

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
ETEC ZONA LESTE**

**Ensino Médio com Habilitação Profissional de Técnico em
Desenvolvimento de Sistemas - AMS**

**Marcos Alves da Cruz Junior
Matheus Brito Alves
Victor Gabriel Monteiro Costa
Wayne André Rocha**

DevsTogether

**São Paulo
2022**

**Marcos Alves da Cruz Junior
Matheus Brito Alves
Victor Gabriel Monteiro Costa
Wayne André Rocha**

**DevsTogether:
Colaboração no Desenvolvimento de Software**

Trabalho de Conclusão de Curso apresentado para conclusão do Curso Técnico em Desenvolvimento de Sistemas da Etec Zona Leste, orientado pelo Professor Ediney, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

**São Paulo
2022**

DEDICATÓRIA

A

Deus

Pela força para perseverar e a sabedoria prática dada durante a produção do TCC.

Professor Ediney Ciasi Barreto e Vilma Cardoso

Por todo o suporte, apoio e incentivo dado aos membros.

Colegas do 3º DS – 2022.

RESUMO

Nosso projeto tem a proposta da criação de uma comunidade para desenvolvedores de software, porém com o diferencial de ser mais dinâmico que outros sites, permitindo aos usuários uma comunicação em tempo real para problemas que precisam de mais tempo para serem resolvidos e para usuários que desejam soluções rápidas, a opção de utilizar nosso fórum, onde a comunidade participara ativamente respondendo dúvidas sobre os códigos, todas as perguntas respondidas vão ser salvas para que qualquer um com o mesmo problema possa saber como resolve-lo.

Palavras-chave: Software. Colaboração. Programação. Comunidade.

ABSTRACT

Our project has the proposal of the creation of a community for software developers, but with the differential of being more dynamic than others websites, allowing the users a real time communication for problems that need more time to be solved and for users that want a faster solution, the option to use our forum, where the community will participate actively responding questions about the codes, all the questions will be saved for anyone with the same problem so that they can know how to solve it.

Keywords: *Software. Collaboration. Programming. Community.*

LISTA DE ABREVIATURAS E SIGLAS

Application Programming Interface (API)

Banco de dados (BD)

Banco de dados relacional (BDR)

Cascading Style Sheets (CSS)

Hypertext Markup Language (HTML)

JavaScript (JS)

JavaScript Extendible (JSX)

Material UI (MUI)

Single Page Application (SPA)

User Interface (UI)

LISTA DE FIGURAS

Figura 1 - Exemplo de código HTML.....	12
Figura 2 - Estilos da página.....	13
Figura 3 - Resultado dos estilos passados pelo desenvolvedor	13
Figura 4 - Exemplo JavaScript	15
Figura 5 - Tipagem.....	16
Figura 6 - <i>Auto complete</i>	16
Figura 7 - Exemplo de componentes.....	18
Figura 8 - Exemplo usando o componente criado.....	18
Figura 9 - Resultado da aplicação com a utilização do componente	19
Figura 10 - Exemplo de Propriedade sendo passada para o componente.....	19
Figura 11 - Resultado do exemplo utilizando o componente com propriedades	20
Figura 12 - Exemplos de CSS dentro de um código.....	20
Figura 13 - Exemplo de JSX sendo utilizado dentro de um código	21
Figura 14 - Exemplo de exportação no React com styled-components	23
Figura 15 - Exemplo de importação no React com styled-components.....	23
Figura 16 - Teste de verificação dentro de um código.....	24
Figura 17 - Resultado do teste no Jest.....	24
Figura 18 - Componentes MUI Renderizando	25
Figura 19 - Exemplo de página criada apenas com <i>Material-UI</i>	26
Figura 20 - Funcionamento de uma aplicação NodeJS.....	27
Figura 21 - Arquitetura do Electron	29
Figura 22 - DOM do VSCode visto pelo DevTools	29
Figura 23 - Exemplo de DER.....	31
Figura 24 - Exemplo do funcionamento do Prisma com o banco de dados	32
Figura 25 - Casos de Uso UML	36
Figura 26 - Diagrama de fluxo sobre Oauth	37
Figura 27 - DER do projeto	38
Figura 28 - MER do projeto.....	39

SUMÁRIO

1. INTRODUÇÃO	9
2. REFERENCIAL TEÓRICO	11
2.1. HTML.....	11
2.2. CSS	12
2.3. JAVASCRIPT.....	14
2.4. TYPESCRIPT.....	15
2.5. REACTJS.....	16
2.6. JEST	23
2.7. MATERIAL UI.....	25
2.8. NODE.JS	27
2.9. NEXTJS.....	27
2.10. ELECTRON.....	28
2.11. BANCO DE DADOS	29
2.11.1. Banco de Dados relacional.....	30
2.11.2. Casos de uso	31
2.11.3. Modelagem de dados	31
2.11.4. Prisma.....	31
2.11.5. Supabase.....	32
2.11.6. Ferramentas em comum	32
3. DESENVOLVIMENTO	33
3.1. PROBLEMAS NA COMUNIDADE DEV E SOLUÇÕES.....	33
3.2. REGRAS DE NEGÓCIO.....	34
3.3. REQUISITOS NÃO FUNCIONAIS	35
3.4. REQUISITOS FUNCIONAIS	35
3.5. CASOS DE USO	35
3.6. DIAGRAMAS DE FLUXO	37
3.7. DER	38
3.8. MER.....	39
4. REFERÊNCIAS.....	40

1. INTRODUÇÃO

Nosso projeto é uma plataforma com uma grande comunidade engajada de desenvolvedores que se ajudam e compartilham experiência entre si da forma mais prática e dinâmica possível. Assim será possível que a comunidade de desenvolvedores evolua de forma muito mais rápida.

Além dos recursos de texto já existentes e amplamente usados serão usados métodos dinâmicos e interativos, com vídeo/áudio chamadas com *chat* e compartilhamento remoto. Assim a pessoa poderá ter acesso ao mesmo ambiente de desenvolvimento que a outra pessoa, trabalhando juntos para resolver um problema.

A plataforma também terá um sistema de gamificação, com pontos de reputação, prêmios e reconhecimentos que darão incentivo, objetivo e foco aos desenvolvedores na comunidade.

O motivo do nosso projeto é simples, o mercado de desenvolvimento vem sofrendo com a escassez de profissionais de qualificados. Isso se deve a vários fatores como: Falta de orientação, falta de comunicação e grande curva de aprendizado.

Todo profissional de TI já foi um estudante, e naturalmente, como em qualquer área existem desafios que atrasam o próprio desenvolvimento, tais como os mencionados acima. Por isso saber pedir ajuda e ter uma boa comunicação é algo de muita importância para crescer no mercado de TI. A pesquisa 2021 Developer Survey mostra que a maioria dos programadores buscam ajuda por 3 fontes: Google, comunidade do StackOverflow e por colegas ou amigos.

A quantidade de programadores que pedem ajuda é muito grande, tanto que 30% deles visitam o StackOverflow diariamente ou algumas vezes por semana, é de fato uma comunidade bem engajada.

Porém há um problema, falta de dinamismo, pois na grande maioria das vezes só é possível dar boas respostas com boas perguntas, e isso é algo que programadores pouco experientes não sabem como fazer. De acordo com a pesquisa 2021 Developer Survey, 59% dos desenvolvedores entrevistados aprenderem a codificar por meio de recursos online, como vídeos e blogs. Muitos desses materiais na internet não são didáticos e podem dificultar no aprendizado causando ainda mais confusão.

Cerca de 15% dos entrevistados são desenvolvedores com mais experiência e tem mais anos de trabalho no ramo, enquanto os menos experientes são apenas 4%. Isso mostra a importância de que haja mais compartilhamento de experiência, e de forma mais prática e dinâmica. Novos desenvolvedores normalmente têm dificuldades como: Aprender uma nova linguagem, confusão para começar a carreira, trabalhar em equipe, falta de comunicação e gastam muito tempo tentando corrigir um pequeno erro.

A falta de profissionais com perfil e qualificação em tecnologia da informação (TI) é uma dificuldade enfrentada pelas empresas em geral, as quais investem crescentemente em TI e cada vez mais são dependentes e influenciadas pela tecnologia. (Ursula Barreto, 2003 pg.7)

2. REFERENCIAL TEÓRICO

Vamos explicar sobre as tecnologias utilizadas para a produção do projeto e qual função cada uma vai ter em nosso projeto

2.1. HTML

O *HyperText Markup Language* (HTML) é o bloco de construção mais básico da web, ele que define a estrutura do conteúdo da web. Hipertexto se refere aos links que conectam as páginas eles são fundamentais no projeto. De acordo com (Flatschart, 2011 p. 9) esta linguagem permite a criação de documentos estruturados em títulos, parágrafos, listas, links, tabelas, formulários e em muitos outros elementos nos quais pode, ser incorporadas imagens e objetos como animação ou vídeo. Sendo assim o HTML a ferramenta de maior importância na criação de um website

Segundo (Silveira,2014 pg.9) um arquivo HTML nos permite não apenas apresentar informações que foram colocadas dentro dele, podemos realizar operações. E para dar um comando assim como em qualquer outra linguagem, nós precisamos utilizar as tags, mas o que são nas tags?

De acordo com (Manzano, 2010 pg.5) uma tag é uma palavra reservada delimitada pelos símbolos "< >" e são usadas para definir a estrutura física de uma página, como alterar fontes, incluir linhas ou imagens e também para indicar links de navegação

As principais tags são:

head: Local para declarar todas as informações, como título e metadados da sua página.

title: Define o título.

body: Local para declarar todos os elementos que irão compor o corpo da página.

form: O form faz com que o usuário interaja com o usuário, fazendo ele submeter dados ao um programa que se encarregará dos dados fornecidos.

link: Como dito anteriormente o link é uma função do HTML que permite inserir os hiperlinks dentro de diversos elementos dentro do código, como imagens e textos.

main: Sua função é definir o conteúdo principal dentro do body.

div: É utilizando para agrupar elementos dentro da página, sua função principal estilizar a página.

