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

TB03 - Relatório de Análise Estática

Food Delivery Website - Tomato

Gabriel Moreira de Andrade - 536708
Álvaro Siqueira Galvão - 537649
Fábio Agostinho Filho - 538521

Histórico de versões

Versão	Data	Autor	Descrição
1.0	10/01/2025	Fábio	Criação da versão inicial do relatório de análise estática
1.1	11/01/2025	Gabriel, Álvaro e Fábio	Realização de Testes de análise estática
1.2	12/01/2025	Gabriel	Realização das seções introdutórias
1.3	13/01/2025	Gabriel, Álvaro e Fábio	Realização da discussão e dos resultados gerais dos testes
1.4	14/01/2025	Fábio	Seções Finais e envio

Sumário

1. Introdução

Esse documento apresenta os testes de análise estática e discussões sobre os resultados do projeto apresentado no TB1 e TB2.

1.1. Aplicação e código fonte

O Tomato é uma aplicação web FullStack open source, construída com a stack MERN (MongoDB, Express.js, React.js e Node.js) feita para gerenciar pedidos de alimentos de forma simples e intuitiva, focando no aprendizado e acessibilidade para desenvolvedores (ATHRVANAIK, 2024).

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

As categorias utilizadas na lista de problemas analisados serão os que a própria ferramenta ESLint possui. Esse foi o log dos erros:

```
/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/Pages/Add/Add.jsx
  1:17  error  'useEffect' is defined but never used  no-unused-vars
  8:13  error  'url' is missing in props validation   react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/Pages/List/List.jsx
  6:14  error  'url' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/Pages/Orders/Orders.jsx
  9:18  error  'url' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/main.jsx
  7:3  error  'React' must be in scope when using JSX  react/react-in-jsx-scope
  8:5  error  'React' must be in scope when using JSX  react/react-in-jsx-scope

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/config/db.js
  5:32  error  'process' is not defined  no-undef

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/controllers/orderController.js
  5:27  error  'process' is not defined  no-undef

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/controllers/userController.js
  31:27  error  'process' is not defined  no-undef

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/middleware/auth.js
```

```
  10:47  error  'process' is not defined  no-undef

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/server.js
  13:13  error  'process' is not defined  no-undef

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/ExploreMenu/ExploreMenu.jsx
  5:23  error  'category' is missing in props validation           react/prop-types
  5:32  error  'setCategory' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/FoodDisplay/FoodDisplay.jsx
  6:23  error  'category' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/FoodItem/FoodItem.jsx
  1:29  error  'useState' is defined but never used                no-unused-vars
  6:20  error  'id' is missing in props validation          react/prop-types
  6:23  error  'name' is missing in props validation                react/prop-types
  6:28  error  'price' is missing in props validation      react/prop-types
  6:34  error  'description' is missing in props validation  react/prop-types
  6:46  error  'image' is missing in props validation               react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/LoginPopUp/LoginPopup.jsx
  1:29  error  'useEffect' is defined but never used               no-unused-vars
  7:23  error  'setShowLogin' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/Navbar/Navbar.jsx
  7:19  error  'setShowLogin' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/context/StoreContext.jsx
  84:9   error  'React' must be in scope when using JSX
react/react-in-jsx-scope
  85:20  error  'children' is missing in props validation  react/prop-types

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/pages/Cart/Cart.jsx
  24:31  error  'index' is defined but never used  no-unused-vars

/home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/pages/Verify/Verify.jsx
  10:25  error  'setSearchParams' is assigned a value but never used
no-unused-vars

✖ 27 problems (27 errors, 0 warnings)
```

2. Resultados gerais

[Fazer uma introdução dos problemas que serão apresentados: quantos e como estão distribuídos nas categorias/regras que a ferramenta cobre, dentre outras informações.]

2.1. Lista de problemas analisados

Erro no.1

Identificador	E1
Ferramentas envolvidas	ESLint
Categoria	Style
Localização	<Admin/src/Pages/Add/Add.jsx, linha 1 >
Mensagem do erro	<1:17 error 'useEffect' is defined but never used no-unused-vars >
Trecho do código	<useEffect(() => {}, []);>
Proposta de solução	<Remover useEffect>
Proposta de solução (ferramenta IA)	<Remover useEffect ou adicionar lógica adequada baseada nas dependências.>
Comentários	O problema identificado deveria ser resolvido apenas removendo essa parte, já que não havia uso naquele momento.

Erro no.2

Identificador	E2
Ferramentas envolvidas	ESLint
Categoria	Prop-Types Missing
Localização	<Frontend/src/components/FoodDisplay/FoodDisplay.jsx>
Mensagem do erro	<’category' is missing in props validation>
Trecho do código	<FoodDisplay({ items })>
Proposta de solução	Adicionar o tipo de variável para evitar erros de tipagem: <FoodDisplay.propTypes = { category: PropTypes.string.isRequired, }; >
Proposta de solução (ferramenta IA)	<Para resolver o erro de **Prop-Types Missing** no React, onde falta a validação de category em props, siga os passos abaixo: **Passos para resolver:** 1. **Verifique a definição de Prop-Types**: Certifique-se de que você está usando PropTypes para validar as propriedades no seu componente. 2. **Adicionar Prop-Types para category**: A linha 6 no seu FoodDisplay.jsx provavelmente deve parecer com isso: import PropTypes from 'prop-types'; 3. 4. const FoodDisplay = ({ food, category }) => { 5. return (6. 7. {/* Código para exibir o alimento */} 8. 9.); 10. }; 11. 12. FoodDisplay.propTypes = { 13. food: PropTypes.object.isRequired, 14. category: PropTypes.string.isRequired // Adicionando PropTypes para `category` 15. }; 16. 17. export default FoodDisplay; 18.

