

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA
SOUZA**

Etec da Zona Leste

Novotec Desenvolvimento de Sistemas

Desirée Constantino de Almeida Barboza

Giovana Marsigli Rodrigues

Isabelle Gomes de Souza Andrade

PILLTRACK: IoT para gerenciamento de medicamentos

São Paulo

2024

Desirée Constantino de Almeida Barboza

Giovana Marsigli Rodrigues

Isabelle Gomes de Souza Andrade

PILLTRACK: IoT para gerenciamento de medicamentos

Trabalho de conclusão de curso apresentado ao Curso Técnico em Desenvolvimento de Sistemas da ETEC Zona Leste, orientado pelo Prof. Esp. Jeferson Roberto de Lima, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

São Paulo

2024

AGRADECIMENTOS

A conclusão deste Trabalho de Conclusão de Curso (TCC) é um grande marco em nossas vidas profissionais e acadêmicas, e é de extrema importância destacar que este projeto não é apenas um marco pessoal, mas também uma conquista compartilhada com aqueles que nos apoiaram desde o início.

Expressamos nossa profunda gratidão a Deus, por nos dar saúde e sabedoria necessárias para superar cada desafio ao longo da nossa trajetória.

Agradecemos de todo o coração às nossas famílias, pelo apoio constante e por sempre acreditarem em nosso potencial, sua paciência e encorajamento foram essenciais para chegarmos até aqui.

Aos nossos coordenadores, Jeferson Roberto de Lima e Rogério Bezerra, expressamos agradecimento pela orientação e dedicação.

Por fim, agradecemos a todos que, de alguma forma, contribuíram para a realização deste trabalho. Cada gesto e palavra de apoio foram fundamentais para alcançarmos nosso objetivo.

Muito obrigada a todos!

“Nunca se satisfaça com o que você alcançou, porque o que você faz hoje vai determinar o que você alcançará amanhã.”

William Pollard

RESUMO

O projeto aborda a criação de um sistema completo e eficiente de gestão de medicamentos, visando assegurar a adesão a tratamentos e melhora de resultados na saúde dos pacientes, especialmente os idosos. O sistema incluirá uma caixa de medicamentos inteligente para notificações de horários de administração e controle de estoque, além de um aplicativo móvel para registro e acesso às informações sobre a rotina medicamentosa dos pacientes por médicos e cuidadores. A metodologia envolverá implementação e teste do sistema em ambiente controlado, seguido de avaliação qualitativa da experiência dos usuários e análise quantitativa dos dados de adesão ao tratamento e resultados de saúde. Espera-se que o sistema melhore significativamente a qualidade de vida dos usuários ao potencializar os resultados de saúde, com benefícios como melhor adesão ao tratamento medicamentoso, redução de erros na administração de medicamentos e melhoria dos indicadores de saúde dos pacientes. Em conclusão do estudo destacará a eficácia da abordagem tecnológica adotada na gestão de medicamentos e seus benefícios para a saúde pública, sublinhando a importância de soluções inovadoras no cuidado com pacientes crônicos, especialmente os idosos.

Palavras-Chave: Gestão, Medicamentos, IoT, Aplicativo, Tratamento, Monitoramento.

ABSTRACT

The project addresses the creation of a complete and efficient medication management system, aimed at ensuring adherence to treatment and improving the health outcomes of patients, especially the elderly. The system will include a smart medicine box for notifications of administration times and stock control, as well as a mobile app for recording and accessing information on patients' medication routine by doctors and caregivers. The methodology will involve implementing and testing the system in a controlled environment, followed by qualitative evaluation of the user experience and quantitative analysis of treatment adherence data and health outcomes. The system is expected to significantly improve users' quality of life by boosting health outcomes, with benefits such as better adherence to drug treatment, a reduction in medication administration errors and an improvement in patients' health indicators. In conclusion, the study will highlight the effectiveness of the technological approach adopted in medicines management and its benefits for public health, underlining the importance of innovative solutions in the care of chronic patients, especially the elderly.

Keywords: Management, Medicines, IoT, Application, Treatment, Monitoring.

LISTA ILUSTRAÇÕES

Figura 1 – Exemplo de estrutura básica de HTML	16
Figura 2 – Exemplo de estrutura HTML para um formulário.....	18
Figura 3 – Exemplo de estrutura HTML para um formulário projetado na <i>web</i>	19
Figura 4 – Exemplo de link CSS para ligação a página HTML.....	20
Figura 5 – Código CSS para estilização de botão HTML	20
Figura 6 – Código CSS para estilização de formulário HTML	22
Figura 7 – Resultado do formulário HTML estilizado em CSS	23
Figura 8 – Elemento “onsubmit” no HTML.....	23
Figura 9 – Exemplo de código JavaScript	24
Figura 10 – Resultado da programação JavaScript	25
Figura 11 – Exemplo de diagrama de Caso de Uso	26
Figura 12 – Exemplo de diagrama de Sequência.....	28
Figura 13 – Exemplo de diagrama de Atividade	28
Figura 14 – Exemplo de diagrama de Máquina de Estados	29
Figura 15 – Exemplo de wireframe de baixa fidelidade	30
Figura 16 – Exemplo de wireframe de alta fidelidade.....	31
Figura 17 – Tela de criação de projetos do Figma	32
Figura 18 – Logotipo Firebase.....	32
Figura 19 – Console Firebase	33
Figura 20 – Cloud Firestore.....	34
Figura 21 – Tela de apresentação do site Tailwind CSS.....	35
Figura 22 – Terminal exibindo a criação de um novo projeto	35
Figura 23 – Visualização da pasta do projeto recém-criado.....	36
Figura 24 – Terminal mostrando a instalação do Tailwind	36
Figura 25 – Terminal mostrando a criação do arquivo tailwind.config.js	36
Figura 26 – Arquivo CSS com a importação do Tailwind	37
Figura 27 – Configuração dos caminhos no arquivo tailwind.config.js	37
Figura 28 – Modelo de formulário usando Tailwind.....	37
Figura 29 – Código do formulário usando Tailwind	38
Figura 30 – Tela de apresentação do site React Native.....	39
Figura 31 – Comando de criação projeto React Native	40
Figura 32 – Exemplo de código React Native	41
Figura 33 – Resultado da codificação React Native	42
Figura 34 – Exemplo de Modelagem 3D	43
Figura 35 – Exemplo de prototipagem 3D impressa e pintada.....	44
Figura 36 – ESP 32	45
Figura 37 – Exemplo de conexão e comunicação do LED RGB Endereçável	45
Figura 38 – Fita LED RGB endereçável.....	46
Figura 39 – Display LCD	47
Figura 40 – Módulo RTC DS3231 (Real Time Clock).....	48
Figura 41 – Buzzer Passivo.....	49
Figura 42 – Sensor Óptico TCRT5000	49

Figura 43 – Exemplo C++.....	50
------------------------------	----

LISTA DE TABELAS

Tabela 1 - Exemplo Documentação de Caso de Uso abertura de uma conta.....	27
---	----

LISTA DE ABREVIATURAS E SIGLAS

Cascading Style Sheet (CSS).

Duas dimensões (2D).

Eletronic Stability Program (ESP).

Ground (GND).

HyperText Markup Language (HTML).

Inter-Intregrated Circuit (I2C).

Interface do usuário (IU).

Integrated Development Environment (IDE).

Internet of Things (IoT).

JavaScript (JS).

Kilobyte (KB).

Light Emitting Diode (LED).

Liquid Cristal Display (LCD).

Milliampere (mA).

Model View Controller (MVC).

Node Package Maneger (NPM).

Not Only SQL (NoSQL).

Pulse Position Modulation (PPM).

Pulse-Width Modulation (PWM).

Real Time Clock (RTC).

Red, Green, Blue (RGB).

Sistema de Gerenciamento de Banco de Dados (SGBD).

Static Random-Acess Memory (SRAM).

Structured Query Language (SQL).

Television (TV).

Tridimensional (3D).

Unified Modeling Language (UML).

Volt (V).

Voltage Common Collector (VCC).

Wireless Fidelity (Wi-Fi).

SUMÁRIO

1.	INTRODUÇÃO	13
2.	REFERENCIAL TEÓRICO	15
2.1	Dificuldades na eficácia da gestão de medicamentos	15
2.2	HTML.....	15
2.3	CSS	19
2.4	JavaScript	23
2.5	UML.....	25
2.5.1	UML - Diagrama de Casos de Uso.....	26
2.5.2	UML - Documentação Caso de Uso	26
2.5.3	UML - Diagrama de Sequência	27
2.5.4	UML - Diagrama de Atividade	28
2.5.5	UML - Diagrama de Máquina de Estados.....	29
2.6	<i>Wireframe</i>	29
2.7	Figma	31
2.8	Banco de dados.....	32
2.8	Firebase.....	32
2.9	Firestore	33
2.10	Node.js & NPM	34
2.11	Tailwind	34
2.11.1	Instalação do Tailwind via NPM (Node Package Manager)	35
2.12	React.....	39
2.13	React Native	39
2.13.1	Criação de um aplicativo em React Native.....	40
2.14	Express.js.....	42
2.15	IoT	42
2.15.1	Modelagem 3D	43
2.15.2	ESP 32	44
2.15.3	Fita LED RGB endereçável	45
2.15.6	Display LCD	46
2.15.5	Modulo RTC DS3231 (Real Time Clock).....	47
2.15.6	Buzzer Passivo	48

2.15.7	Sensor Óptico TCRT5000.....	49
2.15.8	C++.....	50
REFERÊNCIAS.....		51

1. INTRODUÇÃO

Segundo a Eurofarma (2018), é crucial seguir a prescrição médica nos horários corretos, independentemente da idade. O médico adapta o tratamento com base no corpo individual, considerando seus hábitos e rotinas. Em consideração a isso, este projeto foca na seriedade da gestão de medicamentos, assegurando o controle e uso correto, evitando esquecimentos e problemas decorrentes da má disciplina dos pacientes. A solução tecnológica proposta visa organizar e monitorar o tratamento adequado de cada medicamento para o usuário.

Assim, a tecnologia evolui para atender às necessidades da sociedade, acompanhando sua evolução. Isso nos leva ao surgimento da Internet das Coisas (IoT) com grande potencial. (Albertin, 2017). Hoje a Internet das Coisas, de forma rápida, se torna um cenário da realidade, pois ao olhar ao redor podemos ver que nossos dispositivos ficam mais inteligentes a cada dia. (SANTOS, 2018).

Com essa tecnologia, é possível automatizar tarefas demoradas ou esquecidas. Na área da saúde, um sistema IoT se torna uma ferramenta valiosa para pacientes e profissionais, melhorando a compreensão e gerenciamento de atividades. Aliada ao conhecimento médico, essa tecnologia pode atuar na medicina preventiva, melhorando a qualidade de vida e a satisfação dos pacientes. (Massola; Pinto, 2018). A adesão correta aos medicamentos prescritos faz grande diferença na recuperação e evita o agravamento de condições que requerem tratamento.

Conforme dados referenciados pela Sociedade de Cardiologia do Estado de São Paulo. (SOCESP, 2020) “estima-se que metade dos 3,2 bilhões de prescrições médicas realizadas anualmente nos EUA não são seguidas corretamente”. Sendo assim, um dos principais impactos positivos desse sistema é melhorar a adesão medicamentosa. Com o acesso fácil às informações dos medicamentos e notificações automáticas, os usuários são incentivados a seguir as orientações médicas, reduzindo o esquecimento e aumentando a eficácia dos tratamentos a longo prazo.

Outro impacto positivo é o acesso ao sistema, permitindo uma análise mais específica de responsáveis e médicos sobre a adesão ao tratamento pelos pacientes.

A aplicabilidade ocorre através de um sistema embarcado interligado a uma plataforma de aplicativo com a integração de alerta de ambos os sistemas, a parte que o embarcado emitirá um sinal sonoro e acenderá a luz do compartimento do remédio a ser tomado. Além disso, o sistema monitorará o estoque, dará relatórios detalhados e históricos de ingestão, disponibilizando também espaço de relatos a sintomas adversos da gestão dos medicamentos.

No caso de pacientes idosos, que são mais propensos a esquecer doses e horários, a criação da aplicação se torna ainda mais positiva, visto que tende a solucionar esses problemas, contribuindo para a melhoria na qualidade da assistência, assegurando a administração correta dos medicamentos, tornando assim o projeto único e inovador com uma automação na problemática abordada.

2. REFERENCIAL TEÓRICO

Neste capítulo será abordado o embasamento teórico e as tecnologias usadas, assim como seus conceitos e aplicabilidades para construção do nosso projeto de gerenciamento de medicamentos PillTrack.

2.1 Dificuldades na eficácia da gestão de medicamentos

Um dos princípios fundamentais para uma boa recuperação e tratamento é a comunicação direta entre pacientes e médicos, permitindo que juntos cheguem aos melhores resultados no combate às doenças. Pouco tempo de consulta e má comunicação aumentam a propensão ao distanciamento e à hiper formalidade no diálogo, o que contribui para esses problemas. (Pixel Diagnóstico, 2020). Ou seja, um histórico médico detalhado pode funcionar como um mapa para prevenir futuros problemas de saúde. (Unimed Campinas).

Dessa forma, é possível utilizar recursos que facilitem esse acompanhamento. Para ter mais tempo de escutar e acompanhar seus pacientes, os médicos podem contar com tecnologias de *software* para gerenciamento das informações dos pacientes em um só local.