Na figura a seguir temos um formulário desenvolvido em HTML:

Figura 1 - Exemplo de código HTML

```

○ ○ ○

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Validación de Formulario con Javascript</title>
  <link rel="stylesheet" href="https://necolas.github.io/normalize.css/8.0.1/normalize.css">
  <link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="css/estilos.css">
</head>
<body>
  <main>
    <form action="" class="formulario" id="formulario">
      <!-- Grupo: Usuario -->
      <div class="formulario__grupo" id="grupo__usuario">
        <label for="usuario" class="formulario__label">Usuario</label>
        <div class="formulario__grupo-input">
          <input type="text" class="formulario__input" name="usuario" id="usuario"
placeholder="john123">
          <i class="formulario__validacion-estado fas fa-times-circle"></i>
        </div>
        <p class="formulario__input-error">El usuario tiene que ser de 4 a 16 dígitos y solo puede
contener numeros, letras y guion bajo.</p>
      </div>
    </form>
  </main>
</body>
</html>

```

Fonte: <https://github.com/falconmasters/formulario-css-grid/>

2.2. CSS

Abreviação para *Cascading Style Sheet* (CSS), é usado para estilizar elementos em uma linguagem de marcação como o HTML por exemplo. De acordo com (Quierelli, 2012 pg.8) CSS serve para formatar o conteúdo das páginas, tais como, cor de fundo da página, estilo de textos, disposição dos conteúdos e imagens.

Segundo (Jobstraibizer, 2019 pg. 8) a construção de uma folha de estilos pode ser realizada através de qualquer editor de textos como bloco de notas, ou editores HTML. (Oliveira, 2017 pg 15) É necessário identificar o elemento do texto que se deseja modificar o estilo, e essa identificação é feita escrevendo o nome da tag correspondente. Algumas das tags usadas no CSS são:

Font-size: Para definir o tamanho da fonte, os valores usados são px (pixels) e em porcentagens.

Font-weight: Define o peso da fonte.

Font-Style: O estilo, pode usar para os valores normal, oblique ou italic.

Line-Height: Define o espaçamento entre as linhas, como no tamanho da fonte, os valores usados podem ser px, cm, %, etc.

Text-align: usada para determinar o tipo de alinhamento, os valores são *left*, *right*, *center*.

Color: Usado para cores, podendo usar os valores do Photoshop, nome das cores em inglês ou usar o sistema RGB.

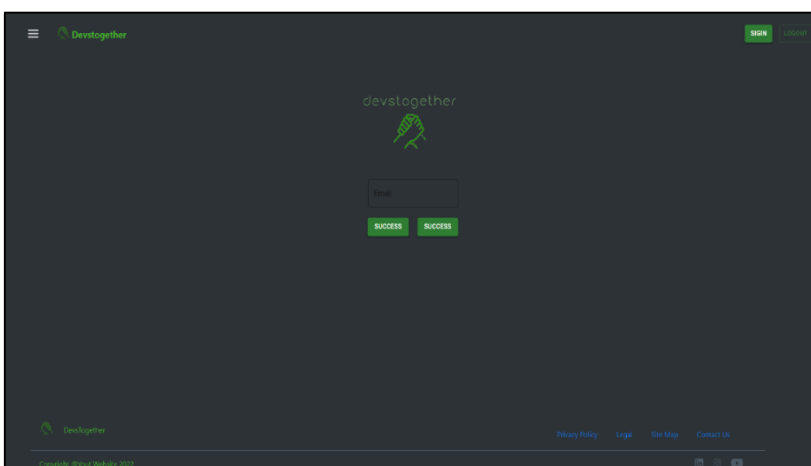
Abaixo temos um exemplo de como funciona o CSS na construção de uma página web. Na primeira imagem temos todos os códigos passados para o desenvolvedor sobre as cores de fundo tamanho e fonte que a página terá, na segunda imagem podemos ver o resultado de todas essas informações passadas pelo desenvolvedor:

Figura 2 - Estilos da página

```
body {
  background-color: #2D3237;
  position: absolute;
  top: 0;
  left: 0;
  z-index: 11;
  width: 100%;
  height: 100%;
  min-height: 100vh;
  font-family: -apple-system, BlinkMacSystemFont, Segoe UI, Roboto, Oxygen,
  Ubuntu, Cantarell, Fira Sans, Droid Sans, Helvetica Neue, sans-serif;
}
```

Fonte: Autoria própria

Figura 3 - Resultado dos estilos passados pelo desenvolvedor



Fonte: (Autoria própria)

2.3. JAVASCRIPT

É uma linguagem de programação que permite implementar itens complexos em páginas web, conteúdos dinâmicos como imagens animadas por exemplo, ele é a terceira camada das tecnologias padrão web, ou seja, é de extrema importância para o visual de um website. É inserido de maneira similar ao CSS, mas o JavaScript utiliza o elemento `<script>`. Como falado por (Flanagan, 2004 pg.1) Java Script é a linguagem de programação mais onipresente na história e faz parte da tríade de tecnologias que todos desenvolvedores web devem conhecer. Uma linguagem que está presente em jogos de celular e computador por exemplo.

Graças ao JavaScript o usuário pode interagir com o site, pois ele deixa atrativo. (Grillo, 2008 pg.1) diz que é uma linguagem dinâmica e que atualmente empresas como Google utilizam em seus sites e por isso são tão populares, pela interação.

São usadas as variáveis para armazenar os dados como se fossem as tags, segundo (Prescott 2016.cap3) variáveis são recipientes de dados e valores, podendo usar expressões para realizar cálculos

Algumas variáveis e estruturas de condições usadas frequentemente no JavaScript:

const: É um usado para criar variáveis e armazenar valores dados pelo usuário

Switch: Armazena valores e executa um bloco de códigos de forma condicional, utilizado para analisar diversos valores diferente de uma mesma variável, um ótimo recurso na construção de códigos de autenticação.

Figura 4 - Exemplo JavaScript

```

○ ○ ○

const formulario = document.getElementById('formulario');
const inputs = document.querySelectorAll('#formulario input');

const expresiones = {
  usuario: /^[a-zA-Z0-9_\-]{4,16}$/, // Letras, numeros, guion y guion_bajo
  nombre: /^[a-zA-ZÀ-ÿ\s]{1,40}$/, // Letras y espacios, pueden llevar acentos.
  password: /^[^].{4,12}$/, // 4 a 12 digitos.
  correo: /^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$/,
  telefono: /^\d{7,14}$/ // 7 a 14 numeros.
}

const campos = {
  usuario: false,
  nombre: false,
  password: false,
  correo: false,
  telefono: false
}

const validarFormulario = (e) => {
  switch (e.target.name) {
    case "usuario":
      validarCampo(expresiones.usuario, e.target, 'usuario');
      break;
    case "nombre":
      validarCampo(expresiones.nombre, e.target, 'nombre');
      break;
    case "password":
      validarCampo(expresiones.password, e.target, 'password');
      validarPassword2();
      break;
    case "password2":
      validarPassword2();
      break;
    case "correo":
      validarCampo(expresiones.correo, e.target, 'correo');
      break;
    case "telefono":
      validarCampo(expresiones.telefono, e.target, 'telefono');
      break;
  }
}

```

Fonte: <https://github.com/falconmasters/formulario-css-grid/>

2.4. TYPESCRIPT

De acordo com (Oliveira), o JavaScript se tornou uma linguagem fraca para o desenvolvimento e manutenção de grandes aplicações. Logo o TypeScript veio para ajudar nas necessidades que o Java não consegue suprir.

É mantida pela Microsoft, que adiciona várias características ao Java, códigos escritos em Type são traduzidos em Java para ser executado no ambiente alvo seja o navegador ou Node (Anjos, 2017). Pode se dizer que o TypeScript é um aprimoramento e cobre todas as necessidades como dito anteriormente.

Segundo (Silva, 2019) TypeScript é mais uma checagem para tornar códigos Java mais seguros, já que o Type adiciona esse suporte para detectar o que está errado em seu código.

Mas não se engane, TypeScript é uma linguagem própria, pois de acordo com (Merlin 2019), afirma que possui módulos, classes e interfaces e que os programadores migrem para esta linguagem pois ela tende a evoluir, abaixo temos alguns exemplos de como funciona a tipagem e o *auto complete* do TypeScript:

Figura 5 - Tipagem

```

class MathController {
  public index (req: Request, res: Response): Response {
    return res.
  }
}

export default

```

The dropdown menu shows the following options:

- listeners
- locals
- location
- off
- on
- once
- pipe
- prependListener
- prependOnceListener
- rawListeners
- redirect
- removeAllListeners

The description for the selected `res.json` property is: (property) Response.json: S * end. Below it, it says "Send JSON response." and "Examples:" followed by code snippets: `res.json(null);`, `res.json({ user: 'tj' });`, `res.status(500).json('oh noes!');`, and `res.status(404).json('I dont`.