	1. **Recompile e verifique**: Após adicionar a validação de Prop-Types, compile sua aplicação e verifique se o erro foi corrigido. **Explicação:** ● PropTypes.object.isRequired valida que a propriedade passada é um objeto e não pode ser undefined. ● PropTypes.string.isRequired valida que a propriedade passada é uma string e não pode ser undefined. >
Comentários	Sugerimos apenas adicionar o tipo “string” como requerido para facilitar a resolução. Além disso, a solução da IA foi imprecisa, já que havia somente uma props a ser recebida “Category” e não 2 como colocado pela IA.

Erro no.3

Identificador	E3
Ferramentas envolvidas	ESLint
Categoria	JSX Scope
Localização	<Frontend/src/context/StoreContext.jsx>
Mensagem do erro	<'React' must be in scope when using JSX>
Trecho do código	<return <>{children}</>; >
Proposta de solução	Apenas importar: <import React from 'react'; >
Proposta de solução (ferramenta IA)	Mesma solução
Comentários	Nesse caso, o erro era relativo a falta da importação do “React” que é essencial para o funcionamento do código

Erro no. 4

Identificador	E4
Ferramentas envolvidas	ESLint
Categoria	PropTypes Missing
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/Pages/Orders/Orders.jsx>, linha 9
Mensagem do erro	<'url' is missing in props validation>
Trecho do código	<function Orders({url}) { [...] }>
Proposta de solução	Instalação e adição no código da biblioteca PropTypes e criação de um objeto dela no código onde se definirá propriamente a variável 'url'. import PropTypes from 'prop-types'; const MyComponent = ({ url }) => { return Link; }; MyComponent.propTypes = { url: PropTypes.string.isRequired, // Marca "url" como obrigatória e do tipo string }; export default MyComponent; >
Proposta de solução (ferramenta IA)	O problema descrito, "url" is missing in props validation, ocorre geralmente quando você está usando **PropTypes** para validar as propriedades de um componente React, mas a propriedade url não foi declarada ou configurada corretamente. **Solução** Você deve adicionar a propriedade url à validação de propriedades do componente. Aqui está como resolver o problema: **1. Usando PropTypes** Certifique-se de que você está importando o PropTypes e declarando que a propriedade url é obrigatória ou opcional, conforme necessário.

	[...] Se a propriedade não for obrigatória, remova o .isRequired:
Comentários	Um erro simples que deixa o código propenso a comportamentos inesperados caso a variável não seja propriamente inicializada, por isso a importância de declarar a variável da maneira correta, usando o PropTypes.

Erro no. 5

Identificador	E5
Ferramentas envolvidas	ESLint
Categoria	“react/react-in-jsx-scope”
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/main.jsx>, linha 7
Mensagem do erro	<‘React’ must be in scope when using JSX>
Trecho do código	< createRoot(document.getElementById('root')).render(**<BrowserRouter>** <App /> </BrowserRouter>) > Obs.: Em negrito o erro apontado.
Proposta de solução	Adicionar explicitamente a importação ‘React’ ao início do código. < import React from 'react' >
Proposta de solução (ferramenta IA)	O erro "React must be in scope when using JSX" ocorre porque, em versões anteriores ao React 17, o compilador JavaScript espera que o objeto React esteja disponível no escopo para interpretar e processar JSX. # Soluções ## 1. Importar o React Explicitamente (Para React 16 ou Inferiores) Se você estiver usando uma versão do React **anterior à 17**, importe explicitamente o React no seu arquivo para garantir que ele esteja no

	escopo. [...] **2. Atualizar para React 17 ou Superior** A partir do React 17, o import do React não é mais necessário para usar JSX, pois o novo compilador do JSX (introduzido com o React 17) não depende mais do objeto React estar no escopo. Certifique-se de que o projeto utiliza pelo menos a versão 17 do React. [...]
Comentários	Apenas uma falta de declaração de biblioteca que atualmente é desnecessária, mas que ao tentar rodar em compiladores antigos não seria possível.

