irmãos, e não globalmente.</p> <p>Se os dados vieram de um banco de dados, os IDs do banco de dados já são exclusivos e são a melhor opção. Caso contrário, usa-se um contador ou gerador UUID</p> <pre> function Blog(props) { return ({props.posts.map((post) => <li key={post.id}> {post.title})}); } </pre>
Comentários	

Identificador	P06
Ferramentas envolvidas	SonarQube
Categoria	Code Smell

Localização	frontend/src/features/products/components/EditProductPopu p.tsx
Mensagem do erro	This assertion is unnecessary since it does not change the type of the expression.
Trecho do código	<pre> 18 luisma... 19 20 const onFormSubmit = async (data: ProductFormType, preview: string) => { 21 const updatedProduct = { 22 ...data, 23 image: data.image as Blob, 24 imagePath: preview as string. </pre>
Proposta de solução	<p>Tanto as conversões redundantes quanto as asserções redundantes[14] não nulas devem ser evitadas no código TypeScript[15], pois adicionam ruído desnecessário, desorganizam o código e levam à confusão.</p> <pre> function getNone (x?: string Pessoa) { if (x) { console . log ("Obtendo nome para " + x); // Não compatível: 'x' é conhecido por ser definido aqui if (tipo de x === "string") { retornar (x como string); // Não compatível: 'x' é conhecido por ser uma string aqui } senão { retornar (x como Pessoa). nome ; // Não compatível: 'x' é definido e não uma string, portanto, uma 'Person' aqui } } return "SemNone" ; } </pre> <p>Remove-se todas as conversões redundantes e asserções não nulas com base nas informações de digitação contextual, conforme inferido pelo compilador TypeScript.</p> <pre> function getNone (x?: string Pessoa) { if (x) { console . log ("Obtendo nome para " + x); if (tipo de x === "string") { retornar x; } senão { retornar x. nome ; } } return "SemNone" ; } </pre>
Comentários	

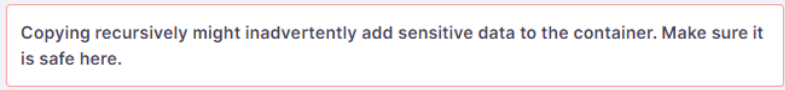
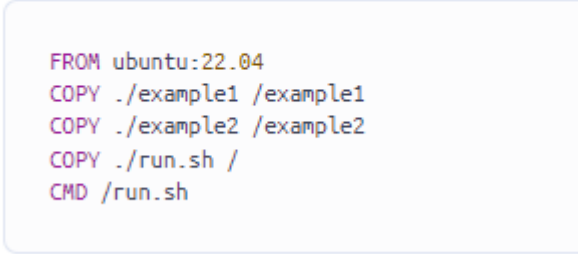
Identificador	P07
Ferramentas envolvidas	SonarQube
Categoria	Code Smell
Localização	backend/src/controllers/webhook.ts

Identificador	P08
Ferramentas envolvidas	SonarQube
Categoria	Code Smell
Localização	frontend/src/components/Elements/Navbar.tsx
Mensagem do erro	<div>'react-router-dom' import is duplicated.</div>

Trecho do código	<pre> 4 import { Link } from "react-router-dom"; 5 import { useAuth } from "../../context/AuthContext"; 6 import { useNavigate } from "react-router-dom"; </pre>
Proposta de solução	<pre> import { B1, B2 } from 'b'; </pre>
Comentários	

Identificador	P09
Ferramentas envolvidas	SonarQube
Categoria	Code Smell
Localização	frontend/src/context/AuthContext.tsx
Mensagem do erro	<p>The 'value' object passed as the value prop to the Context provider changes every render. To fix this consider wrapping it in a useMemo hook.</p>
Trecho do código	<pre> 85 const value = { 86 currentUser, 87 isAdmin, 88 token, 89 signIn, 90 signInWithToken, 91 signInWithGoogle, 92 signUp, 93 signOut 94 }; </pre>
Proposta de solução	<pre> function Component() { const obj = useMemo(() => ({foo: 'bar'}), []); // value is cached by useMemo return (<SomeContext.Provider value={obj}> { /* Compliant */ } <SomeComponent /> </SomeContext.Provider>); } </pre>
Comentários	É uma boa prática de programação em React usar o Hook

	useMemo[16] para guardar o valor em cache para que o React não mude os valores do objeto a cada renderização
--	--

Identificador	P10
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/Dockerfile
Mensagem do erro	
Trecho do código	<pre> 27 # Generate Prisma Client 28 COPY --link prisma . 29 RUN npx prisma generate 30 31 # Copy application code 32 COPY --link . . </pre>
Proposta de solução	<p>É recomendado evitar copiar todo o diretório de contexto para o sistema de arquivos de imagem.</p> <p>Compliant Solution</p>  <pre> FROM ubuntu:22.04 COPY ./example1 /example1 COPY ./example2 /example2 COPY ./run.sh / CMD /run.sh </pre>
Comentários	

Identificador	P11
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/Dockerfile