Fonte: <https://blog.rocketseat.com.br/content/images/2019/03/intellisense-vscode-typescript.png>

Figura 6 - Auto complete

```

app.ts > ...
1 let nome: string;
2 nome = 'treinaWeb';
3
4
5
6
7
8 nome = 23;

```

The error message is: "Type 'number' is not assignable to type 'string'. ts(2322)". The dropdown menu shows the following options:

- let nome: string
- View Problem No quick fixes available

Fonte: <https://blog.rocketseat.com.br/content/images/2019/03/intellisense-vscode-typescript.png>

2.5. REACTJS

Segundo (Lins) ReactJS é uma biblioteca *Front-end* que se baseia na linguagem JavaScript, cujo principal intuito é o de facilitar o desenvolvimento de interfaces, E o que faz o ReactJS ser diferente das demais bibliotecas é que ele é baseado em componentes para aplicações web, E também a biblioteca pode ser usado para o desenvolvimento de

aplicações mobile. A estrutura do *ReactJS* é composta por conceitos importantes sendo eles os Componentes, Propriedades, Estilização e sintaxe JavaScript Syntax Extension (JSX) é necessário ter uma base bem fundamentada destes quatro conceitos, abaixo veremos a função que cada um exerce dentro do ReactJS:

- **Componentes:** Segundo documentação oficial os componentes no React são como funções do JavaScript, eles podem aceitar propriedades e renderiza novo elemento na tela, esta técnica é conhecida como JSX, que é quando a linguagem JavaScript proporciona uma sintaxe familiar à que os desenvolvedores estão acostumados que tem semelhança com o HTML, Abaixo temos um exemplo de componentes de um app com o nome contador com a função de contar os itens selecionados pelo usuário e mostrar a quantidade de itens selecionados no carrinho do usuário, este primeiro componente é responsável por fazer uma parte da lógica do programa a de contar já o segundo é responsável por fazer esta parte lógica aparecer na tela do usuário, E em seguida temos o resultado final com a ação dos componentes que forma utilizados:

Figura 7 - Exemplo de componentes

```

○○○
import React, { Component } from "react";

class Counter extends Component {
  render() {
    return (
      <div>
        <div className="row">
          <div className="">
            <span style={{ fontSize: 24 }} className={this.getBadgeClasses()}>
              {this.formatCount()}
            </span>
          </div>
          <div className="">
            <button
              className="btn btn-secondary"
              onClick={() => this.props.onIncrement(this.props.counter)}
            >
              <i className="fa fa-plus-circle" aria-hidden="true" />
            </button>
            <button
              className="btn btn-info m-2"
              onClick={() => this.props.onDecrement(this.props.counter)}
              disabled={this.props.counter.value === 0 ? "disabled" : ""}
            >
              <i className="fa fa-minus-circle" aria-hidden="true" />
            </button>
            <button
              className="btn btn-danger"
              onClick={() => this.props.onDelete(this.props.counter.id)}
            >
              <i className="fa fa-trash-o" aria-hidden="true" />
            </button>
          </div>
        </div>
      </div>
    );
  }

  getBadgeClasses = () => {
    let classes = "badge m-2 badge-";
    classes += this.props.counter.value === 0 ? "warning" : "primary";
    return classes;
  };

  formatCount = () => {
    const { value } = this.props.counter;
    return value === 0 ? "Zero" : value;
  };
}

export default Counter;

```

Fonte: <https://github.com/falconmasters/formulario-css-grid/>

Figura 8 - Exemplo usando o componente criado

```

○○○
import React, { Component } from "react";
import Counter from "./counter";

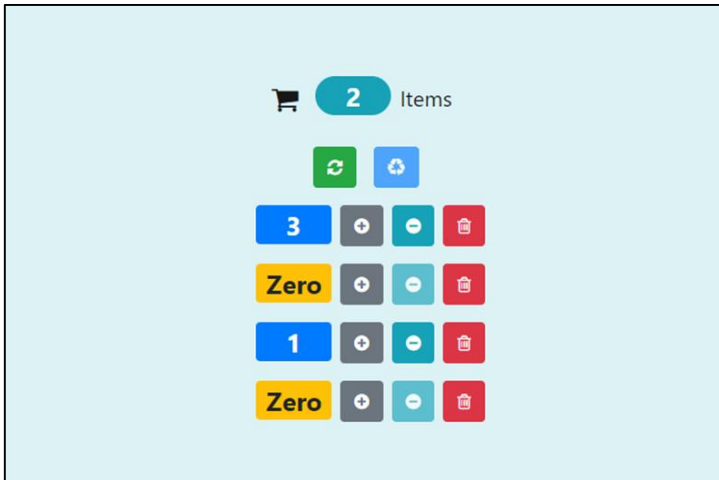
class Counters extends Component {
  render() {
    const { onReset, onIncrement, onDelete, onDecrement, counters, onRestart } =
      this.props;
    return (
      <div>
        <div className="row">
          <div className="">
            <button
              className="btn btn-success m-2"
              onClick={onReset}
              disabled={counters.length === 0 ? "disabled" : ""}
            >
              <i className="fa fa-refresh" aria-hidden="true" />
            </button>
            <button
              className="btn btn-primary m-2"
              onClick={onRestart}
              disabled={counters.length !== 0 ? "disabled" : ""}
            >
              <i className="fa fa-recycle" aria-hidden="true" />
            </button>
          </div>
          <div>
            {counters.map((counter) => (
              <Counter
                key={counter.id}
                counter={counter}
                onIncrement={onIncrement}
                onDecrement={onDecrement}
                onDelete={onDelete}
              />
            ))}
          </div>
        </div>
      </div>
    );
  }
}

export default Counters;

```

Fonte: <https://github.com/falconmasters/formulario-css-grid/>

Figura 9 - Resultado da aplicação com a utilização do componente



Fonte: <https://github.com/arnab-datta/counter-app>

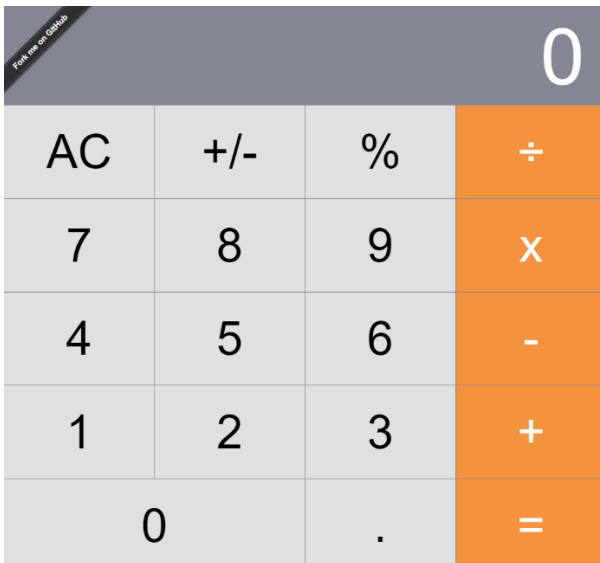
- **Propriedades:** Propriedades é tudo aquilo que podemos passar para o componente alguma informação, essa informação pode ser tanto para aplicar algum tipo de ou para exibir algo do próprio componente, abaixo temos o exemplo de uma calculadora feita em React e na primeira figura temos um exemplo de propriedade sendo passada para um botão e essa propriedade passa uma função na qual o botão fará na segunda temos o resultado final do programa:

Figura 10 - Exemplo de Propriedade sendo passada para o componente

```
○○○  
  
export default class ButtonPanel extends React.Component {  
  static propTypes = {  
    clickHandler: PropTypes.func,  
  };  
  
  handleClick = buttonName => {  
    this.props.clickHandler(buttonName);  
  };  
};
```

Fonte: <https://github.com/arnab-datta/counter-app>

Figura 11 - Resultado do exemplo utilizando o componente com propriedades



Fonte: <https://github.com/ahfarmer/calculator>

- **Estilização:** Estilização é mais uma parte de design do React onde uma das técnicas mais usadas é o CSS-in-JS que se refere como o JavaScript pode ser usado para estilizar componentes, essa é uma das partes mais importantes do ReactJS porque essa parte da sua estrutura é a mais responsável pelo design e beleza da sua aplicação, a estilização no ReactJS é bem parecida com o CSS abaixo temos um exemplo de aplicações.

Figura 12 - Exemplos de CSS dentro de um código

```

○○○
body {
  background-color: #172b4d;
}

input {
  background-color: #fff !important;
  border-radius: 44px !important;
  width: 90% !important;
  padding: 0px 15px !important;
}

input:focus {
  border-bottom: none !important;
  box-shadow: none !important;
}

label {
  display: block;
  color: #fff !important;
  font-size: 1rem !important;
}

```

Fonte: Autoria própria

- **JavaScript Syntax Extension (JSX):** É chamada JSX uma extensão de sintaxe para JavaScript. Quando usamos JSX no ReactJS é para descrever como a UI deverá aparecer na tela do usuário, JSX é famoso por tornar a estrutura e a renderização dos componentes familiares aos que os desenvolvedores estão acostumados a utilizar, que no caso é o HTML, abaixo temos um exemplo de JSX:

Figura 13 - Exemplo de JSX sendo utilizado dentro de um código

```
function formatName(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'Harper',
  lastName: 'Perez'
};

const element = (
  <h1>
    Hello, {formatName(user)}!
  </h1>
);

ReactDOM.render(
  element,
  document.getElementById('root')
);
```

Fonte: Autoria própria

Segundo (de Camargos, J. G. C., Coelho, J. F., Villela, H. F., & Aramuni, J. P.) O React é muito trabalhoso programar sem problemas, tendo um benefício que a maioria dos erros comuns que ocorrem são mostrados em seu console e com as respectivas soluções, o que ajuda na hora de encontrar o erro em seu código.

O React também possui um ótimo algoritmo de manipulação conhecido como DOM, ele atualiza apenas elementos essenciais a tela do usuário e não todos os elementos, o que deixa a experiência do usuário mais fluída e rápida.

Para iniciar o desenvolvimento do seu projeto é necessário que o desenvolvedor instale todos os componentes necessários por meio do gerenciador de pacotes chamado NPM. Existem muitos pacotes no NPM, mas o usuário só irá usar alguns deles, para isso é necessário que o usuário saiba quais componentes ele irá usar em seu projeto.

Segundo (Almeida, Pedro H. Marques) o ReactJS é muito eficaz na hora de desenvolver interfaces de usuários e que o React é uma ótima biblioteca para o desenvolvimento de aplicações de média e pequena escala este foi um dos motivos pelo qual escolhemos o

ReactJS para o nosso projeto por apresentar tantos benefícios e facilitar e tornar a experiência do usuário muito melhor e dinâmica.

- **Styled Components:** Segundo (SILVA, Matheus Procópio da) O *styled-components* é uma forma de modelar estilos de um componentes dentro do React, ele permite a estilização de componentes sem a criação de arquivos, possibilitando ao desenvolvedor uma ótima experiência no seu desenvolvimento de sua aplicação, Segundo (Santiago, Gabriel Mendes de Souza.) o principal foco da biblioteca é a estilização, está entre uma das bibliotecas mais utilizadas no mundo e os benefícios são diversos disponibilizando ao desenvolvedor uma estilização além de criativa ótima para o seu código, deixando-o assim muito mais compreensivo e limpo, em casos de trabalhos em equipes como acontece em muitas empresas este benefício é fundamental, esta biblioteca usa a linguagem JavaScript, isso ajuda o desenvolvedor na hora de fazer a importação e exportação de arquivos de estilo. A biblioteca possibilita criar nomes de classes de forma automática.