Erro no. 6

Identificador	E6
Ferramentas envolvidas	ESLint
Categoria	Object out of scope
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Admin/src/main.jsx>, linha 8
Mensagem do erro	<‘React’ must be in scope when using JSX>
Trecho do código	< createRoot(document.getElementById('root')).render(<BrowserRouter> **<App />** </BrowserRouter>) > Obs.: Em negrito o erro apontado.
Proposta de solução	Adicionar explicitamente a importação ‘React’ ao início do código. < import React from 'react' >
Proposta de solução (ferramenta IA)	O erro "React must be in scope when using JSX" ocorre porque, em versões anteriores ao React 17, o compilador JavaScript espera que o

	objeto React esteja disponível no escopo para interpretar e processar JSX. **Soluções** **1. Importar o React Explicitamente (Para React 16 ou Inferiores)** Se você estiver usando uma versão do React **anterior à 17**, importe explicitamente o React no seu arquivo para garantir que ele esteja no escopo. [...] **2. Atualizar para React 17 ou Superior** A partir do React 17, o import do React não é mais necessário para usar JSX, pois o novo compilador do JSX (introduzido com o React 17) não depende mais do objeto React estar no escopo. Certifique-se de que o projeto utiliza pelo menos a versão 17 do React. [...]
Comentários	Outra ocasião em que ocorreu o erro de identificador XXXXX, o que mostra uma certa limitação da ferramenta, visto que o mesmo erro já havia sido apontado na linha anterior, o que poderia ser interpretado como um mesmo erro.

Erro no. 7

Identificador	E7
Ferramentas envolvidas	ESLint
Categoria	Missing statement
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/config/db.js>, linha 5
Mensagem do erro	<‘process’ is not defined>
Trecho do código	< await mongoose.connect(process.env.MONGO_URI); >
Proposta de solução	Provavelmente um erro da ferramenta, já que o termo ‘process’ deveria ser reconhecido por ser uma variável própria do node. Apenas seria suficiente setar “ “node” : True “ no arquivo de configurações para deixar explícito que se está usando variáveis do node.

Proposta de solução (ferramenta IA)	O erro 'process' is not defined geralmente ocorre em um código JavaScript/Node.js quando você tenta acessar a variável global process, mas ela não está disponível no ambiente em que o código está sendo executado. Isso pode acontecer, por exemplo, ao usar process.env em um ambiente de navegador ou em configurações específicas de ferramentas. ## Causas Comuns e Soluções ## 1. Acessando process em um Ambiente do Navegador O objeto process está disponível apenas no Node.js, não no navegador. Se você estiver tentando usar variáveis de ambiente (process.env) em um ambiente de navegador, será necessário configurar corretamente essas variáveis. **Solução:** Use uma ferramenta como **Webpack** ou **Vite** para definir variáveis de ambiente para o navegador. [...] ## 2. Esquecendo de Configurar process.env em um Projeto Node.js Se o código estiver sendo executado em Node.js, mas process.env não está definido, talvez você precise configurar variáveis de ambiente. **Solução:** Use um pacote como **dotenv** para carregar variáveis de ambiente. [...] ## 3. Erro no ESLint (Variável Global Não Reconhecida) Se o ESLint está reclamando que process não está definido, você pode configurá-lo para reconhecer o ambiente. **Solução:** No arquivo de configuração do ESLint (.eslintrc.json, .eslintrc.js ou similar), inclua o ambiente do Node.js: [...]
Comentários	Apenas um erro na configuração da ferramenta, que poderia vir a ser

	um problema se a configuração da aplicação não estivesse devidamente configurada para usar as variáveis do node, o que não é o caso, e se fosse, acarretaria muitos outros problemas.

Erro no. 8

Identificador	E8
Ferramentas envolvidas	ESLint
Categoria	PropTypes Missing
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/ExploreMenu/ExploreMenu.jsx>, linha 5
Mensagem do erro	<'category' is missing in props validation >
Trecho do código	< const ExploreMenu = ({category,setCategory}) => { [...] } >
Proposta de solução	Instalação e adição no código da biblioteca PropTypes e criação de um objeto dela no código onde se definirá propriamente a variável ‘category’.
Proposta de solução (ferramenta IA)	O problema descrito, "category" is missing in props validation, ocorre geralmente quando você está usando **PropTypes** para validar as propriedades de um componente React, mas a propriedade category não foi declarada ou configurada corretamente. # Solução Você deve adicionar a propriedade category à validação de propriedades do componente. Aqui está como resolver o problema: ## 1. Usando PropTypes Certifique-se de que você está importando o PropTypes e declarando que a propriedade url é obrigatória ou opcional, conforme necessário. [...] Se a propriedade não for obrigatória, remova o .isRequired:
Comentários	Mais um caso de falta de declaração do PropTypes no código.

Erro no. 9

Identificador	E9
Ferramentas envolvidas	ESLint
Categoria	Missing statement
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Backend/controllers/orderController.js>, linha 5
Mensagem do erro	<‘process’ is not defined>
Trecho do código	< const stripe = new Stripe(process.env.STRIPE_SECRET_KEY); >
Proposta de solução	Provavelmente um erro da ferramenta, já que o termo ‘process’ deveria ser reconhecido por ser uma variável própria do node. Apenas seria suficiente setar “ “node” : True “ no arquivo de configurações para deixar explícito que se está usando variáveis do node.
Proposta de solução (ferramenta IA)	O erro 'process' is not defined geralmente ocorre em um código JavaScript/Node.js quando você tenta acessar a variável global process, mas ela não está disponível no ambiente em que o código está sendo executado. Isso pode acontecer, por exemplo, ao usar process.env em um ambiente de navegador ou em configurações específicas de ferramentas. **Causas Comuns e Soluções** **1. Acessando process em um Ambiente do Navegador** O objeto process está disponível apenas no Node.js, não no navegador. Se você estiver tentando usar variáveis de ambiente (process.env) em um ambiente de navegador, será necessário configurar corretamente essas variáveis. **Solução:** Use uma ferramenta como **Webpack** ou **Vite** para definir variáveis de ambiente para o navegador. [...] **2. Esquecendo de Configurar process.env em um Projeto Node.js** Se o código estiver sendo executado em Node.js, mas process.env não está definido, talvez você precise configurar variáveis de ambiente. **Solução:**