Mensagem do erro	<div>This image might run with root as the default user. Make sure it is safe here.</div>
Trecho do código	<pre> 37 # Remove development dependencies 38 RUN npm prune --omit=dev 39 40 41 # Final stage for app image 42 FROM base </pre>
Proposta de solução	<p>Compliant Solution</p> <p>For Linux-based images:</p> <pre> FROM alpine RUN addgroup -S nonroot \ && adduser -S nonroot -G nonroot USER nonroot ENTRYPOINT ["id"] </pre> <p>For Windows-based images, you can use <code>ContainerUser</code> or create a new user:</p> <pre> FROM mcr.microsoft.com/windows/servercore:ltsc2019 RUN net user /add nonroot USER nonroot </pre>
Comentários	

Identificador	P12
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/src/seed.ts
Mensagem do erro	<div>Make sure that using this pseudorandom number generator is safe here.</div>
Trecho do código	<pre> 28 29 const category = categories[Math.floor(Math.random() * 10)]; </pre>

Proposta de solução	<p>É recomendado que se use um gerador de números pseudo aleatórios criptograficamente forte</p> <p>Compliant Solution</p> <pre>// === Client side === const crypto = window.crypto window.msCrypto; var array = new Uint32Array(1); crypto.getRandomValues(array); // Compliant for security-sensitive use cases // === Server side === const crypto = require('crypto'); const buf = crypto.randomBytes(1); // Compliant for security-sensitive use cases</pre>
Comentários	

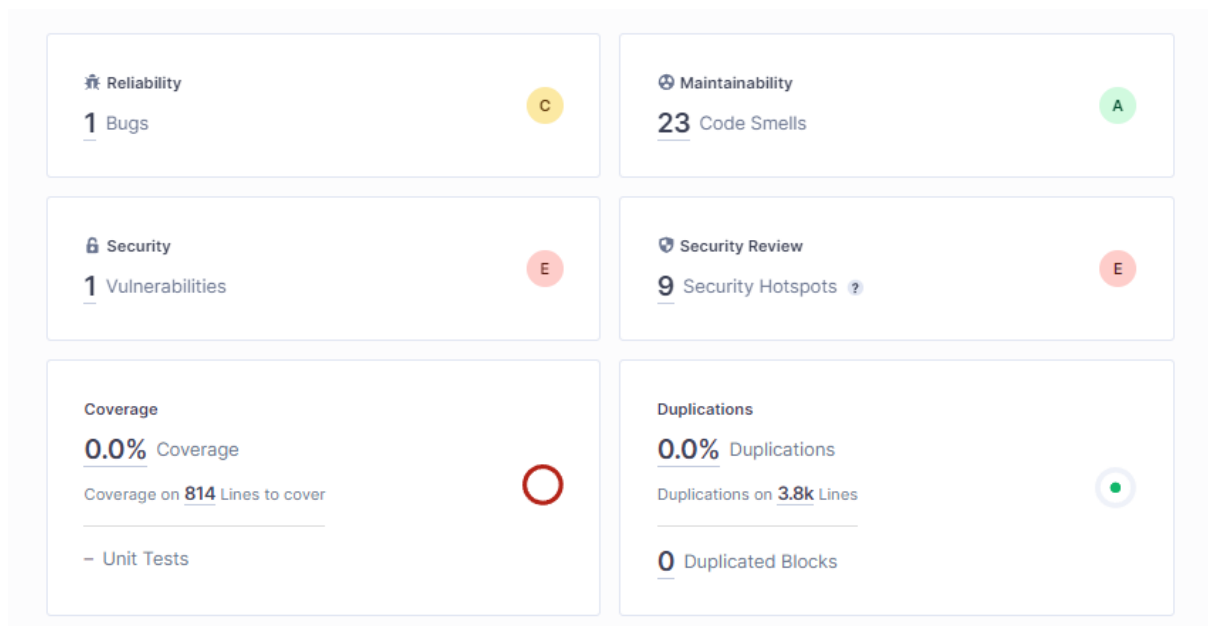
Identificador	P13
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/Dockerfile
Mensagem do erro	<div>Make sure automatically installing recommended packages is safe here.</div>
Trecho do código	<pre>19 # Install packages needed to build node modules 20 RUN apt-get update -qq && \ 21 apt-get install -y build-essential openssl pkg-config python-is-python3</pre>
Proposta de solução	<p>É recomendável que evite instalar dependências de pacotes que não sejam estritamente necessários.</p> <p>Compliant Solution</p> <pre>FROM debian:latest RUN apt install -y --no-install-recommends build-essential RUN apt-get install -y --no-install-recommends build-essential RUN aptitude install -y --without-recommends build-essential</pre>
Comentários	