Segundo (Oliveira, Jackson Vinicius Faria de.) o Styled Components permitem o condicionamento interno do componente, Para usar o styled components dentro do React basta fazer a instalação que está disponível na documentação oficial do React, abaixo temos um exemplo de como o styled components funciona e ajuda na importação e exportação do arquivos de estilo dentro de um código, Abaixo temos um exemplo de como funciona a exportação e importação dentro do código:

Figura 14 - Exemplo de exportação no React com styled-components

```

import styled from "styled-components"

export const Rodapé = styled.div`
  footer {
    display: flex;
    flex-direction: row-reverse;
    justify-content: space-between;
    align-items: center;
    background-color: #2D3237;
    width: 100%;
    align-items: flex-end;
    margin: auto;
    bottom: 0;
    @media (max-width: 737px) { //quando a tela for pequena
      width: 20%;
      align-items: center;
    }
  }
}

```

Fonte: Autoria própria

Figura 15 - Exemplo de importação no React com styled-components

```

import { Rodapé } from './style'; //importando o rodapé com os estilos da pasta style

function Footer(): JSX.Element {
  return (
    <Rodapé> //fazendo ele ficar visível na tela do usuário
      <footer>
        <div className="footerLinks">
          <ul>
            <a href="#">Privacy Policy</a>
            <a href="#">Legal</a>
            <a href="#">Site Map</a>
            <a href="#">Contact Us</a>
          </ul>
        </div>
      </Rodapé>
    </div>
  );
}

```

Fonte: autoria própria

2.6. JEST

Segundo (Damasceno, Alexandre Braga) Estrutura de testes servem para testar como o próprio nome diz o seu sistema JavaScript, estas estruturas geram teste automáticos para programas que contêm campos de entrada, como alguns formulário que solicitam data de nascimento, nome e etc. Mas ele não trata nenhum dos dados inseridos. Estes testes ajudam o desenvolvedor a saber quando e como o seu programa ou aplicativo pode sofrer

uma sobrecarga de processamento e também ajudam a identificar o uso indevido das informações dos usuários, Existem diversas ferramentas de teste como o JSPrime, que tem como principal objetivo a detecção de código malicioso, Ele utiliza um conjunto que verifica o fluxo do código procurando sempre encontrar um caminho de informação sensível, Mas a desvantagem do JSPrime é que ele não executa verificação eval. Existem muitas outras estruturas de testes como Jasmine, Moncha, Puppeteer, AVA e o Jest. Segundo (Spirlandeli, Cleber, e Carlos Eduardo de França ROLAND) O framework Jest.Js é um projeto desenvolvido pelo Facebook ele constantemente recebe atualizações e possui uma documentação excelente, e hoje ele é um dos mais utilizado no mercado o Jest é ótimo quando se trata de validar a resposta da aplicação, ele também é muito bem utilizado é necessário validar as funcionalidades Listar (get) e Excluir (delete) Usuários, para fazer teste unitários com o Jest em sua máquina basta instalar o Jest seguindo a documentação oficial ele pode ser instalado usando o NPM ou o Yarn, Abaixo temos um exemplo de teste usando o framework Jest a primeira figura refere se ao código que será testado, Já a segunda o Jest notificará se o teste passou ou se falhou :

Figura 16 - Teste de verificação dentro de um código



```

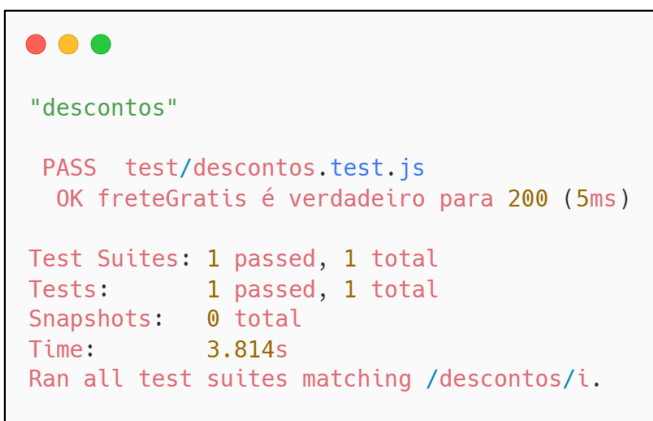
const freteGratis = require('./descontos').freteGratis()

test('freteGratis é verdadeiro para 200', () => {
  expect(freteGratis(200)).toBeTruthy()
})

```

Fonte: Autoria própria

Figura 17 - Resultado do teste no Jest



```

"descontos"

PASS test/descontos.test.js
  OK freteGratis é verdadeiro para 200 (5ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:  0 total
Time:        3.814s
Ran all test suites matching /descontos/i.

```

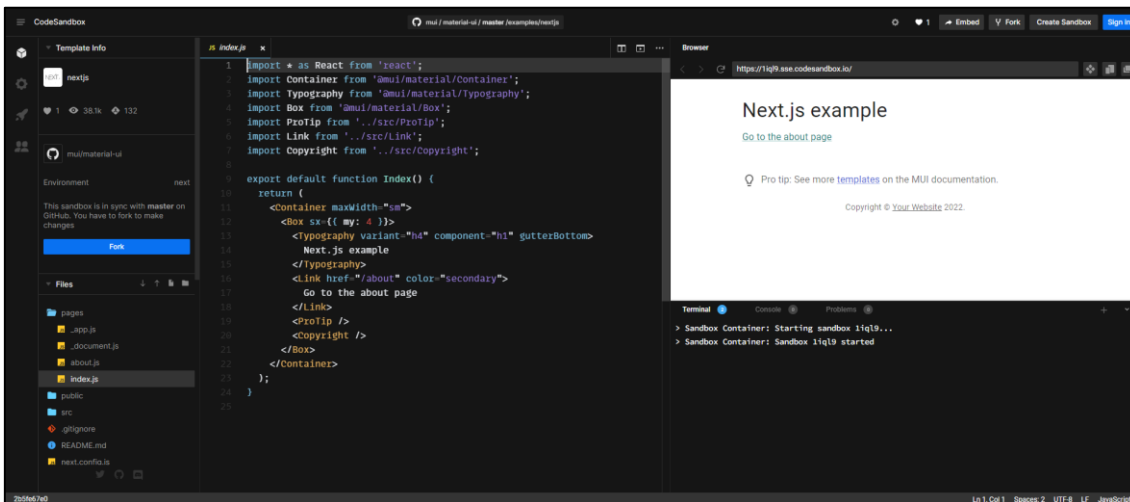
Fonte: <https://www.devmedia.com.br/teste-unitario-com-jest/41234>

Segundo (LEITE, MURILO AUGUSTO KRUGNER ALMEIDA.) Jest também permite o uso de diversas tecnologias como TypeScript, Node.js, React, Angular e Vue.js com ele não é preciso mais fazer testes manuais para capturar erros e bugs o que ajuda a economizar um bom tempo na hora de fazer os testes, Escolhemos o Jest para utilizarmos em nosso projeto, por conta da simplicidade de utilizarmos é claro mas também na eficiência e na segurança que ele possibilita ao desenvolvedor deixando o código muito mais seguro e eficaz.

2.7. MATERIAL UI

Segundo o documento oficial o MUI oferece um grupo abrangente de ferramentas de interface do usuário para ajudá-lo, componentes MUI funcionam de forma muito isolada, ele serve basicamente para facilitar o trabalho do desenvolvedor trazendo componentes prontos e alguns já estilizados, abaixo veja um exemplo do Material UI:

Figura 18 - Componentes MUI Renderizando



Fonte: Autoria própria

Segundo (Silva, Cleviane Rebeca Pinto Cruz, and Leonardo Oliveira Moreira). *Material-UI* é um componente do React, e ele também é baseado no Material Design, e o *Material-UI* foi feito apenas para ser utilizado com React, para isso é necessário que o desenvolvedor tenha conhecimentos básicos em React e siga o passo a passo dentro da documentação oficial do React, Uma das maiores vantagens de usar o *Material-UI* é que ele deixa seu site extremamente rápido além de otimizar o desempenho de seu site ou aplicativo, *Material-UI* é considerado uma das ferramentas *Front-end* mais populares do mundo, ajudando o desenvolvedor a tornar o seu aplicativo muito mais bonito, e deixando o site

com um design fluido e dinâmico, Existem diversos tipos de componentes dentro do *Material-UI* que podemos utilizar a biblioteca é enorme com uma grande variedade deixando assim o desenvolvedor com um leque muito mais abrangente de opções para aprimorar em seu site ou aplicativo, entre alguns deles estão os *cards*, *button*, *dialogs*, *grid* e etc. Todos com a funcionalidades de poupar o tempo do desenvolvedor de ter que criar todos estes componentes do início, fazendo assim o desenvolvedor ter um tempo mais curto na hora de criar os seus componentes e reduzindo a chances de erros ocorrer.