Uma pesquisa realizada por alunos da Universidade de São Paulo (USP) com profissionais da saúde destacou algumas ações e reações ao problema da não adesão ao tratamento adequado. Entre as ações, a avaliação do cumprimento do tratamento inclui medidas para controlar a adesão do paciente, utilizando determinadas tecnologias de controle dentro do contexto institucional. As reações mais citadas pelos profissionais incluíram termos como “gravidade”, “piora”, “preocupação”, “prejudicial” e “morte”, refletindo os efeitos da má adesão ao tratamento tanto para os pacientes quanto para os profissionais. (FERREIRA; CAMPOS, 2023).

2.2 HTML

De acordo com Maurício Samy Silva (2008), HTML é a sigla para *HyperText Markup Language*, que, em português significa linguagem para marcação de hipertexto.

Os elementos HTML ou chamados de *tags* HTML, são utilizados para informar ao navegador que tipo de estrutura está sendo construída, podendo ser títulos,

parágrafos, imagens, *links*, entre outros. (Alura, 2018). Segue abaixo um exemplo de estrutura básica:

Figura 1 – Exemplo de estrutura básica de HTML



```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Exemplo</title>
6  </head>
7  <body>
8
9  </body>
10 </html>
```

Fonte: Do próprio autor, 2024.

Para melhor compreensão, explicaremos as *tags* referente a imagem acima:

- *Doctype*: “<!DOCTYPE html>” essa *tag* informa ao navegador qual é o tipo de documento e a versão em HTML.
- *Html*: “<html lang = “pt-br”></html>” a *tag* essencial para a construção do código em HTML, serve como base e engloba todos os elementos na construção de documentos. Com o LANG, é possível indicar o idioma do texto representado.
- *Head*: “<head></head>” a *tag HEAD* é onde fica toda a parte inteligente da página, contendo as informações sobre o documento e o conteúdo ali publicado.
- *Metatag Charset*: “<meta charset = “UTF – 8”>” *tag* responsável por chavear qual tabela de caracteres a página está utilizando.
- *Title*: “<title></title>” *tag* usada para dar referência ao título da página que será inserido nas guias do navegador.

- *Body*: “<body></body>” a *tag* responsável pela inserção de todo o corpo do código, estabelecendo o conteúdo visual da página, no qual o usuário vê e interage.

Além da estrutura básica do HTML, para a construção de uma página é necessário mais do que as *tags* principais como:

- *Div*: “<div></div>”, define uma divisão da página, funciona como um container para conteúdo fluxo.
- *Br*:
, não necessita de fechamento, ela executa a função de quebra de linha.
- H1, H2, H3, H4, H5, H6: “<h1>”, “<h2>”, “<h3>”, “<h4>”, “<h5>”, “<h6>”, possuem valor semântico, variando entre seis níveis hierárquicos, a página contendo apenas H1 não seria apenas para o título principal, H2 e H3 para títulos de seções e a H4, H5 e a H6 para subtítulos.
- *P*: “<p>”, principal *tag* usada para composição de um parágrafo.
- *Span*: “”, costumam ser utilizadas apenas para pequenas informações, como legendas de um formulário, legendas de uma imagem, entre outros.
- *B*: “”, transforma o conteúdo em negrito.
- *I*: “<i></i>”, transforma o conteúdo em itálico.
- *Hr*: “<hr>”, não necessita de fechamento, ela forma uma linha horizontal.
- *Button*: “<button>”, utilizada para inserir interatividade em uma página ou formulário e tem como função executar uma determinada ação ao receber um clique do usuário.
- *Form*: “<form></form>”, usada para criar campos de entrada interativos, nos quais as pessoas usuárias podem inserir dados.
- *Footer*: “<footer></footer>”, define um rodapé para a página, geralmente são utilizados no final da página.
- *Section*: “<section></section>”, define uma sessão para sua página.
- *Nav*: “<nav></nav>”, define um conteúdo de navegação, portanto é muito utilizado em criação de menus.
- *Header*: “<header></header>”
- *Main*: “<main></main>”, representa o conteúdo principal do seu corpo, ou seja, o conteúdo relacionado diretamente com o tópico central da página.

- *Aside*: “<aside></aside>”, representa uma seção de uma página cujo conteúdo e relacionado ao seu entorno, que pode ser considerado separado do conteúdo.
- *Img*: “”, usada para incluir uma imagem ao seu texto.
- *Video*: “<video>”, usada para indicar inserção de vídeo.
- *Input*: “<input>”, possui o atributo *type*, que varia entre diversos tipos, como “<input type = “text”>”, que é um campo que receberá qualquer caractere, já o “<input type= “email”>”, define um campo que receberá caracteres e verificará se o mesmo consiste em um e-mail válido.

Abaixo temos um exemplo em HTML que provirá de um formulário simples para melhor compreensão de algumas *tags melhor* explicadas acima:

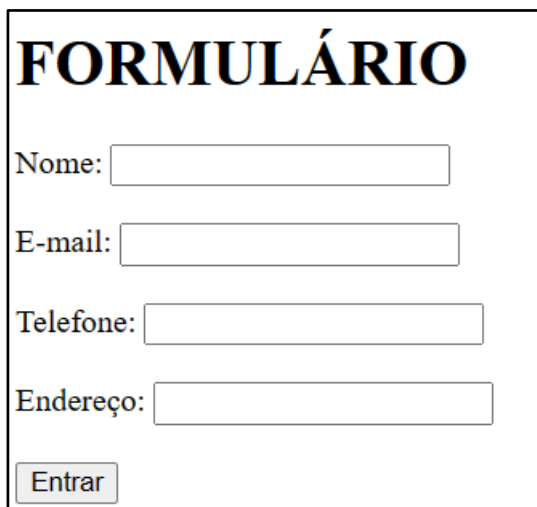
Figura 2 – Exemplo de estrutura HTML para um formulário

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <link rel="stylesheet" href="Css.css">
6      <title>Exemplo Monografia</title>
7  </head>
8  <body>
9      <div>
10         <h1>FORMULÁRIO</h1>
11         <label for="usuario">Nome:</label>
12         <input type="text" id="usuario" name="usuario" required><br><br>
13
14         <label for="email">E-mail:</label>
15         <input type="email" id="email" name="email" required><br><br>
16
17         <label for="telefone">Telefone:</label>
18         <input type="text" id="telefone" name="telefone" required><br><br>
19
20         <label for="telefone">Endereço:</label>
21         <input type="text" id="Endereço" name="Endereço" required><br><br>
22
23         <button type="submit">Entrar</button>
24     </div>
25 </body>
26 </html>
```

Fonte: Do próprio autor, 2024.

Temos agora reprodução gráfica do exemplo HTML projetado acima:

Figura 3 – Exemplo de estrutura HTML para um formulário projetado na *web*



FORMULÁRIO

Nome:

E-mail:

Telefone:

Endereço:

Fonte: Do próprio autor, 2024.

2.3 CSS

Abreviação para o termo *Cascading Style Sheet* que em português possui a tradução como folha de estilo em cascata, desta forma cria-se assim a sigla CSS. Surgiu em 1994 como uma proposta de Tim Berners-Lee que havia desenvolvido um navegador com um conjunto de funcionalidades de estilização padrão, sem definição de estilo feita pelo usuário ou autor. (j, 2011). Criado exclusivamente para uma linguagem de marcação e estruturação, o CSS faz o que HTML não é capaz de proporcionar, como todo o visual de um documento, cabe a essa linguagem a função de estilização do mesmo. (SILVA, 2018).

Possui um funcionamento prático, a formatação feita de seus estilos se dá por fora do documento original HTML, sendo interligado por meio de *link* criado pelo desenvolvedor, que se resulta em uma página formatada conforme a definição feita pelo arquivo CSS. (JOBSTRAIBIZER, [s.d.]).

Figura 4 – Exemplo de *link* CSS para ligação a página HTML



```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <link rel="stylesheet" href="Css.css">
6      <title>Exemplo Monografia</title>

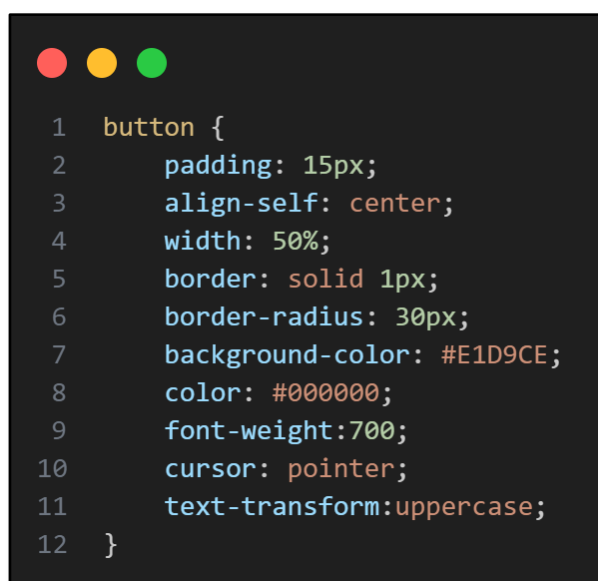
```

Fonte: Do próprio autor, 2024.

Como exemplo a ser explicado iremos usar a estilização de um botão que passara por seletores e propriedades, conforme dito por FERREIRA (2012), os seletores são estruturas para determinar quais elementos serão formatados e as propriedades são características desses elementos que pretende formatar.

Assim, conforme declarado por MAZZA (2014), existem seletores e propriedades básicas para a estilização de um botão como apresentado abaixo:

Figura 5 – Código CSS para estilização de botão HTML



```

1  button {
2      padding: 15px;
3      align-self: center;
4      width: 50%;
5      border: solid 1px;
6      border-radius: 30px;
7      background-color: #E1D9CE;
8      color: #000000;
9      font-weight: 700;
10     cursor: pointer;
11     text-transform: uppercase;
12 }

```

Fonte: Do próprio autor, 2024.

- Seletor: O seletor é *button*, este seletor aplica essas regras a todos os botões no documento HTML a partir das propriedades declaradas.

- *padding*: Cria espaço dentro do botão entre o conteúdo e as bordas do botão.
- *align-self*: Define o alinhamento do botão.
- *width*: Define a largura do botão.
- *border*: Remove a borda do botão. Isso significa que o botão não terá uma borda visível.
- *border-radius*: Define o raio de curvatura das bordas. Isso cria cantos arredondados para o botão.
- *background-color*: Define a cor de fundo do botão.
- *color*: Define a cor do texto do botão.
- *font-weight*: Usada para definição da espessura da fonte
- *cursor*: Define o cursor do mouse ao passar sobre o botão como um ícone de ponteiro, indicando que o botão é clicável.
- *Text-transform*: Usado para especificar como o texto deve ser transformado em relação à sua capitalização e caracteres.

Logo, existem algumas maneiras de se criar uma página formatada em CSS com elementos básicos, sendo possível criar estilizações mais significativas a um documento HTML, como podemos ver na figura abaixo:

Figura 6 – Código CSS para estilização de formulário HTML

```

body {
  font-family: Arial, sans-serif;
  background-color: #F0F2EE;
  display: flex;
  height: 100vh;
  justify-content: center;
  align-items: center;
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

form {
  width: 400px;
  background-color: white;
  padding: 50px;
  border-radius: 30px;
  display: flex;
  flex-direction: column;
}

input[type="text"],
input[type="email"],
input[type="password"],

textarea {
  width: 100%;
  padding: 10px;
  margin-bottom: 15px;
  border: 3px solid #E1D9CE;
  border-radius: 30px;
  box-sizing: border-box;
  justify-content: center;
  flex-direction: column;
}

input[type="submit"] {
  width: 100%;
  padding: 10px;
  border: none;
  border-radius: 15px;
  background-color: #E1D9CE;
  color: #fff;
  cursor: pointer;
}

button {
  padding: 15px;
  align-self: center;
  width: 50%;
  border: solid 1px;
  border-radius: 30px;
  background-color: #E1D9CE;
  color: #000000;
  font-weight: 700;
  cursor: pointer;
  text-transform: uppercase;
}

label {
  font-weight: bold;
}

.error {
  color: red;
  font-size: 14px;
}

```

Fonte: Do próprio autor, 2024.

Desta forma com o código acima resulta-se em um formulário mais apresentável e estilizado de forma simples.

Figura 7 – Resultado do formulário HTML estilizado em CSS



Fonte: Do próprio autor, 2024.

2.4 JavaScript

Tendo sua primeira versão lançada em 1995, JavaScript foi desenvolvido pela Netscape em colaboração com a Sun Microsystems, sendo parte fundamental da tríade de tecnologias *web*, juntamente com o HTML e CSS. (Iepsen, 2022). O JavaScript é altamente eficaz e uma das linguagens mais populares do mundo, ocupando uma posição proeminente na internet. (Groner 2019). Apesar de possuir um nome semelhante, JavaScript diferentemente do Java, evolui de uma linguagem de *script* para uma linguagem de uso geral, com recursos modernos voltados para o desenvolvimento de *software* em larga escala. (Flanagan, 2013). É uma linguagem de programação do lado do cliente, o que significa que sua interpretação e execução dependem das funcionalidades do navegador do usuário, que normalmente possuem um interpretador JavaScript integrado. (Silva, 2010).

Figura 8 – Elemento “onsubmit” no HTML

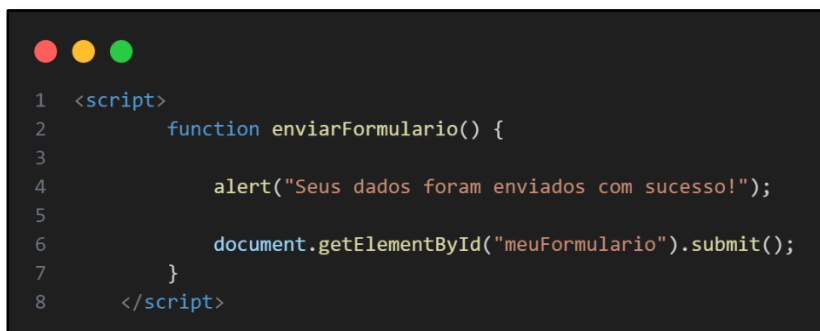