Identificador	P14
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/Dockerfile
Mensagem do erro	<div>Omitting --ignore-scripts can lead to the execution of shell scripts. Make sure it is safe here.</div>
Trecho do código	<pre> 23 # Install node modules 24 COPY --link package-lock.json package.json ./ 25 RUN npm ci --include=dev </pre>
Proposta de solução	<p>É recomendável que a execução de scripts de terceiros deve ser desabilitada se não for estritamente necessária para que as dependências funcionem corretamente. Isso reduzirá a superfície de ataque e bloqueará um vetor de ataque bem conhecido à cadeia de suprimentos. [17]</p> <div> <h3>Compliant Solution</h3> <pre> FROM node:latest RUN npm install --ignore-scripts </pre> <pre> FROM node:latest RUN yarn install --ignore-scripts </pre> </div>
Comentários	

Identificador	P15
Ferramentas envolvidas	SonarQube
Categoria	Security Hotspots
Localização	backend/src/index.ts

Mensagem do erro	<div> <div></div> <div>Make sure disclosing the fingerprinting of this web technology is safe here.</div> </div>
Trecho do código	<pre> 14 import { v2 as cloudinary } from "cloudinary"; 15 16 const app = express(); </pre>
Proposta de solução	<p>É recomendado não divulgar tecnologias utilizadas em um site, com x-powered-by no cabeçalho HTTP, por exemplo.</p> <p>Além disso, é melhor desativar completamente esse cabeçalho HTTP em vez de definir um valor aleatório.</p> <pre> let express = require('express'); let app1 = express(); // Compliant app1.disable("x-powered-by"); let helmet = require("helmet"); let app2 = express(); // Compliant app2.use(helmet.hidePoweredBy()); </pre>
Comentários	

3. Discussão



Os dados coletados inicialmente através do SonarQube revelam as questões iniciais que a aplicação apresentava, conforme evidenciado a seguir:

Nesse contexto, é visível a existência de 1 problema de confiabilidade, 1 fragilidade de segurança, 23 aspectos de código problemáticos no âmbito da manutenção do código, e 9 pontos críticos de segurança sob a perspectiva de revisão de segurança.

Dado o número limitado de dados, o SonarQube não pôde criar gráficos mais detalhados:



4. Conclusão

Foi oferecida uma introdução à aplicação examinada, as ferramentas empregadas, os equívocos identificados e os resultados alcançados por meio da análise estática realizada com o SonarQube. Pudemos concluir que o emprego de ferramentas de análise estática desempenha um papel significativo na aprimoração do código, tanto por meio da detecção de indícios de problemas que poderiam evoluir para erros, quanto pela identificação de falhas e vulnerabilidades que poderiam causar sérios prejuízos durante a utilização.

5. Referências

1. E-commerce GoShop, 2023. Disponível em: <https://ecommerce-goshop.onrender.com/>. Acesso em: 20/10/2023.
2. Github. GoShop Testing, 2023. Disponível em: <https://github.com/luismaia-git/goshop-testing>. Acesso em: 20/10/2023.
3. ExpressJS. Express: Node.js web application framework, 2017. Disponível em: <https://expressjs.com/>. Acesso em: 20/10/2023.
4. React.dev. Built-in React Hooks, 2023. Disponível em: <https://react.dev/reference/react>. Acesso em: 20/10/2023.
5. NodeJS. About Node.js. Disponível em <https://nodejs.org/en>. Acesso em: 20/10/2023.
6. JavaScript.com. Learn JavaScript Online, 2023. Disponível em: <https://www.javascript.com/>. Acesso em: 20/10/2023.
8. Jest. Delightful JavaScript Testing, 2023. Disponível em: <https://jestjs.io/pt-BR/>. Acesso em: 20/10/2023.
10. SonarQube. Code Quality, Security & Static Analysis Tool with SonarQube, 2023. Disponível em: <https://www.sonarsource.com/products/sonarqube/>. Acesso em: 20/10/2023.
11. MySQL, 2023. Disponível em: <https://www.mysql.com/>. Acesso em: 20/10/2023.
14. Asserções. Type Assertions, 2023. Disponível em: <https://www.typescriptlang.org/docs/handbook/2/everyday-types.html#type-assertions>. Acesso em: 24/10/2023.
15. Typescript. The starting point for learning TypeScript, 2023. Disponível em: <https://www.typescriptlang.org/docs/>. Acesso em 24/10/2023.
16. useMemo. useMemo - React. Disponível em: <https://react.dev/reference/react/useMemo>. Acesso em 24/10/2023.
17. ESLint blog - Postmortem for Malicious Packages Publicado em 12 de Julho de 2018. Disponível em: <https://eslint.org/blog/2018/07/postmortem-for-malicious-package-publishes/>. Acesso em: 24/10/2023

Glossário

Termo	Definição
Code Smell	Qualquer característica no código-fonte de um programa que possivelmente indica um problema mais profundo
Security Hotspot	Pela definição do <i>SonarQube</i> [10], é um trecho de código que possui uma sensibilidade, a nível de segurança da aplicação.
Hook	Em React, "hooks" são funções especiais que permitem que você "ligue" o estado e o ciclo de vida de componentes funcionais, que são chamados de "functional components".

HTTP	HTTP é um protocolo de transferência que possibilita que as pessoas que inserem a URL do seu site na Web possam ver os conteúdos e dados que nele existem. A sigla vem do inglês Hypertext Transfer Protocol.
------	---