Segundo (DANTAS, Rafael Klynger da Silva.) *Material-UI* implementa o uso de design *Material Design*, Que é feito para facilitar o seu desenvolvimento do seu aplicativo ou site, Segundo(Nedel, Matheus Berkenbrock.) o *Material-UI* é uma linguagem visual que sintetiza os princípios clássicos do bom design com a inovação da tecnologia e ciência, Abaixo temos um exemplo de um formulário usando algum dos componentes do *Material-UI* como o *Input*, *button* e outros tipos de componentes:

Figura 19 - Exemplo de página criada apenas com *Material-UI*

HOME Register Login

Registration

First Name *

Last Name *


Address *

Email

Password

Gender:

Male Female Other

Date of Birth 

REGISTER

Fonte: <https://code.tutsplus.com/tutorials/how-to-build-a-login-and-registration-ui-with-angular-and-material-design--cms-31794>

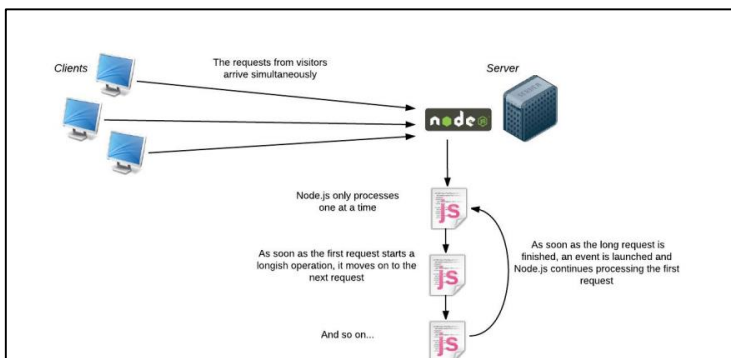
2.8. NODE.JS

Grande parte da internet baseia-se na arquitetura cliente-servidor, onde o servidor provê recursos e os clientes são os requerentes dos recursos ou serviços.

Conforme dito por Gabriel Stedile (2019, pg. 28) “o Node.js é um ambiente de execução JavaScript, que no contexto de modelo de aplicação cliente-servidor, se encontra do lado do servidor.”

O maior fator para o nodeJS ser uma ótima opção para escalabilidade e performance acima da média é a sua natureza orientada a eventos, ou seja, embora só possa manusear uma tarefa ao mesmo tempo, ele usa uma fila de eventos, fazendo com que as tarefas possam ser executadas sobre demanda.

Figura 20 - Funcionamento de uma aplicação NodeJS



Fonte: CODE, Visual Studio. Node. js. 2017

Por exemplo, cozinhar é uma tarefa assíncrona, por que não é necessário parar e esperar o arroz terminar de cozinhar para fazer outras coisas como a salada, feijão ou um frango assado, ou seja é possível realizar os passos necessários para um almoço sobre demanda.

O fato de o nodeJS suportar código não bloqueante e assíncrono o faz a plataforma ideal para aplicações que exigem comunicação em tempo real e processamentos de dados que tomem muito tempo.

2.9. NEXTJS

Segundo Almeida (2022, pg. 13) Framework é um termo em inglês que, em sua tradução direta, significa estrutura. De modo geral, essa estrutura é projetada para resolver um problema específico.

De modo geral ele visa resolver um determinado problema de forma mais ágil mantendo a segurança e a performance do sistema (DA SILVA, Lovato, 2006).

Por exemplo, em um cenário que envolva inteligência artificial e ciência de dados o framework Django pode ser mais adequado, ou em um cenário que envolva uma SPA com interfaces de usuário complexas e forte comunicação em tempo real, frameworks como Angular, Vue ou NextJS seriam a melhor ferramenta a se usar.

Segundo Cláudio Oliveira e Natan Fernandes (2021, pg. 10) em “Uma plataforma para coleta e análise de dados do GitHub”:

o framework Next.js busca reunir diversas funcionalidades como renderização híbrida e estática de conteúdo, suporte a TypeScript, pre-fetching, sistema de rotas, pacotes de funcionalidades e diversos plug-ins. O Next.js adiciona várias funcionalidades em cima do React, com uso do Next.js a interface web da ferramenta é adaptativa para dispositivos móveis.

2.10. ELECTRON

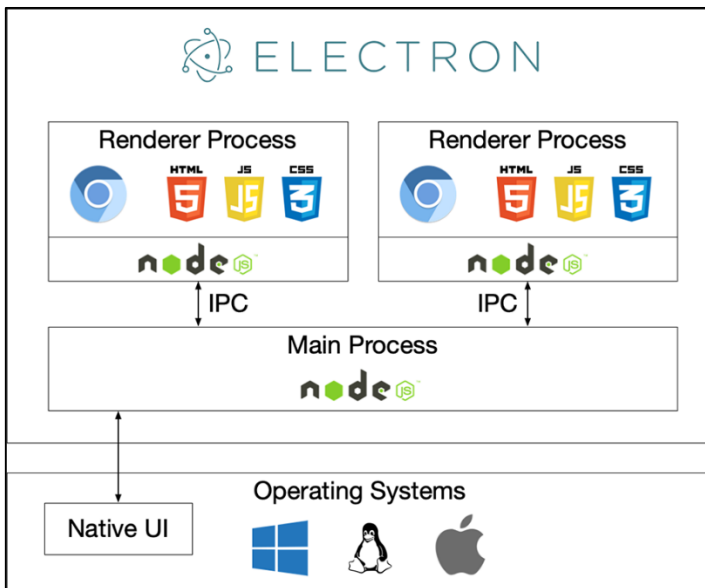
De acordo com Segundo DA SILVA CRUZ (2018, pg. 6) O electron é uma biblioteca que permite que com o uso de tecnologias web, como HTML, CSS e JS sejam criados aplicativos multiplataformas usando o NodeJS como sua base. Com o ele é possível acessar recursos nativos do sistema operacional, tornando disponível ainda mais recursos da máquina do que uma aplicação teria se estivesse executando pelo navegador.

Um exemplo de aplicativos muito famosos que usam o electron são o VSCode e o Discord, que funcionam em navegadores da web, mas que podem ser transformados em aplicativos nativos e ter suas funcionalidades ampliadas pelo fato de ter acesso nativo ao sistema.

O objetivo do Electron é usar tecnologias Web e ainda assim ter o poder e as funcionalidade que um aplicativo nativo teria.

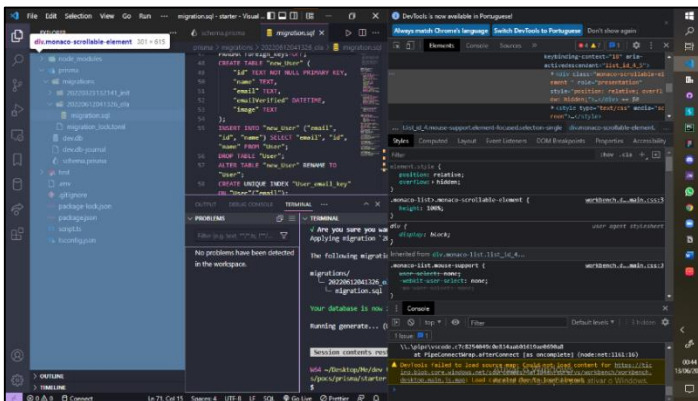
Como a figura a seguir, a arquitetura do Electron consiste em usar o NodeJS como ponte entre o sistema operacional e os processos IPC, que são de fato, a aplicação web. O funcionalidade principal do Electron consiste em trocar informações entre o sistema operacional e a aplicação Web por meio dos processos IPC.

Figura 21 - Arquitetura do Electron



Fonte: <https://insujang.github.io/2019-11-10/code-server/>

Figura 22 - DOM do VSCode visto pelo DevTools



Fonte: Autoria própria

2.11. BANCO DE DADOS

Um Banco de Dados (BD) é um sistema de manutenção de registro de acordo com Silva Filho (2002), ele funciona como um repositório para uma coleção de arquivos computadorizados, guardando informações sobre itens, usuários e outros e permitindo que esses dados sejam alterados e atualizados. Por exemplo um site de mídia social utiliza banco de dados pra guardar seus usuários e é isso o que te permite registrar ou realizar login no site e poder mudar seu nome sempre que desejar. o banco de dados também mante proteção dos seus dados contra usuários maliciosos, apenas quem tem acesso aos dados pode alterá-los. Os usuários do Banco de Dados podem fazer ações como:

- Adicionar novos arquivos
- Inserir dados a arquivos
- Editar arquivos
- Deletar arquivos
- Mover arquivos

Para melhor organização dos dados fornecidos os dados possuem metadados, diz Rob (2011, pg. 6) metadados fornecem uma descrição aos dados, armazenando informações como nome, valor(numérico, data, texto), não permitir que insiram dados vazios, entre outros, com isso nenhum arquivo fica com o mesmo valor ficando mais fácil de encontrar um arquivo específico quando necessário.

Os Banco de dados são utilizados para funções específicas utilizando aspectos do mundo real, por exemplo itens em um estoque ou pedidos de um cliente passam pelo BD para organização e administração desses dados. Elmasri (2005, pg. 3-4)

2.11.1. BANCO DE DADOS RELACIONAL

Um Banco de Dados Relacional (BDR) pode manter uma coleção de tabelas, com nomes únicos e podendo estar relacionadas a mais tabelas, caso um dado específico seja referenciado ele vai ter seu valor da tabela original presente, esse tipo de dado se chama chave estrangeira por vir de uma outra tabela, graças as chaves o arquivo pode se manter consistente sem ter valores diferentes de uma tabela para a outra. Boscarioli (2006, pg. 2)

Toda tabela em um BDR possui uma chave primaria, ela serve pra dar um valor completamente único para as tabelas, sendo um valor que nunca se repete nas outras tabelas, servindo pra identificar uma única linha da tabela. Machado (2020, pg. 26)

O BDR tem a capacidade de restringir integridade dos dados, para a proteção dos dados caso uma informação inserida não condiz com os dados relacionados essa informação, isso restringe a informação de ser inserida caso não cumpra o requisito. Macário (2005, pg. 5)

2.11.2. CASOS DE USO

Os casos de uso servem para definir as funcionalidades necessárias para o sistema, ele participa da fase de análise de requisito, em que procuramos o que é o nosso sistema e o que ele deve ser capaz de fazer, diz Barros (2009 pg. 2-3)

2.11.3. MODELAGEM DE DADOS

A modelagem de dados é um processo feito antes da criação de um Banco de Dados, para realizar a modelagem de dados de acordo com Cintra(2012, pg.18) utilizamos a ideia do Modelo Entidade Relacionamento (MER), identificando as entidades e determinando seus atributos e relacionamentos, resumindo pegamos um elemento real e separamos o que é relevante para o Banco de Dados.