	Use um pacote como **dotenv** para carregar variáveis de ambiente. [...] **3. Erro no ESLint (Variável Global Não Reconhecida)** Se o ESLint está reclamando que process não está definido, você pode configurá-lo para reconhecer o ambiente. **Solução:** No arquivo de configuração do ESLint (.eslintrc.json, .eslintrc.js ou similar), inclua o ambiente do Node.js: [...]
Comentários	Mais outro caso de variável do node não identificada pela ferramenta como já própria da aplicação, desta vez em outro arquivo.

Erro no. 10

Identificador	E10
Ferramentas envolvidas	ESLint
Categoria	no unused vars
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/FoodItem/FoodItem.jsx>, linha 1
Mensagem do erro	<'useState' is defined but never used>
Trecho do código	<import React, { useContext, useState } from 'react'>
Proposta de solução	Apenas a retirada da importação do useState já solucionaria o problema, tendo em vista que ele não é usado em nenhum momento.
Proposta de solução (ferramenta IA)	Vamos detalhar algumas **causas comuns** para o erro 'useState' is defined but never used e suas respectivas **soluções**. **Causas Comuns e Soluções**

1. Importação Desnecessária do useState

- **Causa:** Você importou o hook useState, mas acabou não utilizando ele no seu componente.
- **Solução:** Se o useState não for necessário no componente, basta removê-lo.

Exemplo de código:

```
import React, { useState } from 'react'; // Remover se não for necessário

function FoodItem() {

  // Código sem o uso de useState

  return <div>Item de comida</div>;

}

export default FoodItem;
```

2. Esquecimento de Usar o useState Após a Importação

- **Causa:** Você importou o useState com a intenção de usá-lo, mas acabou não utilizando. Pode acontecer quando você planeja adicionar um estado, mas esquece de implementá-lo.
- **Solução:** Implemente o useState para criar um estado necessário no componente, caso tenha sido planejado. Se o estado for algo simples, como um contador ou uma variável booleana, o useState é a escolha correta.

Exemplo de código:

```
import React, { useState } from 'react';

function FoodItem() {

  // Criando o estado 'quantity' com o useState

  const [quantity, setQuantity] = useState(1);



  const increment = () => {

    setQuantity(prevQuantity => prevQuantity + 1);
```

```
  };


  return (
    <div>
      <h3>Quantidade: {quantity}</h3>
      <button onClick={increment}>Adicionar</button>
    </div>
  );
}

export default FoodItem;
```

3. Uso de useState em um Contexto que Não Exige Estado

- **Causa:** Às vezes, o useState é importado sem uma necessidade real para controlar o estado do componente, como quando o componente não precisa gerenciar nenhuma variável de estado.
- **Solução:** Se não houver necessidade de um estado dinâmico (como valores que mudam durante a interação do usuário), você pode refatorar o componente para não usar useState ou substituir por variáveis simples.

Exemplo de código refatorado sem useState:

```
import React from 'react';


function FoodItem() {
  return <div>Item de comida</div>;
}

export default FoodItem;
```

4. Uso de useState em uma Função ou Componente Condicional

	● **Causa:** Você pode ter planejado usar useState, mas talvez tenha colocado sua lógica de estado dentro de um bloco condicional ou em um lugar que não é executado sempre. Como resultado, o hook não é realmente utilizado no fluxo do componente. ● **Solução:** Assegure-se de que o useState seja utilizado de forma consistente em todo o componente e que seja parte do fluxo de execução. **Exemplo de código:** import React, { useState } from 'react'; function FoodItem({ isAvailable }) { const [quantity, setQuantity] = useState(1); if (!isAvailable) { return Item indisponível ; } return (Quantidade: {quantity} <button onClick={() => setQuantity(quantity + 1)}>Adicionar</button>); } export default FoodItem;

	Resumo 1. **Remover a importação de useState** se ele não for realmente necessário no componente. 2. **Utilizar o useState** corretamente, criando e manipulando um estado se houver a necessidade de variáveis dinâmicas. 3. **Verificar se o useState está sendo usado corretamente em toda a execução do componente**, especialmente se a lógica estiver dentro de blocos condicionais ou execuções não padrão. Essas são as causas comuns que podem gerar o erro 'useState' is defined but never used e as respectivas soluções para corrigir o problema de forma eficaz.
Comentários	Este seria um ótimo exemplo para mostrar a importância da revisão de código e da boa comunicação entre os desenvolvedores