```
1 <form id="meuFormulario" method="post" onsubmit="enviarFormulario();">
```

Fonte: Do próprio autor, 2024.

- Onsubmit: é usado em um formulário HTML para especificar a ação que deve ser executada quando o formulário é enviado.

Figura 9 – Exemplo de código JavaScript



```
1 <script>
2     function enviarFormulario() {
3
4         alert("Seus dados foram enviados com sucesso!");
5
6         document.getElementById("meuFormulario").submit();
7     }
8 </script>
```

Fonte: Do próprio autor, 2024.

- *Script*: “<script></script>” os códigos *script* deve ser escritos dentro da *tag SCRIPT*, ou podem ser colocados em um arquivo separado com extensão .js, que deve ser referenciado no documento HTML.
- *Function*: “function enviarFormulario()” define uma função chamada enviarFormulario() que será chamada quando necessário.
- *Alert*: “alert()” exibirá uma caixa de diálogo com uma mensagem quando a função for executada.
- *Function*: “document.getElementById().submit()” obtém o elemento HTML pelo ID usando a função document.getElementById(). Em seguida, a função submit() é chamada nesse formulário, enviando os dados do formulário para um servidor.

Figura 10 – Resultado da programação JavaScript

A imagem mostra uma interface web com um formulário centralizado. No topo, uma barra de favoritos contém o texto "Adicione seus favoritos aqui na barra de favoritos. Gerenciar favoritos aqui". Abaixo, uma caixa de diálogo branca com o título "Esta página diz" e o conteúdo "Seus dados foram enviados com sucesso!" e um botão "OK" azul. O formulário principal, intitulado "FORMULÁRIO", contém os seguintes campos: "Nome:" com o valor "Mia Rodrigues", "E-mail:" com o valor "miarodrigues@gmail.com", "Telefone:" com o valor "11 99999-9999" e "Endereço:" com o valor "Rua ABC, 123". Um botão "ENTRAR" está localizado na base do formulário.

Fonte: Do próprio autor, 2024.

2.5 UML

O UML, ou melhor *Unified Modeling Language*, é uma linguagem visual muito utilizada para retratar *software* orientado a objetos. É uma linguagem versátil aplicável a diversos domínios e é adotada como padrão internacional na engenharia de *software*. (GUEDES, 2018). Surgiu da união de diversas linguagens gráficas de modelagem orientadas a objetos que surgiram nos anos 80 e 90, assim desde sua introdução em 1997, vem sendo um recurso valorizado por muitos desenvolvedores. (FOWLER, 2005).

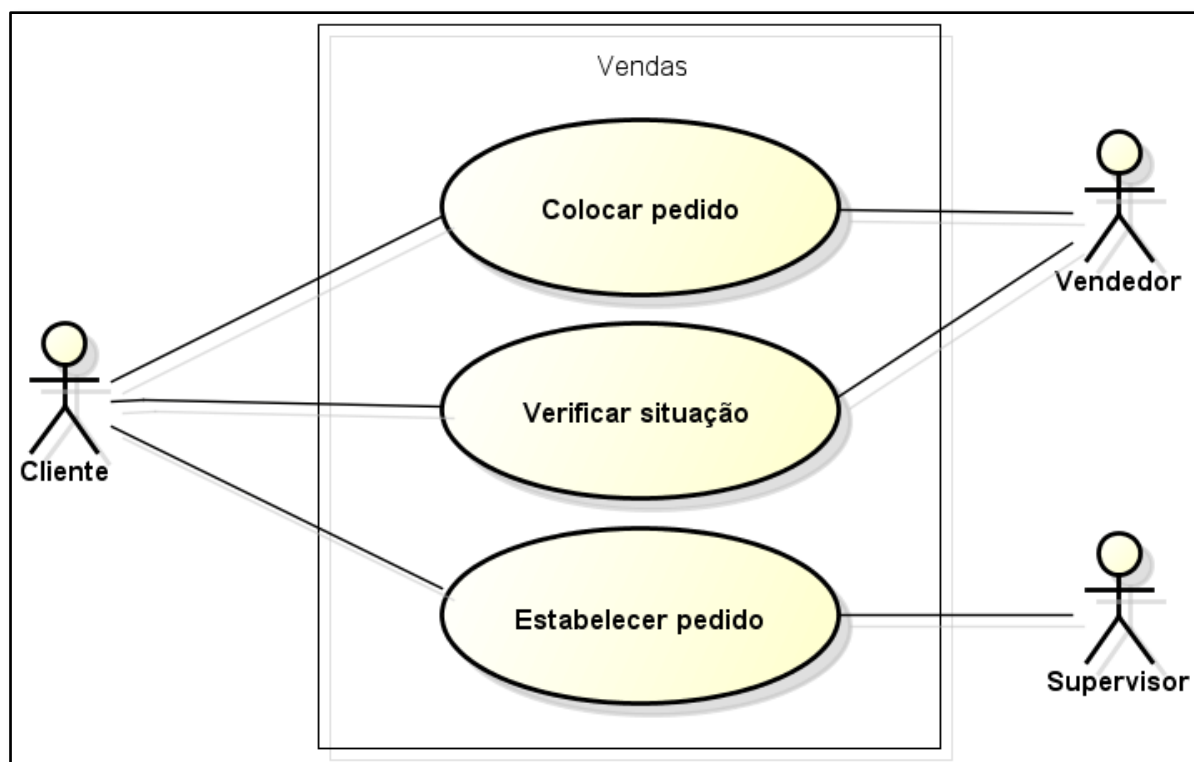
Modelagem está ligada a comunicação, e a UML concede ferramentas importantes para visualizar, especificar, construir e documentar artefatos de diversos sistemas de *software* complexos. (BOOCH, 2006). É importante modelar todo sistema antes de começar sua implementação, pois os sistemas de informação tendem a crescer em tamanho, complexidade e alcance, logo são dinâmicos e estão sempre em evolução. (GUEDES, 2018).

Portanto essa linguagem de modelagem possui diversos diagramas para o melhor entendimento e documentações dos sistemas criados, que serão mais bem abordados nos próximos capítulos.

2.5.1 UML - Diagrama de Casos de Uso

Utilizados para organização e modelagem de como é o comportamento de um sistema, por meio de conjunto de casos de uso que são funcionalidades e atores que interagem com o sistema. (BOOCH, 2006).

Figura 11 – Exemplo de diagrama de Caso de Uso



Fonte: Lima, 2009.

2.5.2 UML - Documentação Caso de Uso

A documentação de um caso de uso descreve, através de uma linguagem simples, informações nas quais os atores interagem aos casos de uso, como execução dos parâmetros, restrições e validações. (GUEDES, 2018). Abaixo podemos analisar de uma forma lúdica um exemplo de documentação de caso de uso da abertura de uma conta.

Tabela 1 - Exemplo Documentação de Caso de Uso abertura de uma conta

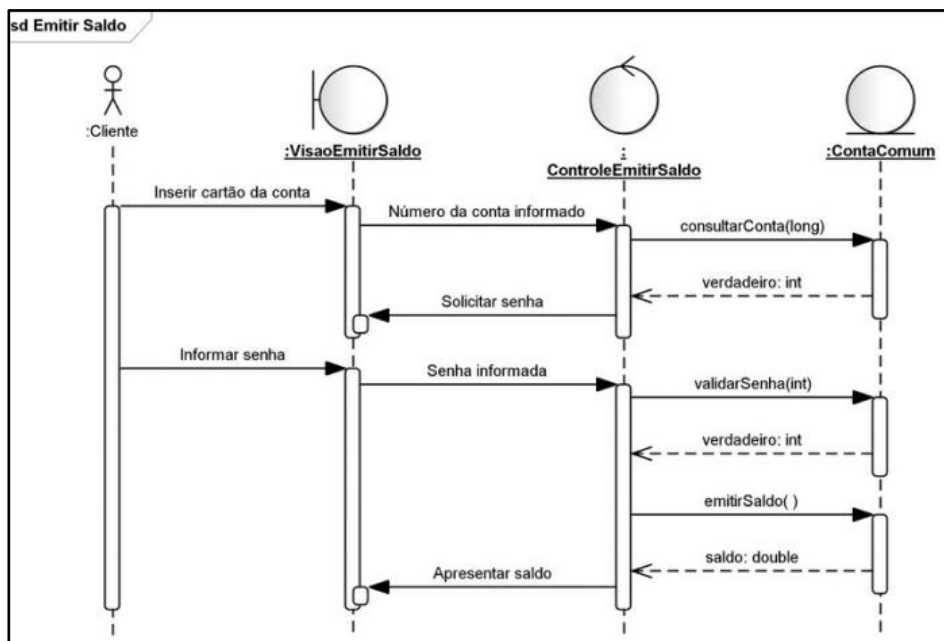
Nome do Caso de Uso		UC01 – Abrir Conta
Caso de Uso Geral		
Ator Principal		Funcionário
Atores Secundários		Cliente
Resumo		Esse caso de uso descreve as etapas percorridas por um cliente, intermediado por um funcionário, para abrir uma conta-corrente
Pré-condições		O pedido de abertura precisa ter sido previamente aprovado
Pós-condições		É necessário realizar um depósito inicial
		Cenário Principal
Ações do Ator		Ações do Sistema
2. O funcionário informa o CPF ou CNPJ do cliente e consulta seu registro		3. Consultar cliente por seu CPF ou CNPJ
4. O cliente informa a senha da conta		5. Abrir conta
6. O cliente fornece um valor a ser depositado		7. Executar caso de uso "Realizar Depósito" para registrar o depósito do cliente
		8. Emitir cartão da conta
Restrições/Validações		1. Para abrir uma conta-corrente, é preciso ser maior de idade
		2. O valor mínimo de depósito é R\$ 5,00
		3. O cliente precisa fornecer algum comprovante de residência
		Cenário Alternativo – Manutenção do Cadastro do Cliente
Ações do Ator		Ações do Sistema
		1. Executar o Caso de Uso "Gerenciar Clientes", para registrar um novo cliente ou atualizar o cadastro do cliente consultado
		Cenário de Exceção – Cliente menor de idade
Ações do Ator		Ações do Sistema
		1. Comunicar ao cliente que ele não tem a idade mínima para possuir uma conta-corrente
		2. Recusar o pedido

Fonte: Guedes, 2018.

2.5.3 UML - Diagrama de Sequência

Utilizado para explorar como determinada ação ocorre, apresentando diversos exemplos de comunicações entre os elementos passados pelos objetos envolvidos nos casos de uso (FOWLER, 2005).

Figura 12 – Exemplo de diagrama de Sequência

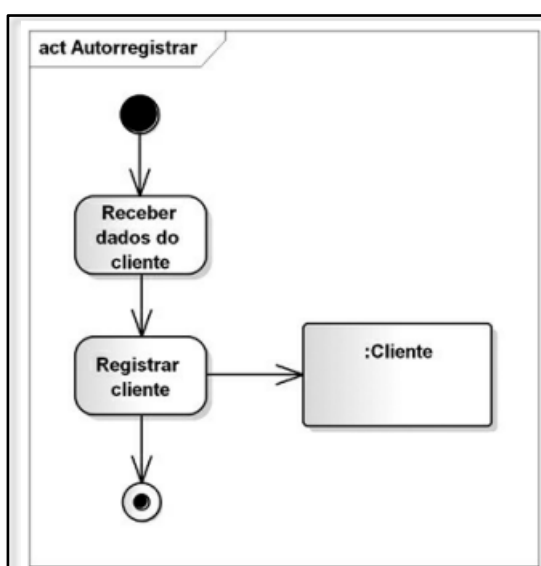


Fonte: Guedes, 2018.

2.5.4 UML - Diagrama de Atividade

Os diagramas de atividades são usados para modelar aspectos dinâmicos de um sistema, mostrando etapas sequenciais ou concorrentes de um processo computacional e o fluxo de objetos entre estados (BOOCH, 2006).