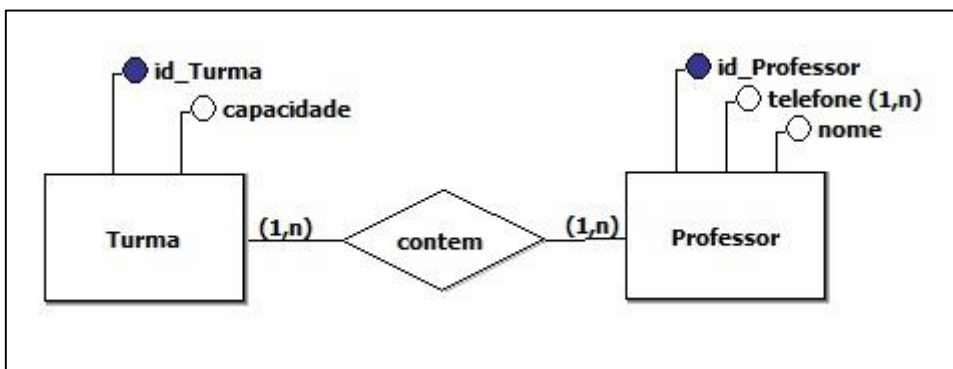
Entidades são objetos ou coisas do mundo real, uma pessoa é considerada uma Entidade, os atributos são os componentes dessa Entidade como o nome ou idade. Exemplo:

Entidade: turma e professor

Atributos: capacidade da turma, telefone do professor, nome do professor

Após fazermos a separação podemos fazer o modelo visual que seria o Diagrama Entidade Relacionamento (DER)

Figura 23 - Exemplo de DER



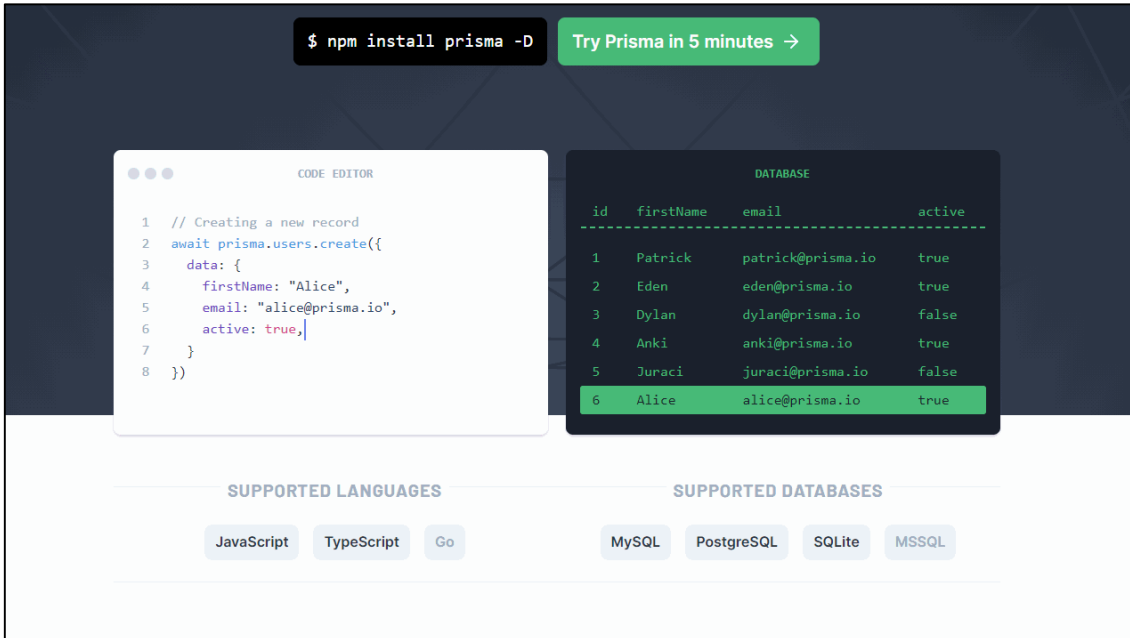
Fonte: <http://spaceprogrammer.com/bd/introducao-ao-modelo-de-dados-e-seus-niveis-de-abstracao/>

2.11.4. PRISMA

PRISMA: O Prisma é uma das melhores bibliotecas para a criação de um sistema CRUD, segundo (Medeiros, Miguel Pereira, et al.) Prisma é um mapeador relacional de objetivos, Ressalta (Borges, Gisele Cristina.) que o termo CRUD significa *create*, *read*, *update* and *delete* que são as quatro operações básicas que um banco de dados pode fazer, Segundo (Soares, Daniel Pontes Freitas, and Italo Borges Evangelista.) Prisma é

responsável pela comunicação com banco de dados e ajuda a fazer as operações básicas que os dados podem permitir, abaixo temos um exemplo do funcionamento básico do Prisma para interagindo com o banco de dados:

Figura 24 - Exemplo do funcionamento do Prisma com o banco de dados



Fonte: <https://prensa.li/prensa/como-criar-um-sistema-crud-com-o-prisma/>

2.11.5. SUPABASE

Supabase é uma ferramenta de comunicação em tempo real com o banco de dados, sendo uma alternativa ao Firebase para projetos com códigos abertos.

Embora semelhante ao Firebase, sua diferença é o serviço que oferece tanto o suporte para uma ferramenta, como o de criar a ferramenta caso ela não exista. Como de acordo com a documentação oficial do Supabase ele funciona como um Banco de Dados relacional.

2.11.6. FERRAMENTAS EM COMUM

PostgreSQL: Sistema Gerenciador de Banco de Dados relacional, serve para o armazenamento de informações.

(Milani, 2008)

Real time: Servidor Elixir permite inserir, atualizar e excluir dados de um banco de dados.

3. DESENVOLVIMENTO

3.1. PROBLEMAS NA COMUNIDADE DEV E SOLUÇÕES

O objetivo do DevsTogether é ser uma plataforma para desenvolvedores, com uma boa comunidade, e principalmente resolver pelo menos parcialmente os problemas de outras comunidades e ser algo prático. O DevsTogether tem inspiração em muitas plataformas, e uma delas é o StackOverflow, que é uma comunidade amplamente reconhecida e popular, mas como qualquer outra plataforma, existe defeitos e falhas, nosso objetivo é criar uma que também resolva esses problemas.

Na opinião de moderadores e usuários, os novos programadores estão indo contra a filosofia original do site, de fazer boas perguntas e querer aprender, e apenas se interessam por respostas e resultados. O usuário

(Relatos de usuários, analisar para ver se coloca mesmo)

Iremos utilizar a página de comunidade apenas para perguntas que querem respostas diretas, logo a pessoa que quer tirar a dúvida não será obrigada a entrar em uma chamada para solucionar o problema.

Outro erro frequente e que dificulta bastante, é quando alguém pede auxílio e manda uma captura da tela, em nossa plataforma o usuário será obrigado antes de fazer uma postagem escrever em campos o título, qual o problema a ser resolvido com código fonte e alguma mensagem do erro e o que foi tentado para resolver.

(Relatar matéria do StackOverflow)

Opiniões são saudáveis e podem ser de grande ajuda para o aprendizado, o problema é quando essas dicas não têm um embasamento técnico e podem acabar influenciando negativamente o estudo. Por exemplo perguntar sobre o mercado de trabalho ou qual a melhor linguagem de programação, mas porque não é adequado fazer perguntas como essa? Pois opiniões não tem uma resposta concreta e correta, afinal é algo de gosto pessoal.

Vamos criar uma página separada dedicada apenas para discussões da área de tecnologia, para interação e crescimento da comunidade

Uma boa comunidade é formada por pessoas que se respeitam e que sejam civilizados, onde mostrem equilíbrio, sensatez, cordialidade, amizade, humildade, honestidade e claro, estar disposto a ajudar e receber orientações também. Parece obvio, porém como em

qualquer plataforma existem pessoas ruins e desrespeitosas e cabe a própria comunidade e aos moderadores ter esse controle.

3.2. REGRAS DE NEGÓCIO

A plataforma terá um sistema de gamificação, com pontos de reputação, prêmios e reconhecimentos que darão incentivo, objetivo e foco aos desenvolvedores na comunidade. Contendo níveis de usuário: Principiante, Aluno, Dev Junior I, II e III, Dev pleno I, II e III e Dev Sênior I, II e III. Determinadas funções só serão possíveis se o usuário estiver no nível adequado (modificar essa parte para termos técnicos)

Além das regras de convivência já citadas anteriormente, teremos também algumas regras durante as chamadas como: Manter o respeito e não ofender a outra pessoa, não praticar atos de teor sexual ou imoral, não deixar a pessoa esperando sem um bom motivo e o conhecimento da pessoa e realizar qualquer ação criminosa. Todo usuário novo terá uma reputação inicial de acordo com os dados de sua conta GitHub.

RN01	Ao se cadastrar, a pessoa terá um dos 4 tipos de acesso, sendo eles admin, mod, members . por padrão seu acesso é de member até que alguém com cargo superior atribua um novo cargo
RN02	O usuário que iniciou a call deve aceitar a entrada da pessoa dentro da reunião, que enquanto não for aceita, ficará numa sala de espera
RN03	No final de uma sessão, a pessoa que ajudou poderá dar uma resposta escrita descrevendo a solução do problema. Essa resposta será anexada á respectiva pergunta que deu origem a chamada.
RN04	Em salas privadas não são atribuídas recompensas, e não há anexação de respostas. Caso um membro queira adicionar uma resposta á um problema não resolvido ainda na comunidade a pessoa pode criar uma pergunta e responder á própria pergunta.
RN05	Usuários com permissão igual ou acima de moderador poderão ter acesso a um painel onde haverá uma lista de pessoas com denúncias. Caso os motivos de denúncias sejam válidos ele terá o poder de bani-las temporariamente ou permanentemente ou remover acesso á recursos específicos na plataforma.