Erro no. 11

Identificador	E11
Ferramentas envolvidas	ESLint
Categoria	No unused vars
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/LoginPopUp/LoginPopup.jsx>, linha 1
Mensagem do erro	<'useEffect' is defined but never used>
Trecho do código	<import React, { useContext, useEffect, useState } from "react";>
Proposta de solução	Retirar o useEffect da importação já funcionaria, pois ele não é utilizado em nenhum momento do código
Proposta de solução (ferramenta IA)	Esse erro, "'useEffect' is defined but never used", ocorre quando você importa o hook useEffect em um arquivo, mas não o utiliza dentro do código do componente. Abaixo, apresento algumas **causas comuns** para esse erro e suas respectivas **soluções**. **Causas Comuns e Soluções** **1. Importação Desnecessária do useEffect**

- **Causa:** Você importou o useEffect, mas não está utilizando ele no componente. Isso é comum quando você planeja adicionar efeitos colaterais (side effects) no componente, mas depois esquece ou decide não implementar.
- **Solução:** Se não for necessário utilizar useEffect, remova a importação dele.

Exemplo de código:

```
import React from 'react'; // Remover importação de useEffect se não for usado

function LoginPopup() {

  return <div>Popup de Login</div>;

}

export default LoginPopup;
```

2. Esquecimento de Usar useEffect Após Importação

- **Causa:** Você importou o useEffect com a intenção de usá-lo, mas depois esqueceu de implementar a lógica do efeito. Pode ser que você tenha planejado um efeito colateral (como fazer uma requisição de API, alterar o título da página, etc.), mas acabou não implementando.
- **Solução:** Implemente o useEffect se você realmente precisa de um efeito colateral no componente, como atualizar o título da página, fazer uma requisição assíncrona, ou adicionar listeners de eventos.

Exemplo de código com useEffect:

```
import React, { useEffect } from 'react';



function LoginPopup() {

  // Exemplo de uso do useEffect para mudar o título da página quando o popup for exibido

  useEffect(() => {

    document.title = 'Login - Tomato Food Ordering'; // Muda o título da página
```

```
    // Cleanup (opcional) para quando o componente for desmontado

    return () => {

      document.title = 'Tomato Food Ordering'; // Restaura o título ao
desmontar o popup

    };

  }, []); // O array vazio [] garante que o efeito só será executado uma
vez



  return <div>Popup de Login</div>;

}

export default LoginPopup;
```

3. Uso de useEffect de Forma Condicional ou Incorreta

- **Causa:** O useEffect pode ter sido incluído, mas colocado dentro de um bloco condicional ou de maneira que ele não seja executado como esperado. Isso pode levar ao erro, pois o hook é importado, mas nunca é usado de fato.
- **Solução:** Verifique se o useEffect está sendo utilizado de maneira correta e que ele é chamado durante o ciclo de vida do componente conforme o esperado.

Exemplo de código correto com useEffect:

```
import React, { useState, useEffect } from 'react';



function LoginPopup() {

  const [isOpen, setIsOpen] = useState(false);

  useEffect(() => {

    if (isOpen) {
```

```
    console.log('O popup de login foi aberto.');
  }
 }, [isOpen]); // O efeito será executado toda vez que 'isOpen' mudar

 return (
  <div>
   <button onClick={() => setIsOpen(!isOpen)}>Abrir Popup</button>
   {isOpen && <div>Popup de Login</div>}
  </div>
 );
}

export default LoginPopup;
```

4. Importação de useEffect para Funções Não Componente

- **Causa:** Em alguns casos, o useEffect pode ser importado em arquivos onde ele não é necessário, como dentro de funções auxiliares ou arquivos onde você não está lidando com componentes React.
- **Solução:** Certifique-se de que o useEffect é importado apenas dentro dos componentes React onde ele será realmente utilizado. Se você não estiver dentro de um componente React, remova a importação de useEffect.

Exemplo de código corrigido (sem necessidade de useEffect):

```
// Remova importação de useEffect se não for necessário em funções auxiliares
```

Resumo

1. **Remova a importação de useEffect** se não for realmente necessário no componente.
2. **Utilize useEffect corretamente** para efeitos colaterais, como modificações do DOM, requisições de API, entre outros.
3. **Verifique a lógica do useEffect** para garantir que ele está sendo chamado corretamente e dentro do ciclo de vida do

	componente. 4. **Evite importar useEffect em arquivos ou funções que não precisam dele**. Essas são as principais causas para o erro 'useEffect' is defined but never used e as soluções adequadas para corrigir o problema.
Comentários	Outro exemplo para demonstrar que é necessário revisar o código ao longo de sua produção.