Figura 13 – Exemplo de diagrama de Atividade

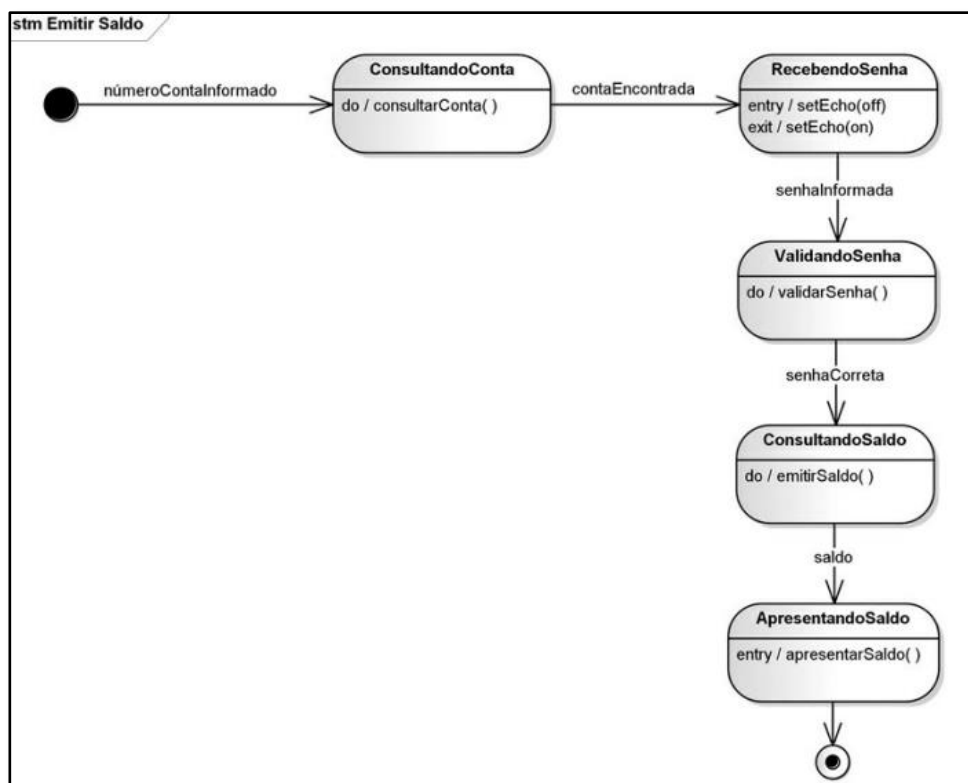


Fonte: Guedes, 2018.

2.5.5 UML - Diagrama de Máquina de Estados

Os diagramas de máquina de estados descrevem o comportamento de um sistema, mostrando o ciclo de vida de um objeto de uma classe eles são usados desde os anos 60, especialmente em técnicas orientadas a objetos (FOWLER, 2005).

Figura 14 – Exemplo de diagrama de Máquina de Estados



Fonte: Guedes, 2018.

2.6 Wireframe

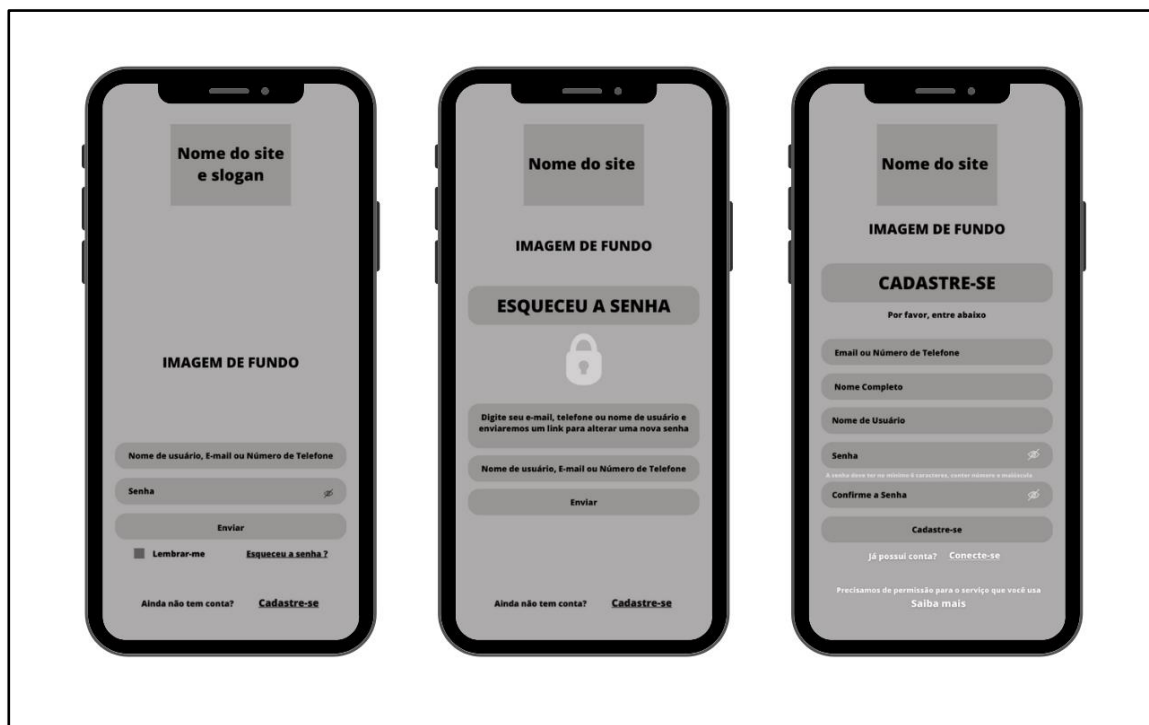
O termo “*wireframe*” descreve a modelagem no desenvolvimento de páginas na *web* em 3D para animações por computadores e em 2D para aplicativos móveis. (LUCIDCHART, 2023).

O aspecto crucial para utilização de *wireframes* é a aproximação para a prévia do *design* para o cliente, justamente pela capacidade de análise crítica e construtiva do mesmo. (Orgânica Digital, 2022).

Existem três tipos de *wireframe* de acordo com (ROCKCONTENT, 2019) o de Baixa Fidelidade que é feito manualmente e sem muitos detalhes, o Anotado que contém legendas e descrições, e o de Alta Fidelidade que é o mais próximo da versão final.

Para maior compreensão abaixo podemos encontrar exemplos de *wireframes* de Baixa Fidelidade e de Alta fidelidade.

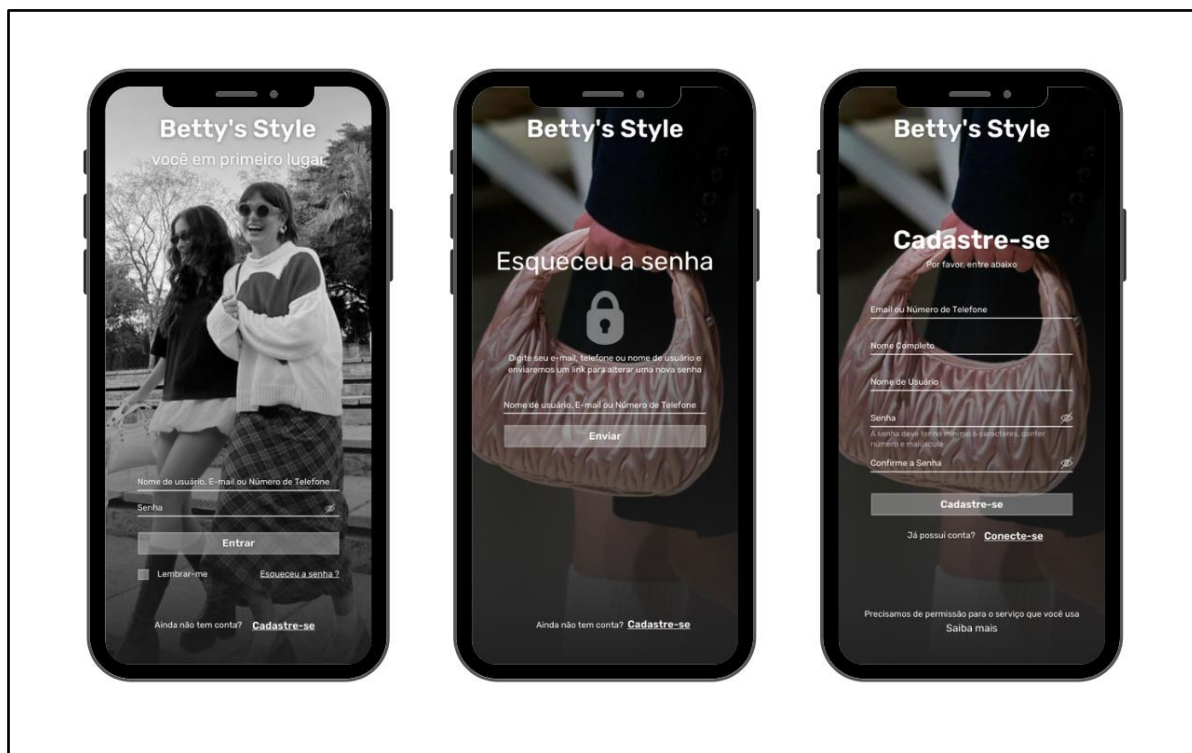
Figura 15 – Exemplo de *wireframe* de baixa fidelidade



Fonte: Do próprio autor, 2024.

Acima, podemos ver o *wireframe* de baixa fidelidade, com as telas de *Login*, *Esqueceu a Senha* e *Cadastre-se*, e abaixo, o de alta fidelidade.

Figura 16 – Exemplo de *wireframe* de alta fidelidade



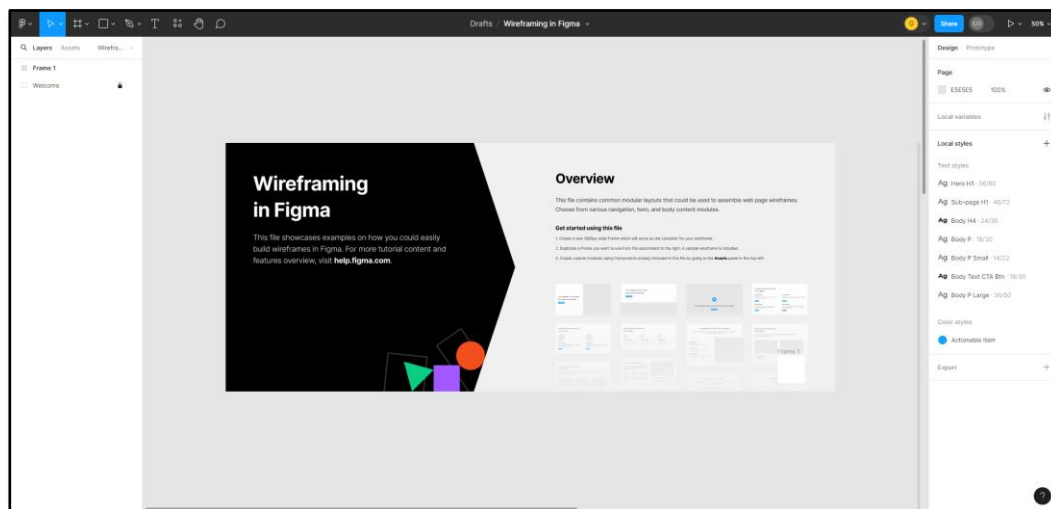
Fonte: Do próprio autor, 2024.

2.7 Figma

O Figma é uma plataforma de design que permite criar, colaborar e prototipar interfaces de usuários e *designs* interativos. Diferentemente de outras ferramentas de design o Figma é totalmente acessível pelo navegador. (Cubos Academy, 2023).

De acordo com Ebac Online (2023), vale ressaltar a importância do Figma como uma ferramenta que tem sido muito utilizada para entrega de projetos digitais de ponta a ponta, sendo um projeto *web* ou um aplicativo *mobile*. Abaixo encontra-se uma imagem da tela principal de criação de projetos do aplicativo.

Figura 17 – Tela de criação de projetos do Figma



Fonte: Figma, 2024.

2.8 Banco de dados

Um sistema de banco de dados é basicamente um sistema computadorizado de manutenção de registros. Os registros em questão podem ser qualquer coisa que tenha significado ao indivíduo ou a organização a que o sistema deve servir. (DATE, 2023).

2.8 Firebase

Conforme Oracle (2023), o Firebase é uma plataforma de desenvolvimento de aplicativos *mobile* e *web* do Google, é vantajoso incluindo um ambiente de desenvolvimento ponta a ponta, rápido para criar aplicativos e infraestrutura escalável.

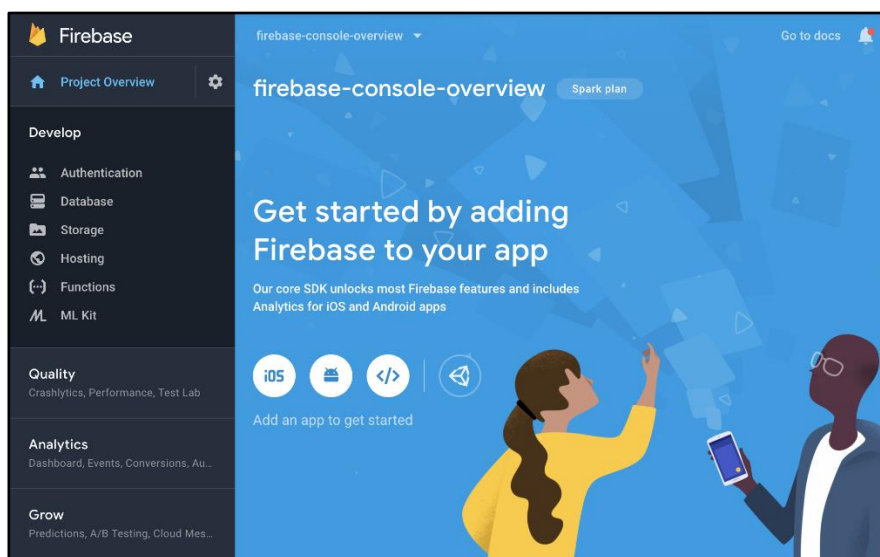
Figura 18 – Logotipo Firebase



Fonte: Firebase, 2023.

O console do Firebase oferece o ambiente mais sofisticado para gerenciar produtos, apps e configurações do nível do projeto Firebase, abaixo podemos perceber que sua lista de produtos é organizada por categorias de nível superior. (FIREBASE, 2023).

Figura 19 – Console Firebase



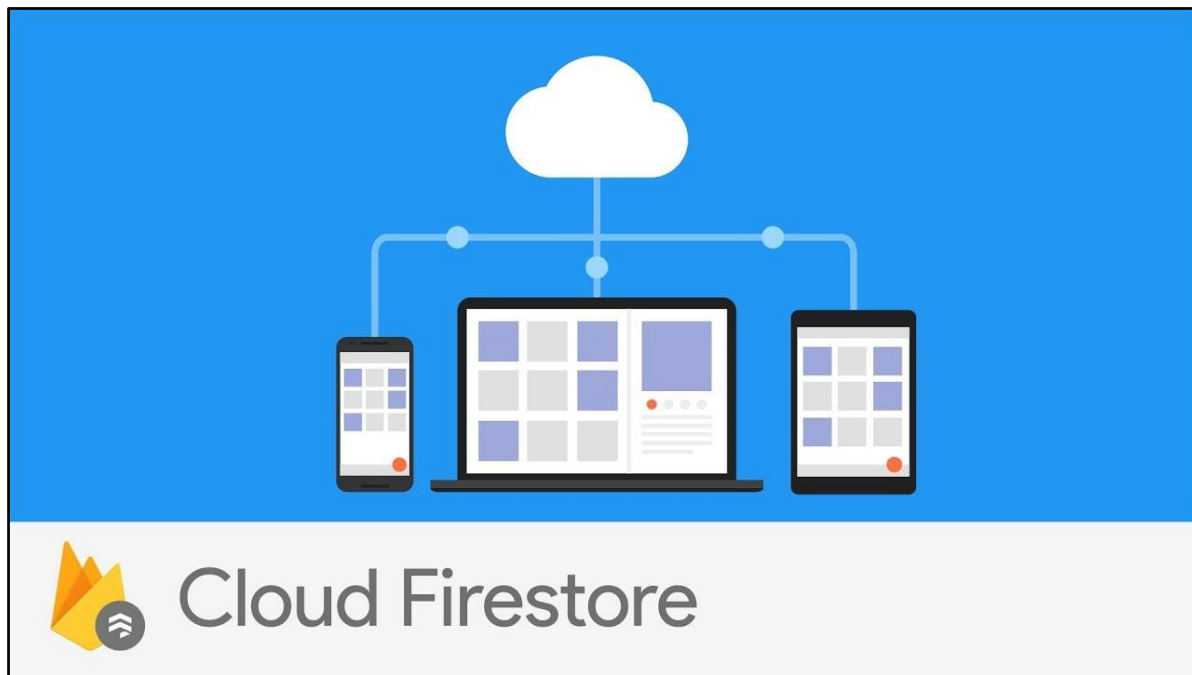
Fonte: Firebase, 2023.

2.9 Firestore

O Cloud Firestore é um banco de dados NoSQL, que permite armazenar, sincronizar e consultar dados nos dispositivos móveis e *web*, além disso disponibiliza regras de segurança para acessar o banco sem precisar manter o próprio servidor. (Firebase).

Além disso, tem o foco no desenvolvimento do aplicativo usando banco de dados de documentos gerenciado e sem servidor, fazendo ajustes com o objetivo de atender qualquer demanda sem janelas de manutenção ou inatividade. (GOOGLE CLOUD, 2024).

Figura 20 – Cloud Firestore



Fonte: Firebase, 2023.

2.10 Node.js & NPM

Node.js, criado em 2009, é uma plataforma de execução de JavaScript no servidor, conhecida por sua arquitetura assíncrona e eficiência em operações de I/O, ideal para desenvolver APIs, aplicações de tempo real e *backends* escaláveis. (PEREIRA, 2014).

O Node.js utiliza um modelo *non-blocking thread*, otimizando o processamento ao eliminar paralisações por I/O, isso permite criar aplicações escaláveis e eficientes, sem esperas prolongadas, devido à sua arquitetura orientada a eventos. (DUARTE, 2020).

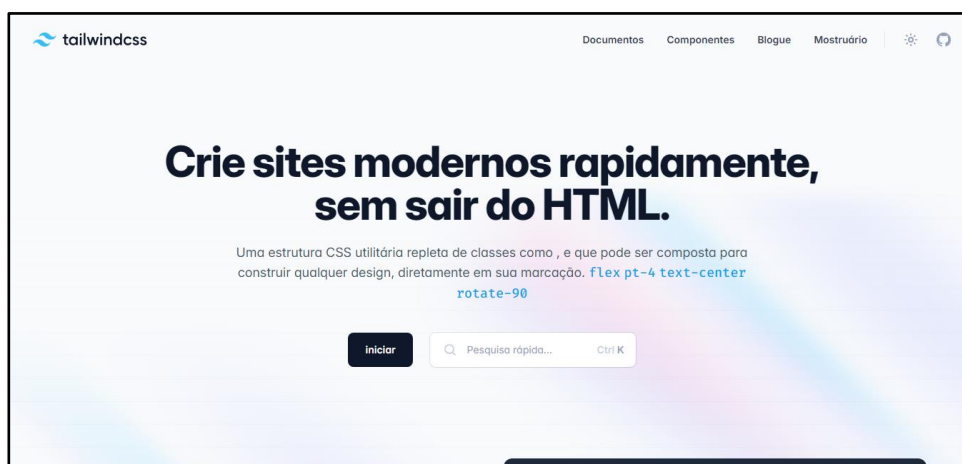
NPM é o gerenciador de pacotes do Node.js, facilitando a gestão de dependências, acesso a bibliotecas JavaScript e controle de versões, simplificando o desenvolvimento e compartilhamento de projetos. (GADO, 2021).

2.11 Tailwind

O Tailwind é um *framework* que facilita a estilização de páginas HTML, evitando a sobrecarga de CSS, que com suas classes, mantém cores, espaçamento, tipografia e sombras, criando um design elegante e eficiente. (TAILWIND CSS). O projeto

começou em 2017 por Adam Wathan, visando criar um *framework* CSS mais fácil e com maior flexibilidade nos estilos, o que na época já existiam, mas que para ele eram complicados de personalizar. (MARCELA, 2023). Oferece classes CSS pré-definidas para estilizar páginas *web* de forma direta, dentro da marcação HTML, podendo aplicar estilos facilmente, agilizando o processo de desenvolvimento *front-end*. (ABBA, 2022).

Figura 21 – Tela de apresentação do site Tailwind CSS



Fonte: Tailwind CSS, 2024.

2.11.1 Instalação do Tailwind via NPM (Node Package Manager)

Deve se atentar em ter o Node.js instalado no sistema, que pode ser baixado em seu site oficial. Após a certificação do mesmo, inicia-se o processo de instalação do Tailwind CSS como descrito abaixo:

- 1 - Crie um projeto ou abra um projeto existente no terminal.

Figura 22 – Terminal exibindo a criação de um novo projeto

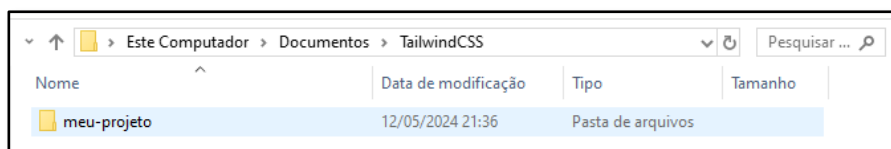
```