3.3. REQUISITOS NÃO FUNCIONAIS

RNF01	Em uma call, os usuários podem mandar mensagens no chat
RNF02	Durante uma call os usuário tem a opções de ativar/desativar câmera, microfone e som. (caso queira bloquear o áudio das outras pessoas na call para si)
RNF03	Perguntas públicas podem ser avaliadas com “good question” ou “bad question”. Caso 75% ou mais dos votos, contando a partir dos 5 primeiros votos, sejam negativos a pergunta é fechada e ficará indisponível á novas respostas. Isso garantirá que maus hábitos e perguntas ruins não permaneçam.
RNF04	Respostas á perguntas podem ser votadas com “good question” ou “bad question”. Também cada resposta estará aberta á comentários de outras pessoas.
RNF05	usuários podem fazer denúncias de outros usuários, perguntas, respostas ou comentários
RNF06	o usuário poderá ver e alterar as informações de seu perfil na página de configurações

3.4. REQUISITOS FUNCIONAIS

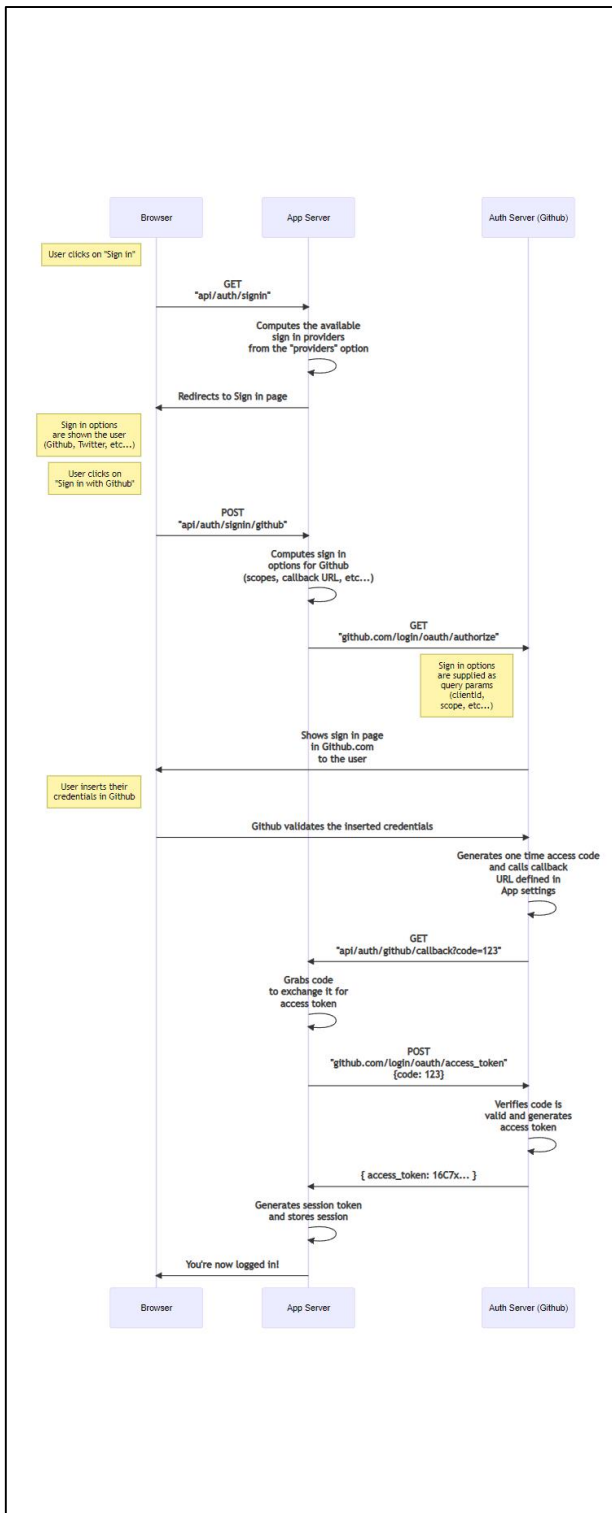
RF01	O sistema usará um banco de dados relacional postgres para armazenamento dos dados
RF02	Para comunicação por voz e vídeo, e chat será usado as tecnologias WebSocket e WebRTC

3.5. CASOS DE USO

Os casos de uso servem para definir as funcionalidades necessárias para o sistema, ele participa da fase de análise de requisito, em que procuramos o que é o nosso sistema e o que ele deve ser capaz de fazer, diz Barros (2009 pg. 2-3).

3.6. DIAGRAMAS DE FLUXO

Figura 26 - Diagrama de fluxo sobre OAuth

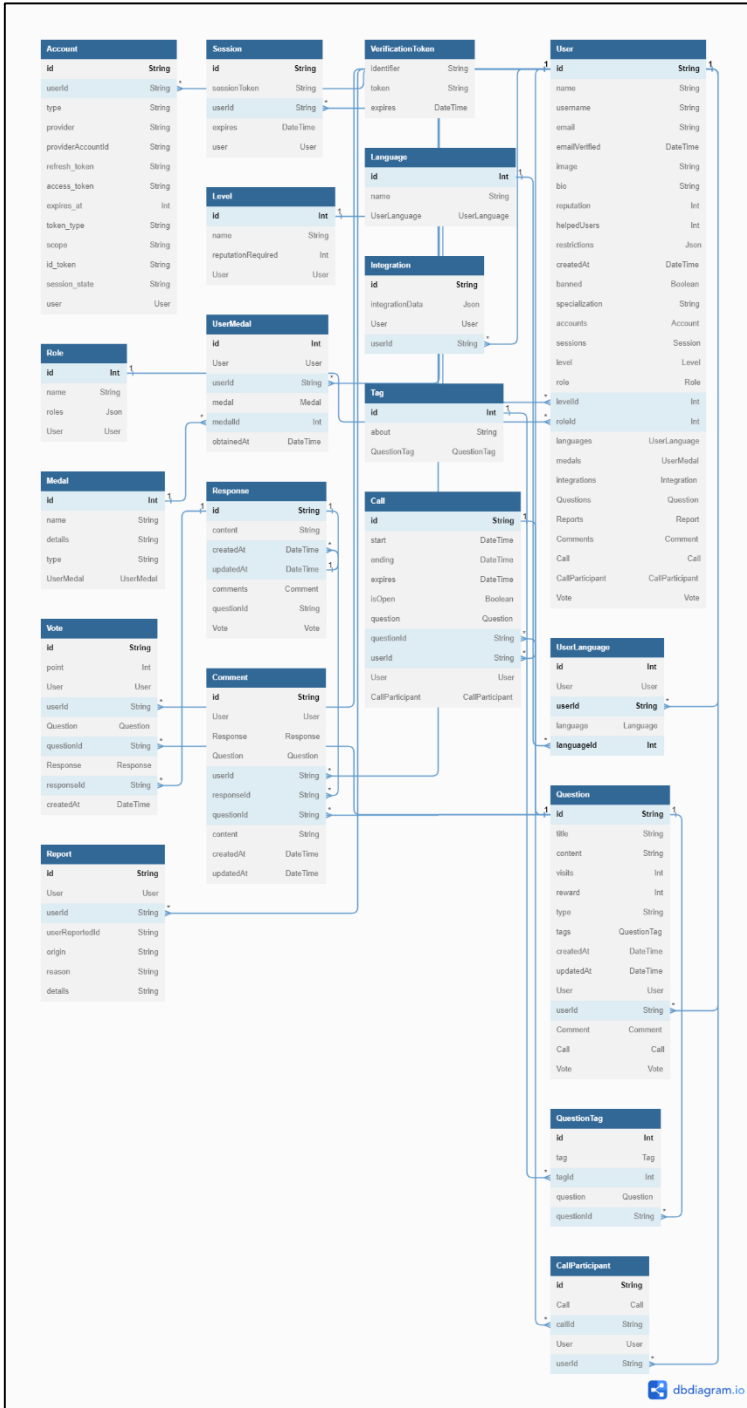


Fonte: <https://next-auth.js.org/>

Utilizando o seu navegador, o usuário entra em sua conta pelo GitHub logo após um processo de links até ser validado gerando tokens para enfim estar logado no site. Um processo feito por meio do Navegador, App server e Auto Server para garantir uma boa segurança nos projetos.

