CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

ETEC DA ZONA LESTE

Mtec Desenvolvimento de Sistemas AMS

Alex Expedito Silva Santos

Danilo Santos Soares

Endrigo Gustavo Brandão de Oliveira

DAELINK:
Vaga de Emprego para PCD

São Paulo

2024

Alex Expedito Silva Santos

Danilo Santos Soares

Endrigo Gustavo Brandão de Oliveira

DAELINK:
Vaga de Emprego para PCD

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Desenvolvimento de Sistemas da Etec Zona Leste, orientado pelo Prof. Jeferson Roberto de Lima, como requisito parcial para conclusão de curso integrado ao ensino médio.

São Paulo

2024

Dedicatória

Dedicamos este trabalho a todos que se sentiam marginalizados, e excluídos devido a algum tipo de deficiência. Em que desenvolvemos apenas um método para que mostrem suas habilidades profissionais, e possam se sentir integralmente da sociedade.

Agradecimentos

Nós, DaeLink, gostaríamos de expressar nossa gratidão a todos que contribuíram para a construção deste projeto para que pudesse ter gerado frutos, nos foi dado as contribuições diretas e indiretas de pessoas, o que nos levou a ter forças para continuar a criação deste projeto, principalmente aos que acreditaram em nossa capacidade. Cada um teve um papel essencial no nosso processo e na concretização de nossa meta.

Primeiramente iremos dar o devido reconhecimento a cada membro da equipe, que demonstraram comprometimento, criatividade e colaboração ao longo do projeto. O empenho de cada um fez toda a diferença.

Agradecemos aos nossos professores que nos instruíram e ensinaram, maneiras e conhecimento para que pudéssemos ter feito até o final, desta maneira nos foi passado a confiança necessária para a construção de um projeto que tem grande potencial para a sociedade.

Também queremos ressaltar da importância de nossos pais que nos motivaram e a ajudaram da melhor maneira possível até o devido momento, compartilhando maneiras para a resolução de problemas da sua maneira.

Nosso sincero obrigado a todos. Este projeto é fruto de um trabalho conjunto, e somos extremamente gratos por cada contribuição recebida.

RESUMO

Este trabalho tem como objetivo tratar sobre a inserção de pessoas com deficiência (PCDs) ao setor trabalhista através de um sistema baseado em plataformas digitais que promovem conectividade. As cotas para PCDs frequentemente não são preenchidas corretamente, devido à falta de busca das empresas junto de fatores preconceituosos. Embora existam ações que estabeleçam cotas, a inclusão dessas pessoas se depara com desafios significativos, gerando menores taxas de participação no campo de trabalho, ao se comparar com pessoas sem deficiência. Desta maneira o projeto consiste no estudo de desenvolver, e facilitar a inclusão profissional destas pessoas. A metodologia empregada inclui a análise das necessidades de PCDs e empresas, e através dela moldando o projeto para se adequar a elas. Os resultados indicam a criação de um protótipo, composto por site, e aplicativo que pode ser utilizado pelas empresas para preencher estas vagas de forma facilitada. A sua construção é pensada essencialmente para empresas, consistindo em um site e aplicativo que mostram os principais candidatos para vagas remanescentes de determinadas áreas. Portanto, demonstra-se que a plataforma pode ampliar o preenchimento de vagas.

Palavras-Chaves: Inclusão; PCD; Mercado de trabalho; Plataformas digitais; Ferramentas

ABSTRACT

This work aims to deal with the inclusion of people with special needs (PWDs) in the labor sector through a system based on digital platforms that promote connectivity. Quotas for them are often not filled due to a lack of demand from companies and prejudice. Although there are actions to establish quotas for People with special needs, inclusion faces significant challenges, resulting in lower participation rates in the labor market compared to people without special needs; in this way, the project consists of the study of developing a digital platform that promotes professional inclusion to facilitate the integration of PWN into the job market. The theoretical support used to include the analysis of the needs of PWN and companies. Results indicate that a prototype system, consisting of a website and an application, can be created that companies can use to fill the remaining vacancies within their institutions. Its construction is designed for companies that highlight the leading candidates for remaining vacancies in certain areas. Therefore, the project can be qualified to enhance the inclusion of PWN in the Labor Market.