Prompt de Comando
Microsoft Windows [versão 10.0.19045.4291]
(c) Microsoft Corporation. Todos os direitos reservados.
C:\Users\desiree>CD C:\Users\desiree\Documents\TailwindCSS
C:\Users\desiree\Documents\TailwindCSS>mkdir meu-projeto
C:\Users\desiree\Documents\TailwindCSS>cd meu-projeto
C:\Users\desiree\Documents\TailwindCSS\meu-projeto>_
```

Fonte: Do próprio autor, 2024.

Pode-se perceber que entre as imagens o processo de criação e organização de projetos. O terminal exibe o início do projeto, enquanto a pasta mostra a estrutura e localização dos arquivos.

Figura 23 – Visualização da pasta do projeto recém-criado



Fonte: Do próprio autor, 2024.

- 2 - No terminal dentro da pasta do projeto criado, instale o Tailwind CSS via npm e faça a criação do arquivo *tailwind.config.js* (TAILWIND CSS) como mostrado abaixo:

Figura 24 – Terminal mostrando a instalação do Tailwind

```
C:\Users\desiree\Documents\TailwindCSS\meu-projeto>npm install -D tailwindcss
up to date, audited 113 packages in 15s
29 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

Fonte: Do próprio autor, 2024.

Entre as imagens, acompanhamos o processo de configuração do Tailwind CSS no projeto.

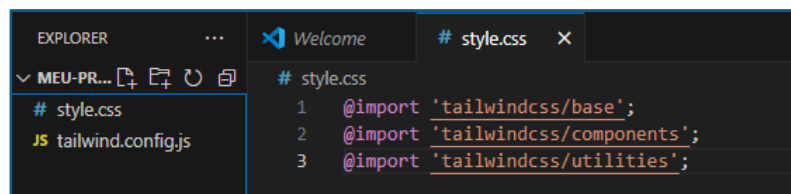
Figura 25 – Terminal mostrando a criação do arquivo *tailwind.config.js*

```
C:\Users\desiree\Documents\TailwindCSS\meu-projeto>npx tailwindcss init
Created Tailwind CSS config file: tailwind.config.js
```

Fonte: Do próprio autor, 2024.

- 3 – Crie um arquivo CSS faça a importação o Tailwind e adicione os caminhos de modelo no arquivo *tailwind.config.js*. (MARCELA, 2023).

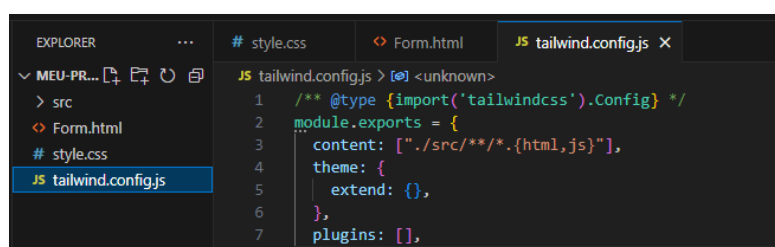
Figura 26 – Arquivo CSS com a importação do Tailwind



Fonte: Do próprio autor, 2024.

No espaço entre as ilustrações, podemos visualizar as etapas para integrar o Tailwind CSS ao projeto.

Figura 27 – Configuração dos caminhos no arquivo tailwind.config.js



Fonte: Do próprio autor, 2024.

Em suma, é possível criar projetos mais elegantes em menos tempo de forma mais eficiente e com qualidade, no qual abaixo trata-se de um modelo de formulário usando esse *framework*:

Figura 28 – Modelo de formulário usando Tailwind.

Fonte: Do próprio autor, 2024.

Aqui, apresentamos a representação visual de um formulário criado em Tailwind CSS, juntamente com o código respondente abaixo.