Erro no. 12

Identificador	E12
Ferramentas envolvidas	ESLint
Categoria	react/prop-types
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/components/Navbar/Navbar.jsx>, linha 7
Mensagem do erro	<'setShowLogin' is missing in props validation>
Trecho do código	<const Navbar = ({ setShowLogin }) => {>
Proposta de solução	Validar esse argumento como sendo uma prop já resolveria isso, pois o Eslint pode vir a mostrar esse erro quando não identifica essa validação
Proposta de solução (ferramenta IA)	O erro 'setShowLogin' is missing in props validation ocorre quando uma prop é passada para um componente, mas não há uma validação explícita dessas props usando a biblioteca prop-types. A validação de props é uma boa prática para garantir que o componente receba os tipos corretos de dados e evitar erros em tempo de execução. Causas Comuns 1. **Falta de validação de prop**: Quando uma prop é passada para um componente, mas não é validada com PropTypes, o ESLint gera esse erro. 2. **Componente esperando uma prop sem definição no PropTypes**: Pode ser que a prop setShowLogin esteja sendo passada para o componente Navbar, mas você não declarou

no PropTypes do componente.

Soluções

1. Adicionar Validação de Props Usando PropTypes

Você pode usar o PropTypes para declarar as props esperadas e seus tipos. No seu caso, parece que setShowLogin é uma função que deve ser passada como prop, então a solução seria adicionar uma validação para garantir que ela seja do tipo func.

Aqui está como você pode corrigir isso:

```
import React from 'react';
import PropTypes from 'prop-types';  // Importe o PropTypes


function Navbar({ setShowLogin }) {
  return (
    <nav>
      <button onClick={() => setShowLogin(true)}>Login</button>
    </nav>
  );
}

// Validação das props
Navbar.propTypes = {
  setShowLogin: PropTypes.func.isRequired,  // Valide que setShowLogin é uma função obrigatória
};

export default Navbar;
```

2. Adicionar defaultProps (opcional)

Caso você queira fornecer um valor padrão para a prop setShowLogin (no caso de não ser passada), você pode usar defaultProps. Isso pode ser útil se o comportamento padrão da função for um simples console.log() ou algo semelhante.

Exemplo:

```
Navbar.defaultProps = {
  setShowLogin: () => console.log('Função setShowLogin não foi passada'),
};
```

	3. Certificar-se de que a prop é passada corretamente Se o setShowLogin estiver sendo passado para o componente de outro componente pai, verifique se a prop está sendo corretamente passada: Componente Pai: import React, { useState } from 'react'; import Navbar from './Navbar'; function App() { const [showLogin, setShowLogin] = useState(false); return (<Navbar setShowLogin={setShowLogin} /> {showLogin && Login Form }); } export default App; Resumo ● **Erro**: Você não está validando a prop setShowLogin usando PropTypes, o que é gerado pelo ESLint como um erro. ● **Solução**: Adicione a validação de setShowLogin usando PropTypes.func.isRequired para garantir que ele seja uma função e que sempre seja passado para o componente. ● **Opção adicional**: Adicione defaultProps caso queira fornecer um valor padrão para a prop caso ela não seja passada. Essa é a maneira adequada de resolver o erro 'setShowLogin' is missing in props validation.
Comentários	Neste caso, não é nada que afete diretamente na aplicação, mas o analisador detecta como sendo uma fonte de um provável erro em outra parte do código, como erro em uma parte que chame essa prop no código.

Erro no. 13

Identificador	E13
Ferramentas envolvidas	ESLint
Categoria	react/react-in-jsx-scope
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/context/StoreContext.jsx>, linha 84
Mensagem do erro	<'React' must be in scope when using JSX >
Trecho do código	< <StoreContext.Provider value={contextValue}> >
Proposta de solução	Importar a biblioteca React no começo do código já funcionaria, pois no código são importadas apenas funções de React, mas não o React em si
Proposta de solução (ferramenta IA)	O erro “ 'React' must be in scope when using JSX” ocorre porque, no React 17 e versões anteriores, o JSX precisava que o React fosse importado no arquivo para que ele pudesse ser usado. Esse erro está sendo gerado porque o JSX está sendo usado no arquivo StoreContext.jsx, mas o React não foi importado. Causa do Erro ● **Causa:** Em versões anteriores do React (antes da 17), era necessário importar explicitamente o React em arquivos que usavam JSX. A partir do React 17, a importação explícita de React não é mais necessária para usar JSX, mas algumas configurações de linting ainda podem exigir essa importação. Soluções 1. Importar o React Se você está usando uma versão do React anterior à 17 ou se o seu projeto está configurado para requerer a importação do React mesmo na versão 17, basta importar o React no início do arquivo. Isso resolve o erro imediatamente. Exemplo de correção: import React from 'react'; // Adicione a importação do React import { createContext, useContext, useState } from 'react'; const StoreContext = createContext(); export const StoreProvider = ({ children }) => {

```
  // Lógica do contexto aqui

  return (
    <StoreContext.Provider value={{}}>
      {children}
    </StoreContext.Provider>
  );
};

export const useStore = () => useContext(StoreContext);
```

2. Verificar a versão do React

Se você estiver usando o React 17 ou superior e não quiser importar o React explicitamente em cada arquivo, você pode tentar ajustar a configuração do ESLint ou verificar se a versão do React e as dependências estão atualizadas corretamente.

Verifique a versão do React no seu package.json:

```
 "react": "^17.0.0" // ou superior
```

1.