3.7. DER

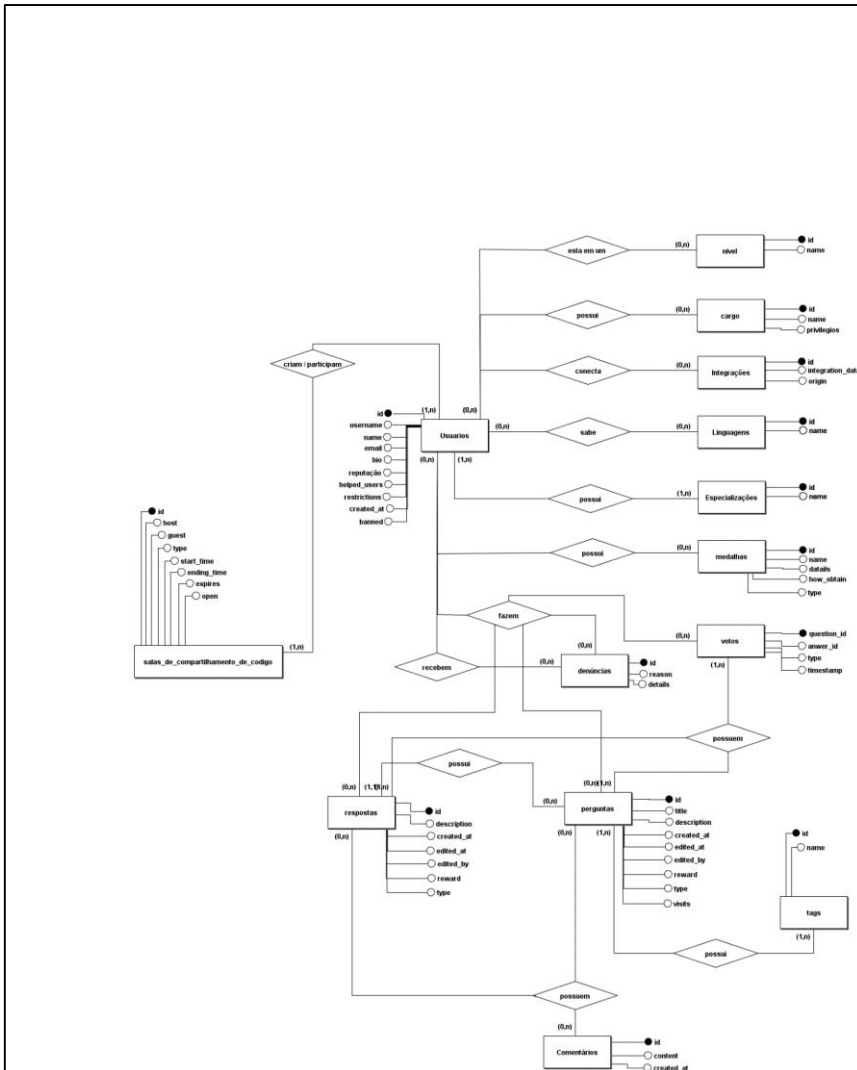
Figura 27 - DER do projeto



Fonte: Autoria Própria

3.8. MER

Figura 28 - MER do projeto



Fonte: Autoria Própria

4. REFERÊNCIAS

- SOLÓRZANO, Ana Luísa V.; CHARÃO, Andrea S. Plataforma Heroku In: SOLÓRZANO, Ana Luísa V. Explorando a Plataforma de Computação em Nuvem Heroku para Execução de Programas Paralelos com OpenMP. [S. l.: s. n.], 2017. cap. 2, p. 1-2.
- MILANI, André. O que é o PostgreSQL. In: MILANI, André. PostgreSQL GUIA DO PROGRAMADOR. [S. l.: s. n.], 2008. cap. 1.1, p. 25-26.
- SUPABASE. Como funciona. In: SUPABASE. Supabase. GitHub, 2022. Disponível em: <https://github.com/supabase/supabase/blob/master/i18n/README.pt.md>. Acesso em: 3 jun. 2022.
- MENDES, Antonio. Arquitetura de Software: desenvolvimento orientado para arquitetura. Editora Campus. Rio de Janeiro - RJ, 2002.
- CODE, Visual Studio. Node. js. 2017.
- DE SOUSA, Aislan Rafael Rodrigues; CARVALHO, Washington Henrique. Desenvolvendo Aplicações RESTFul utilizando Node. js.
- OLIVEIRA, Natan Fernandes Cláudio; SIRQUEIRA, Tassio Ferenzini Martins. Uma plataforma para coleta e análise de dados do GitHub. ANALECTA-Centro Universitário Academia, v. 7, n. 2, 2022.
- DA SILVA, Edson Coutinho; LOVATO, Leandro Alvarez. Framework Scrum: eficiência em projetos de software. Revista de Gestão e Projetos, v. 7, n. 2, p. 01-15, 2016
- ALMEIDA, Vinícius Souza. Plataforma de e-commerce integrada a microserviços. 2022.
- DA SILVA CRUZ, Vitor; PETRUCELLI, Erick Eduardo; SOTTO, Eder Carlos Salazar. A LINGUAGEM JAVASCRIPT COMO ALTERNATIVA PARA O DESENVOLVIMENTO DE APLICAÇÕES MULTIPLATAFORMA. Revista Interface Tecnológica, v. 15, n. 2, p. 39-49, 2018.
- STEDILE, Gabriel et al. Sistema web de suporte ao processo de voluntariado da Iniciativa Computação na Escola. 2020.
- META PLATFORMS. reactjs.org. Brasília: Meta Platforms, 2022. Disponível em: <https://pt-br.reactjs.org/docs/getting-started.html>. Acesso em: 31 mai. 2022.
- <https://jestjs.io/>. Disponível em: <https://jestjs.io/pt-BR/docs/getting-started>. Acesso em: 31 mai. 2022

MATERIAL UI SAS. <https://mui.com>. [S.I.]. Material UI SAS, 2022. Disponível em: <https://mui.com/pt/material-ui/getting-started/usage/>. Acesso em: 31 mai. 2022

LIMA, Natã Wereclys Bandeira. Protótipo de uma ferramenta gamificada para a aplicação de atividades práticas em sala de aula em uma disciplina de introdução à programação. 2022.

SILVA, Matheus Procópio da et al. Atingindo interfaces de usuários consistentes por composição e sistemas de tokens. 2019.

MANGIA, Ursula Barreto. Antecedentes à transição de carreira na área de tecnologia da informação. 2013. Tese de Doutorado.

FRANCISCO, Igor Moreira. PLATAFORMA WEB PARA AUXÍLIO DE TREINAMENTO DO USO DE PRÓTESES PARA INDIVÍDUOS COM AMPUTAÇÃO DE MEMBROS SUPERIORES. PLATAFORMA WEB PARA AUXÍLIO DE TREINAMENTO DO USO DE PRÓTESES PARA INDIVÍDUOS COM AMPUTAÇÃO DE MEMBROS SUPERIORES, [S. l.], p. 24, 1 dez. 2020. Disponível em: <https://repositorio.ufu.br/bitstream/123456789/31109/1/PlataformaWebPara.pdf>. Acesso em: 3 ago. 2022.

MARCELINO, Eduardo Rosalém; ROSATELLI, Marta Costa. Ensino de Programação em um ambiente colaborativo. SBC, p. 100, 2008.

STACKOVERFLOW (org.). 2021 Developer Survey. 2021. Disponível em: <https://insights.stackoverflow.com/survey/2021>. Acesso em: 09 mar. 2022

TYPESCRIPT for the New Programmer. [S. l.], 15 jun. 2022. Disponível em: <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>. Acesso em: 15 jun. 2022.

Lins, Gabriel de Souza. "Utilizando ReactJS para o desenvolvimento de um sistema de alocação e reserva de salas no campus da UFC em Quixadá." (2019).

de Camargos, J. G. C., Coelho, J. F., Villela, H. F., & Aramuni, J. P. (2019). Uma Análise Comparativa entre os Frameworks Javascript Angular e React. *Computação & Sociedade*, 1(1).

Almeida, Pedro H. Marques, et al. "Análise Comparativa das Características de Performance dos JavaScript Frameworks React e Vue."

Flatschart, Fábio. *HTML 5-Embarque Imediato*. Brasport, 2011.

MANZANO, JOSÉ AUGUSTO NG, and SUELY ALVES DE TOLEDO. Guia de orientação e desenvolvimento de sites: HTML, XHTML, CSS E JAVASCRIPT/JSCRIPT. Saraiva Educação SA, 2010.

Silveira, Paulo, and Adriano Almeida. Lógica de programação: Crie seus primeiros programas usando JavaScript e HTML. Editora Casa do Código, 2014.

Jobstraibizer, Flávia. *Criação de sites com o CSS*. Universo dos Livros Editora, 2009.

Quierelli, Davi Antonio. *Programação Para Internet*. Clube de Autores, 2013.

OLIVEIRA, Andressa Cruz de. "Desenvolvimento de material didático para ensino de estática com HTML5, CSS e Javascript." (2017).

Flanagan, David. *JavaScript: o guia definitivo*. Bookman Editora, 2004.

Prescott, Preston. *Programação em JavaScript*. Babelcube Inc., 2016.

Grillo, Filipe Del Nero, and Renata Pontin de Mattos Fortes. "Aprendendo JavaScript." *São Carlos: USP* (2008).

de Oliveira Lopes, Jean. "PHP ou TypeScript: uma comparação de duas linguagens para web pelas suas características."

Silva, Afonso Dias de Oliveira Conceição. "Correções automáticas de problemas em TypeScript e JavaScript via Pull Requests." (2019).

MERLIN, João Paulo. Desenvolvimento de uma ferramenta de scaffolding para criação de código fonte para front-end. 2018. 53 f. Trabalho de Conclusão de Curso (Graduação) - Universidade Tecnológica Federal do Paraná, Pato Branco, 2018.

Anjos, Luiz Felipe Rosa dos. "Evolução do Javascript em aplicações multiplataforma." (2017).

Damasceno, Alexandre Braga. "TaintJSec: um método de análise estática de marcação em código Javascript para detecção de vazamento de dados sensíveis." (2017).

Spirlandeli, Cleber, and Carlos Eduardo de França ROLAND. "A utilização de testes automatizados no desenvolvimento de software." (2019).

LEITE, MURILO AUGUSTO KRUGNER ALMEIDA. “DESENVOLVIMENTO DE UM APLICATIVO MÓVEL PARA PRESTAÇÃO DE SERVIÇOS UTILIZANDO A PLATAFORMA ANDROID.” (2021).

Silva, Cleviane Rebeca Pinto Cruz, and Leonardo Oliveira Moreira. “Um estudo de frameworks de design responsivo e avaliação na perspectiva da acessibilidade na web.” (2019).

DANTAS, Rafael Klynger da Silva. “Gerador de código base para aplicações back-end que usam MVC.” (2019).

Nedel, Matheus Berkenbrock. “Sistema integrado para atendimento em restaurantes.” Sistemas de Informação-Florianópolis (2020).

Borges, Gisele Cristina. Organização em redes e plataformas digitais: uma contribuição para os gestores de núcleos de inovação tecnológica. Editora Dialética, 2022.

Medeiros, Miguel Pereira, et al. “TICKET: AVALIAÇÃO DO PROCESSO DE CRIAÇÃO DE UM SISTEMA DE GERENCIAMENTO DE CHAMADOS.” Pensar Acadêmico 19.4 (2021): 1223-1243.

Soares, Daniel Pontes Freitas, and Italo Evangelista. “Sistema para gerenciamento e controle interno de micro e pequenas empresas: aplicado em uma empresa do ramo de vidraçaria.” (2021).

Oliveira, Jackson Vinicius Faria de. Biblioteca colaborativa Booker. BS thesis. Universidade Tecnológica Federal do Paraná, 2021.

Santiago, Gabriel Mendes de Souza. “Severino: uma aplicação web para encontrar profissionais.” (2021).

SILVA, Matheus Procópio da. “Atingindo interfaces de usuários consistentes por composição e sistemas de tokens.” (2019).