Figura 29 – Código do formulário usando Tailwind

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Formulário com Tailwind CSS</title>
7    <link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">
8  </head>
9  <body>
10   <div class="min-h-screen flex items-center justify-center bg-gray-50 py-12 px-4 sm:px-6 lg:px-8">
11     <div class="max-w-md w-full space-y-8">
12       <div>
13         <h2 class="mt-6 text-center text-3xl font-extrabold text-gray-900">
14           Cadastre-se
15         </h2>
16       </div>
17       <form class="mt-8 space-y-6" action="#" method="POST">
18         <input type="hidden" name="remember" value="true">
19         <div class="rounded-md shadow-sm -space-y-px">
20           <div>
21             <label for="email-address" class="sr-only">Endereço de e-mail</label>
22             <input id="email-address" name="email" type="email" autocomplete="email"
23               required class="appearance-none rounded-none relative block w-full px-3 py-2 border
24               border-gray-300 placeholder-gray-500 text-gray-900 rounded-t-md focus:outline-none
25               focus:ring-indigo-500 focus:border-indigo-500 focus:z-10 sm:text-sm" placeholder="Endereço de e-mail">
26           </div>
27           <div>
28             <label for="password" class="sr-only">Senha</label>
29             <input id="password" name="password" type="password" autocomplete="current-password"
30               required class="appearance-none rounded-none relative block w-full px-3 py-2 border
31               border-gray-300 placeholder-gray-500 text-gray-900 rounded-b-md focus:outline-none
32               focus:ring-indigo-500 focus:border-indigo-500 focus:z-10 sm:text-sm" placeholder="Senha">
33           </div>
34         </div>
35
36         <div class="flex items-center justify-between">
37           <div class="flex items-center">
38             <input id="remember-me" name="remember-me" type="checkbox" class="h-4 w-4 text-indigo-600
39               focus:ring-indigo-500 border-gray-300 rounded">
40             <label for="remember-me" class="ml-2 block text-sm text-gray-900">
41               Lembrar-me
42             </label>
43           </div>
44
45           <div class="text-sm">
46             <a href="#" class="font-medium text-indigo-600 hover:text-indigo-500">
47               Esqueceu sua senha?
48             </a>
49           </div>
50         </div>
51
52         <div>
53           <button type="submit" class="group relative w-full flex justify-center py-2 px-4 border border-transparent
54             text-sm font-medium rounded-md text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-2
55             focus:ring-offset-2 focus:ring-indigo-500">
56             <span class="absolute left-0 inset-y-0 flex items-center pl-3">
57               <svg class="h-5 w-5 text-indigo-500 group-hover:text-indigo-400" xmlns="http://www.w3.org/2000/svg"
58                 viewBox="0 0 20 20" fill="currentColor" aria-hidden="true">
59                 <path fill-rule="evenodd" d="M3 4a2 2 0 012-2h10a2 2 0 012 2v6a2 2 0 01-2 2H5a2 2 0 01-2 2V4zm2-1a1 1 0 00-1 1v6a1 1 0 001 1h10a1 1 0 001 1v6a1 1 0 00-1 1H5a1 1 0 00-1 1H3z" clip-rule="evenodd" />
60               </svg>
61             </span>
62             Entrar
63           </button>
64         </div>
65       </form>
66     </div>
67   </div>
68 </body>
69 </html>
70
71

```

Fonte: Do próprio autor, 2024.

2.12 React

React é uma biblioteca em JavaScript que simplifica e agiliza a tarefa de desenvolvimento de interfaces de usuário interativas e de alto desempenho, é usado por grandes plataformas como Facebook e Instagram. (SILVA, 2021).

Suas aplicações são compostas por componentes, que são para da IU (interface do usuário) que possui sua própria lógica e aparência. (React, 2024).

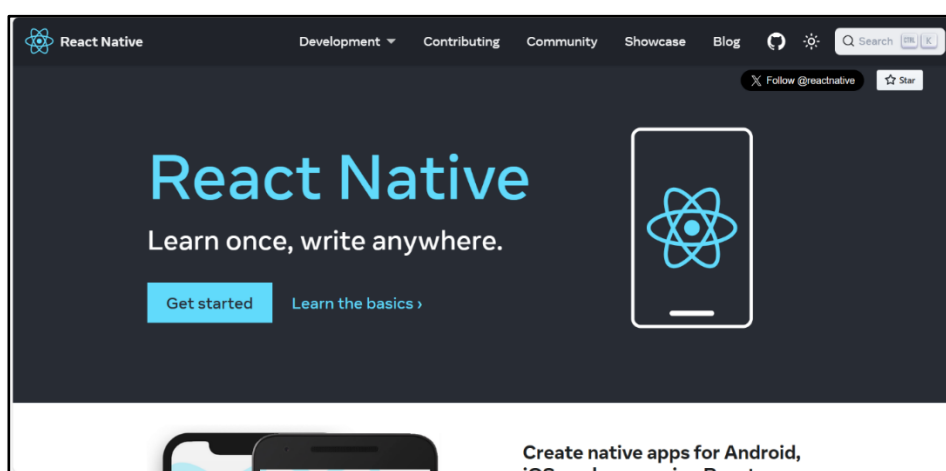
2.13 React Native

O React Native, baseado no React e desenvolvido pelo Facebook, é um *framework* JavaScript para criar aplicativos multiplataforma. Ele combina desenvolvimento nativo com React, oferecendo uma experiência autêntica aos usuários. (Escudelario; Pinho, 2020).

Com uma abordagem declarativa e compartilhamento de código, ele agiliza o desenvolvimento, possibilitando que equipes criem aplicativos nativos de forma eficiente. (React Native).

O funcionamento do React Native depende do Node.js, que converte o código JavaScript para plataformas como Android e iOS. Além disso, o Node.js gerencia as bibliotecas e pacotes essenciais para o projeto. (DevMedia).

Figura 30 – Tela de apresentação do site React Native

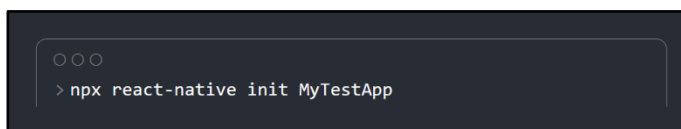


Fonte: React Native, 2024.

2.13.1 Criação de um aplicativo em React Native

Para criar um aplicativo React Native, você pode utilizar o seguinte comando no Prompt de Comando, após instalar o Node.js e entrar no diretório desejado:

Figura 31 – Comando de criação projeto React Native



Fonte: React Native, 2024.

Isso criará um aplicativo React Native chamado MyTestApp no diretório atual. Após a criação do projeto, você poderá desenvolver a aplicação utilizando a IDE de sua preferência.

Figura 32 – Exemplo de código React Native