Caso esteja usando o React 17 ou superior e o erro persista, é possível ajustar a configuração do ESLint para não exigir a importação de React. Para isso, no arquivo .eslintrc.js ou similar, você pode definir a regra para permitir o JSX sem a importação explícita do React:

Exemplo de configuração ESLint para React 17+ (sem exigir importação do React):

```
 module.exports = {
  parserOptions: {
    ecmaVersion: 2020,
    sourceType: 'module',
  },
  extends: [
    'eslint:recommended',
    'plugin:react/recommended',
  ],
  rules: {
    'react/react-in-jsx-scope': 'off',  // Desabilita a exigência de importar o React
  },
};
```

	2. 3. Confirmar a versão do ESLint e do Plugin React Certifique-se de que as versões do eslint-plugin-react e do ESLint são compatíveis com o React 17 ou superior. Instale ou atualize o eslint-plugin-react: npm install eslint-plugin-react --save-dev 1. 2. Certifique-se de que sua configuração do ESLint está alinhada com a versão do React que você está usando. Resumo ● **Erro**: O React precisa ser importado explicitamente em arquivos com JSX no React 16 e versões anteriores. ● **Solução 1**: Importe React no início do arquivo onde você está utilizando JSX. ● **Solução 2**: Se estiver usando o React 17 ou superior, desabilite a regra react/react-in-jsx-scope no ESLint para não exigir a importação de React. Isso deve resolver o erro 'React' must be in scope when using JSX.
Comentários	É um erro básico, mas que compromete esse componente por inteiro, pois ele não será renderizado corretamente na página e poderá comprometer a usabilidade e outros componentes.

Erro no. 14

Identificador	E14
Ferramentas envolvidas	ESLint
Categoria	no unused vars
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/pages/Cart/Cart.jsx>, linha 24
Mensagem do erro	<'index' is defined but never used>
Trecho do código	< {food_list.map((item, index) => { >
Proposta de	No código, ele fez de um jeito que não necessite usar o index, então

solução	só retirá-lo não fará diferença e solucionará
Proposta de solução (ferramenta IA)	O erro 'index' is defined but never used ocorre quando uma variável ou parâmetro é declarada, mas não está sendo utilizada no código. No seu caso, a variável index foi definida em algum lugar, provavelmente como um parâmetro de uma função ou dentro de um método (como em um map ou forEach), mas não foi realmente usada no corpo da função ou método. Causas Comuns 1. **Variável index declarada mas não utilizada**: Pode ser que você tenha declarado a variável index, talvez dentro de um .map(), .forEach(), ou em algum outro loop, mas depois não tenha utilizado ela. 2. **Parâmetro index não utilizado**: Se você está desestruturando ou passando um parâmetro index para uma função, mas não o usa dentro da função. Soluções 1. Remover a variável index se não for necessária Se o index não estiver sendo usado em nenhum lugar, você pode simplesmente removê-lo. Isso vai eliminar o erro e deixar o código mais limpo. Exemplo: Se você tiver algo como: {items.map((item, index) => ({item.name}))} Mas não estiver usando index, a correção seria: {items.map(item => ({item.name}))} Aqui, o parâmetro index foi removido porque não está sendo utilizado no corpo da função. 2. Usar o parâmetro index se for necessário Se a variável index foi declarada com a intenção de ser usada (por exemplo, para adicionar uma chave única, manipular elementos, ou

	realizar alguma lógica baseada na posição), então você deve usá-la de forma apropriada. Exemplo: {items.map((item, index) => ({index + 1}. {item.name} {/* Usando 'index' */}))} Aqui, o índice está sendo utilizado para mostrar a posição do item na lista. 3. Ignorar a variável index (se for um parâmetro não utilizado) Se você não for usar o index mas não puder ou não quiser removê-lo (por exemplo, se for parte da assinatura de uma função anônima dentro de um .map()), pode simplesmente adicionar um underscore (_) para indicar que a variável não será usada. Exemplo: {items.map((item, _) => ({item.name}))} No exemplo acima, _ é usado como um nome de variável convencionalmente indicando que a variável não será usada. Resumo ● **Erro**: A variável index é definida, mas não está sendo usada. ● **Solução 1**: Se não precisar de index, remova-o da declaração. ● **Solução 2**: Se index for necessário, utilize-o corretamente no código. ● **Solução 3**: Se não for usar index mas precisar manter a estrutura, substitua por um _ (underscore) para indicar que a variável não será utilizada. Isso deve resolver o erro 'index' is defined but never used.
Comentários	Novamente, algo que não influencia na execução do componente, mas que poderá deixar o código confuso para conferência futura.

Erro no. 15

Identificador	E15
Ferramentas envolvidas	ESLint
Categoria	no unused vars
Localização	</home/alvaro/UFC_2024-2/VeV/Tomato-FoodOrdering/Frontend/src/pages/Verify/Verify.jsx>, linha 10
Mensagem do erro	<'setSearchParams' is assigned a value but never used>
Trecho do código	<const [searchParams,setSearchParams] = useSearchParams();>
Proposta de solução	Criar uma função que utilize esse set, pois seria útil para a melhoria do site, pois poderia gerar uma certa dinamicidade e melhoraria a experiência do usuário na página
Proposta de solução (ferramenta IA)	O erro 'setSearchParams' is assigned a value but never used ocorre quando a variável setSearchParams é declarada e atribuída a algum valor, mas não é utilizada no código. Isso normalmente acontece quando você tem um hook como o useSearchParams (usado para manipulação de parâmetros de URL), mas não está utilizando a função setSearchParams. Causas Comuns 1. **Uso de setSearchParams sem necessidade**: Você declarou setSearchParams (provavelmente de um hook useSearchParams), mas nunca usou essa função para modificar os parâmetros de URL. 2. **Intenção de usar setSearchParams mas esquecimento**: Você planejou usar setSearchParams para atualizar os parâmetros da URL, mas esqueceu de implementá-lo. Soluções 1. Remover a declaração de setSearchParams se não for necessária Se você não precisa atualizar os parâmetros da URL ou não está usando setSearchParams em nenhum lugar do seu componente, a solução mais simples é remover a declaração. Exemplo: Se o seu código for algo como: import { useSearchParams } from 'react-router-dom'; function Verify() {

```
  const [searchParams, setSearchParams] = useSearchParams(); // 'setSearchParams' não é utilizado

  return (
    <div>
      Verificando...
    </div>
  );
}

export default Verify;
```

Você pode corrigir isso removendo a variável setSearchParams:

```
import { useSearchParams } from 'react-router-dom';

function Verify() {
  const [searchParams] = useSearchParams(); // Não usa 'setSearchParams'

  return (
    <div>
      Verificando...
    </div>
  );
}

export default Verify;
```

2. Usar setSearchParams se for necessário

Se você precisa usar setSearchParams para alterar os parâmetros da URL (como, por exemplo, ao alterar filtros de pesquisa ou status de verificação), você pode utilizá-lo conforme necessário. Aqui está um exemplo básico de como usar setSearchParams para atualizar a URL:

Exemplo:

```
import { useSearchParams } from 'react-router-dom';

function Verify() {
  const [searchParams, setSearchParams] = useSearchParams();
```

	`` const handleChangeParams = () => { // Atualiza os parâmetros da URL setSearchParams({ status: 'verified' }); }; return (<button onClick={handleChangeParams}>Alterar Status</button> Status: {searchParams.get('status')}); } export default Verify; ``` Nesse exemplo, o botão chama setSearchParams para alterar os parâmetros de URL quando é clicado. 3. Substituir setSearchParams por outro valor, se necessário Se você estava planejando usar setSearchParams mas, no fim, não precisou, você pode substituir sua implementação. Por exemplo, você pode configurar uma variável de estado em vez de usar os parâmetros de URL diretamente, dependendo da lógica que você está implementando. Resumo ● **Erro**: Você declarou setSearchParams, mas nunca usou. ● **Solução 1**: Remova a declaração de setSearchParams se não for necessário. ● **Solução 2**: Use setSearchParams para atualizar os parâmetros da URL se você precisar dessa funcionalidade. ● **Solução 3**: Caso não queira usar setSearchParams mas precise de algum outro valor, considere substituir sua implementação. Essas soluções devem corrigir o erro 'setSearchParams' is assigned a value but never used.
Comentários	Como dito, é um erro que não afeta diretamente a execução, mas que poderia ser visto de outra forma, podendo gerar uma melhora no site e na experiência do usuário.

3. Discussão

Foi utilizado somente a ferramenta ESLint como previsto no TB2. Percebeu-se que a ferramenta é limitada para fazer análise estática, visto que as regras sobre o que será considerado um erro ou não são definidas pelo time de testes. Assim, a depender do que for configurado no arquivo de regras, certos problemas podem não ser identificados, o que impacta diretamente na eficácia da análise estática.

Além disso, o ESLint não é capaz de detectar problemas relacionados ao comportamento ou à lógica do código, embora seja muito boa pensando em estrutura, estilo e boas práticas.

Em relação ao uso de IA, percebeu-se que, no geral, as soluções sugeridas eram satisfatórias e frequentemente úteis como suporte ao processo de desenvolvimento. No entanto, foi observado que, quase sempre, as soluções fornecidas eram mais complexas do que as implementadas pelo time, o que, em alguns casos, tornava o código menos legível e mais difícil de manter. Outro fator importante é que a IA chegou a errar como em E2.

4. Conclusão

Esse documento apresentou a análise estática do projeto do TB1 com a ferramenta ESLint. Discussões foram feitas para compreender as diferentes soluções propostas pela IA com as do time de testes e a aplicabilidade da ferramenta ESLint em diferentes contextos.

5. Referências

ATHRVANAIK. GitHub - AthrvaNaik/Tomato-FoodOrdering: The Deployed Website. Disponível em: <https://github.com/AthrvaNaik/Tomato-FoodOrdering?tab=readme-ov-file>. Acesso em: 13 jan. 2025.

Glossário

Termo	Definição
Props	São propriedades passadas para componentes em React, usadas para personalizar seu comportamento e exibir informações dinâmicas.
Hooks	Funções em React que permitem a adição de lógicas e estados ao componente funcional, como useState, useEffect, useContext, etc.