```

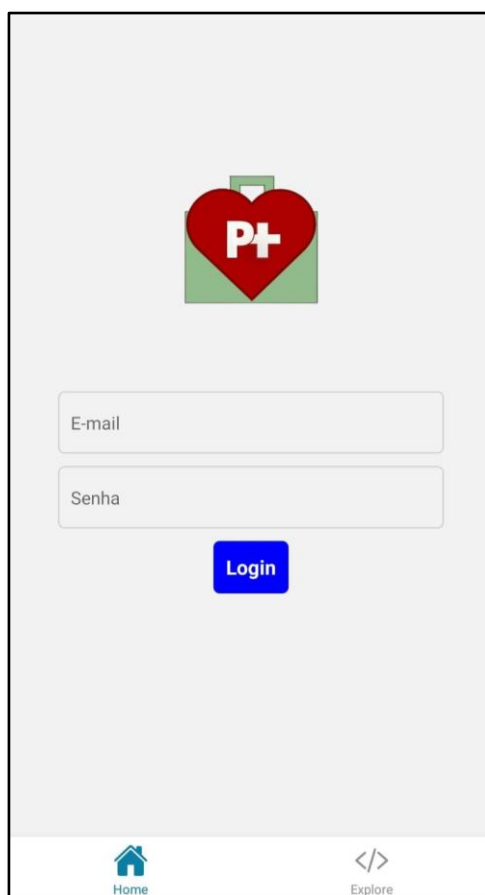
1  import React from 'react';
2  import { View, Text, TextInput, TouchableOpacity, StyleSheet, Image } from 'react-native';
3
4  const LoginScreen = () => {
5    return (
6      <View style={styles.container}>
7        <Image
8          source={require('./src/assets/logor.png')}
9          style={styles.image}
10         />
11        <TextInput
12          style={styles.input}
13          placeholder="E-mail"
14          autoCapitalize="none"
15         />
16        <TextInput
17          style={styles.input}
18          placeholder="Senha"
19          secureTextEntry
20         />
21        <TouchableOpacity style={styles.button}>
22          <Text style={styles.buttonText}>Login</Text>
23        </TouchableOpacity>
24      </View>
25    );
26  };
27
28  const styles = StyleSheet.create({
29    container: {
30      flex: 1,
31      justifyContent: 'center',
32      alignItems: 'center',
33    },
34    image: {
35      width: 200,
36      height: 200,
37      marginBottom: 10,
38    },
39    input: {
40      width: '80%',
41      marginBottom: 10,
42      padding: 10,
43      borderWidth: 1,
44      borderColor: '#ccc',
45      borderRadius: 5,
46    },
47    button: {
48      backgroundColor: 'blue',
49      padding: 10,
50      borderRadius: 5,
51    },
52    buttonText: {
53      color: 'white',
54      fontSize: 16,
55      fontWeight: 'bold',
56    },
57  });
58
59  export default LoginScreen;
60

```

Fonte: Do próprio autor, 2024.

Aqui, exibimos o formulário criado com React Native, junto com o código equivalente.

Figura 33 – Resultado da codificação React Native



Fonte: Do próprio autor, 2024.

2.14 Express.js

Express.js é um framework back-end minimalista, rápido e escalável para Node.js. Ele oferece funcionalidades robustas, um sistema de roteamento eficiente e recursos simplificados para estender o framework, permitindo a criação de componentes poderosos. (KINSTA). Ele gerencia dados, autenticação de usuários e integração com APIs, atendendo às necessidades do aplicativo. (MARCELA, 2023).

2.15 IoT

A Internet das Coisas, conhecida como, IoT é a rede de dispositivos e, em geral, as coisas que estão conectadas e se comunicam entre si e através de redes, para realizar determinadas tarefas sem exigir interação entre seres humanos. (SANTOS, 2018).

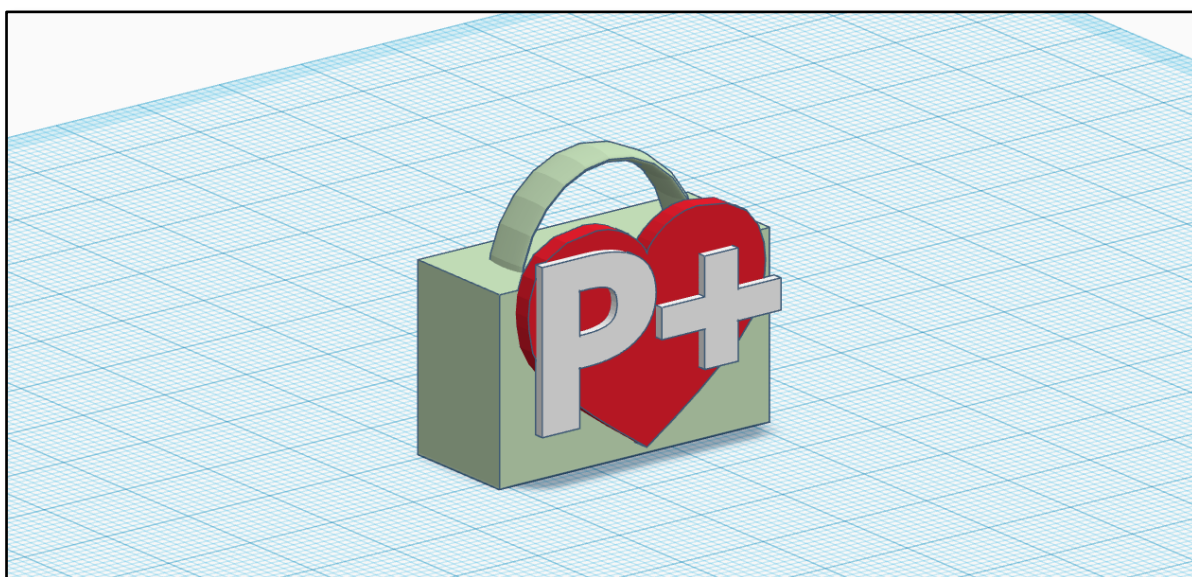
O “produto IoT” vai além do produto inteligente e do produto conectado, por explorarem toda a capacidade da internet em produtos físicos, sendo assim, ele é efetivamente um sistema, ou, melhor dizendo um sistema de sistemas. (SINCLAIR, 2018).

O avanço das tecnologias, especialmente as digitais, iram afetar profundamente todas as estruturas econômicas e sociais, e talvez, a mais impactante e pervasiva dessas tecnologias digitais seja a internet das coisas. (MAGRANI, 2018).

2.15.1 Modelagem 3D

A modelagem 3D é considerada um processo de desenvolvimento de personagens, objetos ou cenários em três dimensões, ou seja, possuem profundidade além de altura e largura. (LOPES, 2023). Abaixo temos um exemplo de Modelagem 3D.

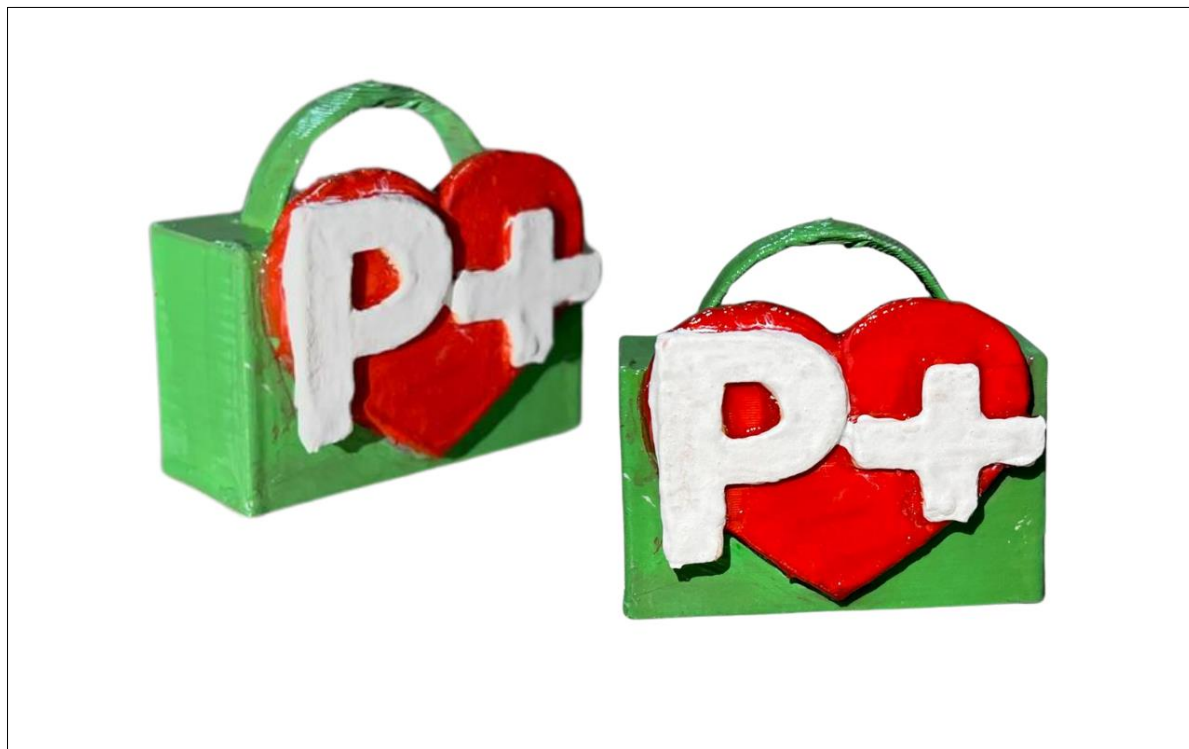
Figura 34 – Exemplo de Modelagem 3D



Fonte: Do próprio autor, 2024.

Acima, apresentamos o protótipo de um modelo criado para impressão 3D. Abaixo, você pode ver a foto do modelo após ser impresso e pintado.

Figura 35 – Exemplo de prototipagem 3D impressa e pintada



Fonte: Do próprio autor, 2024.

2.15.2 ESP 32

Desenvolvido pela Espressif Systems, o ESP-32 é um microcontrolador eficiente e de baixo custo, excelente para projetos de IoT, entretenimento e automação residencial, possuindo suporte de WIFI, Bluetooth e outros tipos de conexão. (MAKIYAMA, 2023).

Posto isso, ESP-32 se torna ideal para projetos de IoT visto que possui capacidade de conexão à internet e a outros dispositivos, com processador dual-core e 500 KB de SRAM, que permite execução de programas complexos. (PEREIRA, 2020). Assim o ESP-32 possui 36 pinos digitais com 16 utilizáveis como PWM, e portando entradas e saídas, onde entradas, são como pressionar um botão, que enviam sinais ao microcontrolador, e pode ativar saídas como *LEDs* e motores. (ELETRÔNICA ÔMEGA, 2021)

Figura 36 – ESP 32

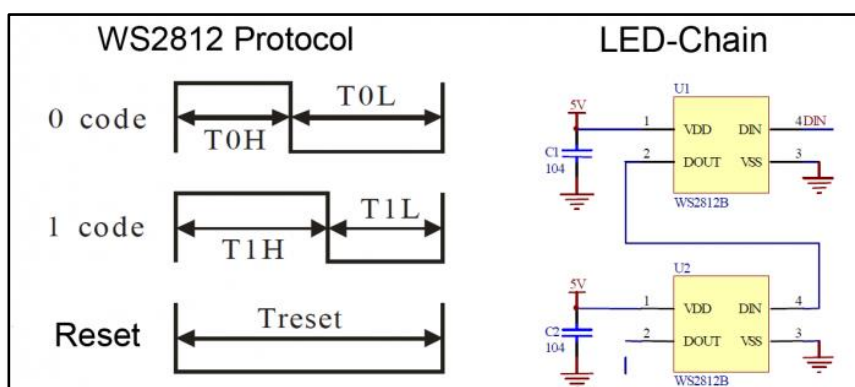


Fonte: RoboCore, 2024.

2.15.3 Fita LED RGB endereçável

A fita *LED (Light Emitting Diode) RGB (Red, Green, Blue)* endereçável, foi desenvolvida para suprir uma nova necessidade do mercado, possui um *LED RGB* em seu encapsulamento e um microcontrolador que é responsável por receber um sinal digital de um protocolo de comunicação específico. (CASTRO), seus *LEDs* possuem um endereço individual e único, e com isso pode-se controlar qualquer *LED* com somente um fio de dados, simplesmente passando o endereço do *led* e a cor que se deseja, que é controlada pelo sistema de cores *RGB*. (STROSCHON, 2020). Abaixo encontra-se um esquema básico de ligação e controle:

Figura 37 – Exemplo de conexão e comunicação do LED RGB Endereçável



Fonte: Tim's Blog

É possível observar na imagem acima, um modelo de componente que possui 4 terminais, porém neste caso 2 são para alimentação positiva e negativa, enquanto os outros dois são para receber e enviar sinais digitais. (CASTRO). Segue agora a imagem de uma fita *LED RGB* endereçável.

Figura 38 – Fita LED RGB endereçável



Fonte: RoboCore, 2024.

2.15.6 Display LCD

Criado pelo engenheiro George Heilmer em 1964, *LCD (Liquid Crystal Display)* é uma tecnologia que utiliza de cristais líquidos e polarizadores de luz para formar imagens, comuns em eletrônicos como celulares e TVs (HIGA; MARQUES, 2023). Os displays *LCD* alfanuméricos são encontrados em diversos aparelhos, possuem interfaces práticas e embora sejam uma tecnologia de mais de vinte anos, continuam populares e econômicos. (PUHLMANN, 2015). A interface para conectar um microcontrolador é padronizada, variando entre 14 e 16 pinos conforme a presença de *backlight*, que possui uma corrente típica de 60 mA e máxima de 75 mA, com tensões diretas de 3,5V e 3,6V, respectivamente (PUHLMANN, 2015).

Figura 39 – Display LCD



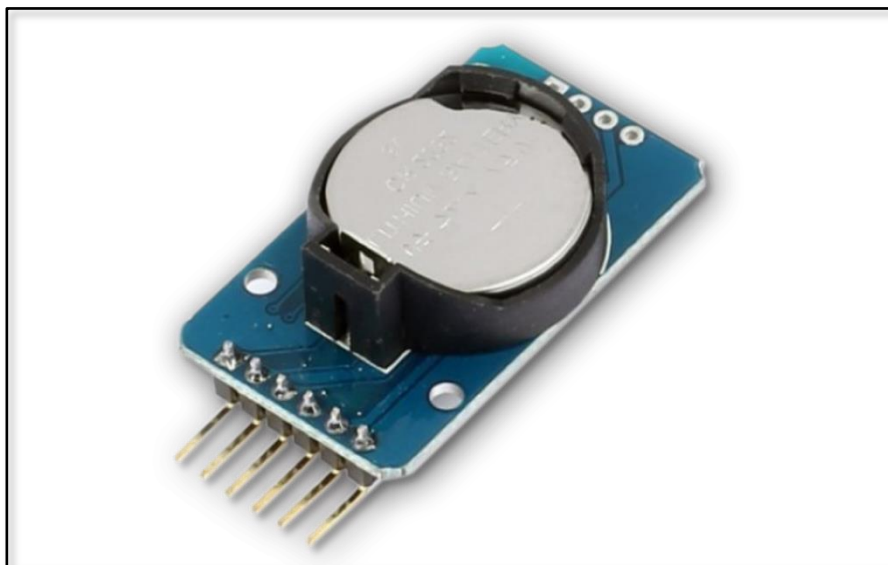
Fonte: ArduCore, 2024.

2.15.5 Módulo RTC DS3231 (Real Time Clock)

O RTC é um circuito capaz de fornecer ao microcontrolador data e hora atualizada. Sua bateria é o que o permite continuar calculando o horário mesmo não sendo alimentado pelos pinos VCC e GND. (GUIMARÃES, 2022).

O módulo RTC DS3231 usa o protocolo I2C para se comunicar com o ESP, e sua precisão é de 2ppm. Para sua integração ao microcontrolador, há bibliotecas disponíveis no ambiente de desenvolvimento. (OLIVEIRA, 2017).

Figura 40 – Modulo RTC DS3231 (*Real Time Clock*)



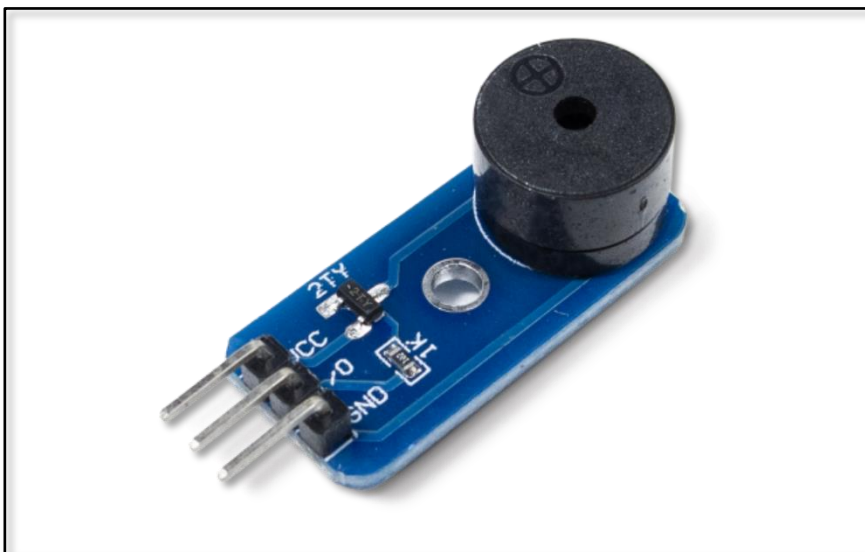
Fonte: RoboCore, 2024.

2.15.6 Buzzer Passivo

Os buzzers são dispositivos eletromecânicos que transformam energia elétrica em som audível, sendo amplamente utilizados em diversas aplicações para emitir alertas ou melodias devido ao seu baixo custo e facilidade de conexão e operação. (Alvarez, 2023).

O buzzer passivo reproduz o som conforme a forma do sinal elétrico que o aciona, permitindo imitar sons específicos, ao contrário do buzzer ativo que emite apenas um apito com timbre próprio. (GUIMARÃES, 2017).

Figura 41– Buzzer Passivo

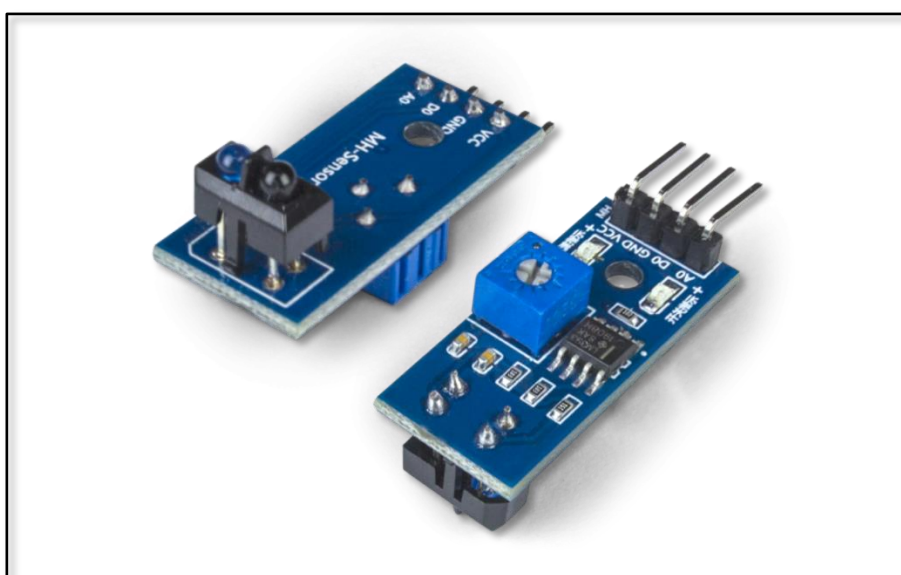


Fonte: RoboCore, 2024.

2.15.7 Sensor Óptico TCRT5000

O TCRT5000 é um sensor óptico reflexivo que utiliza a tecnologia de reflexão infravermelha para detectar a presença de objetos. (ELETROGATE, 2017). Ele é composto por um LED infravermelho e um fototransistor IR, ambos presos num suporte plástico, quando um objeto se aproxima do sensor, a luz infravermelha é refletida para o fototransistor, ativando-o. (Arduino e Cia).

Figura 42 – Sensor Óptico TCRT5000

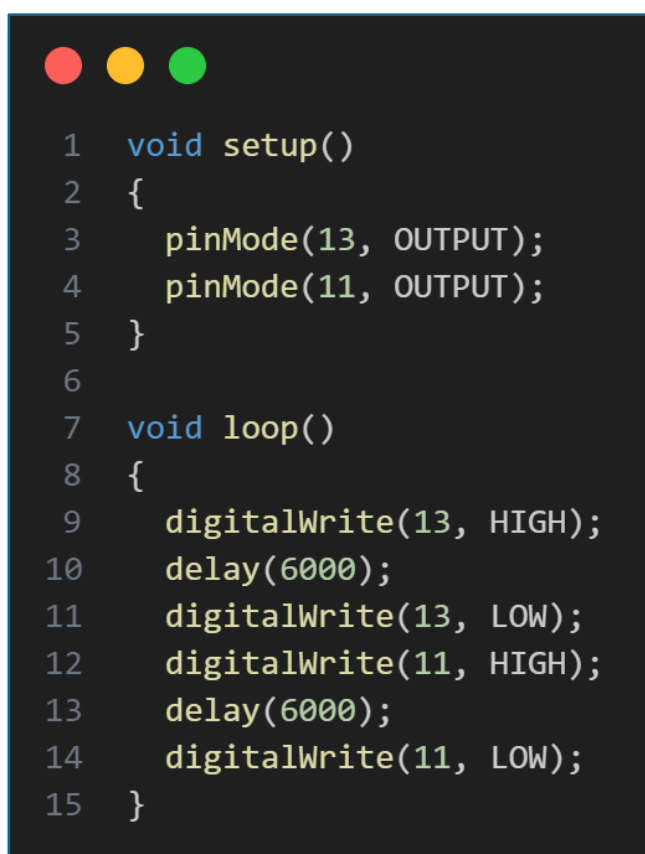


Fonte: RoboCore, 2024.

2.15.8 C++

Desenvolvida por Bjarne Stroustrup na Bell Laboratories nos anos 80, C++ foi criado sendo uma linguagem de programação de baixo nível e fornecendo recursos adicionais a linguagem C. (DEITEL; PAUL, 2015). Sua sintaxe versátil suporta programação orientada a objetos, procedural, genérica e funcional, tornando-a uma escolha poderosa para uma variedade de aplicações em diferentes plataformas. (Locaweb). A filosofia do C++ de construir sobre si mesmo simplifica a criação de compiladores e leva a uma linguagem com uma biblioteca padrão e muitas opções de compiladores. (VILLAS-BOAS, 2001).

Figura 43 – Exemplo C++



```
1  void setup()
2  {
3      pinMode(13, OUTPUT);
4      pinMode(11, OUTPUT);
5  }
6
7  void loop()
8  {
9      digitalWrite(13, HIGH);
10     delay(6000);
11     digitalWrite(13, LOW);
12     digitalWrite(11, HIGH);
13     delay(6000);
14     digitalWrite(11, LOW);
15 }
```

Fonte: Do próprio autor, 2024.

REFERÊNCIAS

- BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivan. **UML: Guia Do Usuário**. 2. ed. [S. l.]. GEN LTC, 2006.
- DATE, Chris John. **Introdução a Sistemas de Banco de Dados**. 1. ed. São Paulo: GEN LTC, 2023.
- DEITEL, Harvey; DEITEL, Paul. **C++ Como Programar**. 5. ed. São Paulo: Pearson Prentice Hall, 2015.
- DUARTE, Luiz. **Programação Web com Node.js**. 6. ed. [S.l.]: LuizTools, 2017.
- ELETRÔNICA ÔMEGA. **E-book Internet das Coisas para Iniciantes com ESP-32**. 1. ed. [S.l.]: Arduino Ômega, 2021. Livro Digital.
- ESCUDELARIO, Bruna; PINHO, Diego. **React Native: Desenvolvimento de Aplicativos Mobile com React**. [S.l.]: Casa do Código, 2020.
- FLANAGAN, David. **JavaScript: O Guia Definitivo**. 6. ed. Porto Alegre: Bookman, 2013.
- FOWLER, Martin. **UML Essencial: Um Breve Guia para a Linguagem Padrão**. [S.l.]: [s.n.].
- GRONER, Loiane. **Estruturas de Dados e Algoritmos com JavaScript: Escreva um Código JavaScript Complexo e Eficaz Usando a Mais Recente ECMAScript**. 2. ed. [S.l.]: Novatec Editora, 2019.
- GUEDES, G. T. A. **UML 2 - Uma Abordagem Prática**. [S.l.]: Novatec Editora, 2018.
- IEPSEN, Edécio Fernando. **Lógica de Programação e Algoritmos com JavaScript - 2ª Edição: Uma Introdução à Programação de Computadores com Exemplos e Exercícios para Iniciantes**. 2. ed. São Paulo: Novatec Editora, 2022.
- JOBSTRAIBIZER, Flávia. **Criação de Sites com o CSS**. São Paulo: Universo dos Livros Editora, 2009.

LUCIANO, J.; ALVES, W. J. B. **Padrão de Arquitetura MVC: Model-View-Controller**. [S.l.]: [s.n.], 2011.

MAGRANI, Eduardo. **A Internet das Coisas**. 1. ed. Rio de Janeiro: FGV Editora, 2018.

PEREIRA, Caio Ribeiro. **Aplicações Web Real-Time com Node.js**. [S.l.]: Casa do Código, 2014.

OLIVEIRA, Sérgio de. **Internet das Coisas com ESP8266, Arduino e Raspberry Pi**. São Paulo: Novatec Editora, 2017.

SILVA, Maurício Samy. **Criando Sites com HTML: Sites de Alta Qualidade com HTML e CSS**. 1. ed. São Paulo: Novatec Editora, 2008.

SILVA, Maurício Samy. **JavaScript - Guia do Programador: Guia Completo das Funcionalidades de Linguagem JavaScript**. 1. ed. São Paulo: Novatec Editora, 2010.

SILVA, Maurício Samy. **React - Aprenda Praticando: Desenvolva Aplicações Web Reais com Uso da Biblioteca React e de Seus Módulos Auxiliares**. 1. ed. São Paulo: Novatec, 2021.

SANTOS, S. **Introdução IoT: Desvendando a Internet das Coisas**. North Charleston, SC, USA: Createspace Independent Publishing Platform, 2018.

VILLAS-BOAS, Sergio Barbosa. **C/C++ e Orientação a Objetos em Ambiente Multiplataforma**. Rio de Janeiro, RJ, Brasil: [s.n.], 2001.

PEREIRA, Marcelo Robson Sousa. **A aplicação do microcontrolador ESP32 no ensino: medindo posições em função do tempo utilizando o sensor VL53L0X associado ao ESP32**. Orientador: Victor Montero Del Águila. 2020. 24 f. Trabalho de Conclusão de Curso (Especialização) - Departamento de Pós-Graduação, Universidade Federal do Amapá, Macapá, 2020. Disponível em: <http://repositorio.unifap.br:80/jspui/handle/123456789/898>. Acesso em: 20 maio. 2024.

PUHLMANN, Henrique Frank W. **Módulo de display LCD**. [S.l.]: [s.n.], 2015.

ALBERTIN, A. L.; ALBERTIN, R. M. DE M. **A internet das coisas irá muito além das coisas**. GV-executivo, v. 16, n. 2, p. 12–17, 2017.

KATZ, Marcelo; FEITOSA, Gustavo Freitas; PINTO, Ibraim Masciarelli F.; FELIX, Marcelo de Maria; BORTOLOTTTO, Luiz Aparecido. **Uso da tecnologia para engajar pacientes e otimizar a adesão terapêutica**. Revista da Sociedade de Cardiologia do Estado de São Paulo, [S.l.], v. 30, n. 3, p. 352-357, 20 outubro 2020. Disponível em: [07_revistasocesep_v30_03.pdf](#). Acesso em: 10 de março de 2024.

MASSOLA, S. C.; PINTO, G. S. **Uso da internet das coisas (IoT) a favor da saúde**. Revista Interface Tecnológica, v. 15, n. 2, p. 124–137, 2018.

AGREDA, Victor. **O que é Firebase?** Oracle, 27 de janeiro de 2023. Disponível em: <https://developer.oracle.com/pt-BR/learn/technical-articles/what-is-firebase>. Acesso em: 19 de maio de 2024.

ALVAREZ, Danilo. **Buzzer com ESP32 - Curso ESP32 básico**. Portal Vida de Silício, 6 de novembro de 2023. Disponível em: <https://portal.vidadesilicio.com.br/buzzer-com-esp32-curso-esp32-basico>. Acesso em: 23 de maio de 2024.

Sensor óptico reflexivo TCRT5000 com Arduino. Arduino e Cia. Disponível em: <https://www.arduinoecia.com.br/sensor-optico-reflexivo-tcrt5000-arduino/>. Acesso em: 18 jun. 2024.

BECKER, Lauro. **Wireframes, o que são e por que os utilizamos?** Orgânica Digital, 20 de abril de 2022. Disponível em: <https://www.organicadigital.com/blog/o-que-sao-wireframes-e-por-que-os-utilizamos/>. Acesso em: 27 de abril de 2024.

CAMPINAS, U. **Entenda o impacto e a importância do acompanhamento médico**. Disponível em: <https://www.unimedcampinas.com.br/blog/viver-com-saude/entenda-o-impacto-e-a-importancia-do-acompanhamento-medico>. Acesso em: 26 de março de 2024.

CASTRO, Giovanni. **Fita LED Endereçável com BlackBoard Uno – Tutoriais.** RoboCore, [S.d.]. Disponível em: <https://www.robocore.net/tutoriais/fita-led-enderecavel-ws2812b-blackboard-uno>. Acesso em: 17 de maio de 2024.

DEVMEDIA. **Primeiro app com React Native.** [S.d.]. Disponível em: <https://www.devmedia.com.br/primeiro-app-com-react-native/38189>. Acesso em: 04 de maio de 2024.

EUROFARMA. **A importância de tomar remédio na hora certa.** 2018. Disponível em: <https://eurofarma.com.br/artigos/a-importancia-de-tomar-remedio-na-hora-certa-inclusive-para-idosos>. Acesso em: 12 de março de 2024.

ELETROGATE. **Sensor Óptico TCRT5000 com Arduino.** Disponível em: <https://blog.eletrogate.com/sensor-optico-tcrt5000-com-arduino/>. Acesso em: 19 jun. 2024.

FERREIRA, A. P. C.; CAMPOS, E. M. P. **A Equipe de Saúde Diante do Paciente Não Aderente ao Tratamento.** Psicologia: Ciência e Profissão, v. 43, p. e244855, 20 de fevereiro de 2023. Disponível em: <https://www.scielo.br/j/pcp/a/TQtxVL3fdgXTYvhyRfXFvJp/>. Acesso em: 26 de março de 2024.

FIREBASE. **Make your app the best it can be with Firebase and generative AI.** [S.l.]. Disponível em: <https://firebase.google.com/?hl=pt-br>. Acesso em: 19 de maio de 2024.

FIREBASE. **Cloud Firestore | Armazene e sincronize dados do app em escala global.** [S.l.]. Disponível em: <https://firebase.google.com/products/firestore?hl=pt-br>. Acesso em: 18 de junho de 2024.

GADO, Wesley. **O que é NPM e como usar uma biblioteca instalada por ele.** Blog da TreinaWeb. Disponível em: <https://www.treinaweb.com.br/blog/o-que-e-npm-e-como-usar-uma-biblioteca-instalada-por-ele/>. Acesso em: 15 de maio de 2024.

GOOGLE CLOUD. **Firebase.** [S.l.]. Disponível em: <https://cloud.google.com/firestore#benefits>. Acesso em: 18 de junho de 2024.

GUIMARÃES, Fábio. **Buzzer - Como usar com o Arduino**. Mundo Projetado, 11 de agosto de 2017. Disponível em: <https://mundoprojetado.com.br/buzzer-como-usar-com-o-arduino>. Acesso em: 23 de maio de 2024.

GUIMARÃES, Fabio. **Módulo RTC DS3231**. Mundo Projetado, 16 fev. 2022. Disponível em: <https://mundoprojetado.com.br/modulo-rtc-ds3231>. Acesso em: 23 de maio de 2024.

HIGA, P.; MARQUES, A. **O que é LCD? Conheça os tipos e as vantagens dessa tecnologia**. Tecnoblog. Disponível em: <https://tecnoblog.net/responde/o-que-e-lcd/>. Acesso em: 21 de maio de 2024.

KINSTA. **O Que é Express.js? Tudo o Que Você Precisa Saber**. Disponível em: <https://kinsta.com/pt/base-de-conhecimento/o-que-e-express-js/>. Acesso em: 17 de junho de 2024.

LOCAWEB. **C++: guia sobre a linguagem de programação**. Disponível em: <https://www.locaweb.com.br/blog/temas/codigo-aberto/c-plus-plus/>. Acesso em: 26 de maio de 2024.

LOPES, Michele. **Modelagem 3D: o que é e como funciona**. Ebac Online, 29 de setembro de 2023. Disponível em: <https://ebaonline.com.br/blog/modelagem-3d-o-que-e-e-como-funciona/>. Acesso em: 26 de maio de 2024.

LOPES, Michele. **O que é Figma e como usar?** Ebac Online, 02 de outubro de 2023. Disponível em: <https://ebaonline.com.br/blog/o-que-e-figma-e-como-usar/>. Acesso em: 26 de abril de 2024.

MAKIYAMA, M. **Placa ESP32: O que é, para que serve e uso!** Disponível em: <https://victorvision.com.br/blog/placa-esp32>. Acesso em: 20 de maio de 2024.

MOURA, Beatriz; HENRIQUE, Mateus. **O que é HTML? Suas tags - Parte 1: Estrutura Básica**. Alura, 8 de fevereiro de 2024. Disponível em: <https://www.alura.com.br/artigos/o-que-e-html-suas-tags-parte-1-estrutura-basica>. Acesso em: 23 de abril de 2024.

MARCELA. **Tailwind CSS: o Guia Completo para iniciantes.** Awari. Disponível em: https://awari.com.br/tailwind-css/?utm_source=blog&utm_campaign=projeto+blog&utm_medium=Tailwind%20CSS:%20o%20guia%20completo%20para%20iniciantes. Acesso em: 12 de maio de 2024.

MARCELA. **Desenvolvimento de Aplicativos com React Native e Backend: Guia Completo para Iniciantes.** Awari. Disponível em: <https://awari.com.br/desenvolvimento-de-aplicativos-com-react-native-e-backend-guia-completo-para-iniciantes>. Acesso em: 17 de junho de 2024.

PIXEL DIAGNÓSTICOS. **A importância da comunicação entre paciente e profissional da saúde.** Disponível em: <https://pixeldiagnostico.com.br/blog/post/2020-10-10-a-importancia-da-comunicacao-entre-o-paciente-e-o-profissional-de-saude>. Acesso em: 26 de março de 2024.

PUHLMANN, H. F. W. **Módulo de Display LCD.** Disponível em: <https://embarcados.com.br/modulo-de-display-lcd>. Acesso em: 25 junho de 2024.

REACT. **Aprenda React.** Disponível em: <https://pt-br.react.dev/learn>. Acesso em: 18 de junho de 2024.

REACT NATIVE. **React Native - A framework for building native apps using React.** Disponível em: <https://reactnative.dev/>. Acesso em: 03 de maio de 2024.

RODRIGUES, Kauê. **Introdução ao figma: guia completo para iniciantes no figma.** Cubos Academy, 06 de novembro de 2023. Disponível em: <https://blog.cubos.academy/figma-para-iniciantes/>. Acesso em: 23 de abril de 2024.

STROSCHON, Gustavo Rodolfo. Como Utilizar Led Endereçável? **Usinainfo**, 22 de janeiro de 2020. Disponível em: <https://www.usinainfo.com.br/blog/led-enderecavel-como-utilizar/>. Acesso em: 17 de maio de 2024.

TAILWIND CSS. **Tailwind CSS - Rapidly build modern websites without ever leaving your HTML.** Disponível em: <https://tailwindcss.com/>. Acesso em: 12 de maio de 2024.

O que é wireframe? **Lucidchart**. Disponível em:

<https://www.lucidchart.com/pages/pt/o-que-e-wireframe>. Acesso em: 27 de abril de 2024.

Wireframes: quais os tipos e as principais ferramentas de criação. **Rockcontent**, 26 de fevereiro de 2019. Disponível em:

<https://rockcontent.com/br/blog/wireframes/>. Acesso em: 27 de abril de 2024.