Key words: Inclusion; PWD; Labor Market; Digital platform; Tools

Lista de Figuras

Figura 1 – Exemplo Html 19
Figura 2 – Exemplo HTML 2 20
Figura 3 – Exemplo Formulario 21
Figura 4 – Exemplo tag “link” 22
Figura 5 – Exemplo CSS parte 1 23
Figura 6 – Exemplo CSS parte 2 24
Figura 7 – Exemplo CSS parte 3 25
Figura 8 – Exemplo Formulário Estilizado 25
Figura 9 – Exemplo de formulário com javascript 26
Figura 10 – Exemplo alerta de teste 27
Figura 11 – Exemplo javascript sucesso 28
Figura 12 – Exemplo Javascript CANCELAR 28
Figura 13 – Exemplo Tailwind Css 29
Figura 14 – Tailwind formulário 29
Figura 15 – Exemplo Vite 30
Figura 16 – React Vite exemplo 31
Figura 17 – Exemplo React 31
Figura 18 – React Formulario 32
Figura 19 – React Native Exemplo 33
Figura 20 – React Native 2 34
Figura 21 – Exemplo Expo 35
Figura 22 – Exemplo Node 36
Figura 23 – Exemplo teste de senha Python 38
Figura 24 – Exemplo Python com importe 40
Figura 25 – Exemplo de instalação de pacotes 41
Figura 26 – Exemplo de emoji com Python 42
Figura 27 – Exemplo Jupyter 43
Figura 28 – Exemplo Pandas 44
Figura 29 – Exemplo Sckit Learn 45
Figura 30 – Exemplo Der 46
Figura 31 – Exemplo Mer 47
Figura 32 – Exemplo de banco de dados com o Firebase 48

Figura 33 – Exemplo de Diagrama de Caso de Uso 49
Figura 34 – Exemplo Documentação Caso de Uso 50
Figura 35 – Exemplo Diagrama de Atividade 52
Figura 36 – Exemplo Diagrama de Sequência 54
Figura 37 – Exemplo de Diagrama de Máquina de Estado 56
Figura 38 – Exemplo Wireframe de Baixa Fidelidade 58
Figura 39 – Exemplo Wireframe Baixa 2 58
Figura 40 – Diagrama de Caso de Uso Geral do DaeLink 63
Figura 41 – Diagrama de Atividade: Manter Vagas 73
Figura 42 – Diagrama de Atividade: Gerenciar empresa 73
Figura 43 – Diagrama de Atividade: Gerenciar apuração 74
Figura 44 – Diagrama de Atividade: Gerenciar Documento 75
Figura 45 – Diagrama de Atividade: Gerenciar mensagens 75
Figura 46 – Diagrama de Atividade: Visualizar Apuração 76
Figura 47 – Diagrama de Atividade: Gerenciar vagas Ingressadas 76
Figura 48 – Diagrama de Atividade: Visualizar Vaga 77
Figura 49 – Diagrama de Atividade: Manter PCD 78
Figura 50 – Diagrama de Atividade: Visualizar PCD 79
Figura 51 – Diagrama de Atividade: Relacionar PCD a Vaga 79
Figura 52 – Diagrama de Atividade: Fazer Login 80
Figura 53 – Diagrama de Atividade: Fazer cadastro 81
Figura 54 – Diagrama de Máquina de Estado: Manter Vagas 82
Figura 55 – Diagrama de Máquina de Estado: Gerenciar Empresa 83
Figura 56 – Diagrama de Máquina de Estado: Gerenciar Apuração 84
Figura 57 – Diagrama de Máquina de Estado: Manter Documentos 85
Figura 58 – Diagrama de Máquina de Estado: Gerenciar Mensagens 86
Figura 59 – Diagrama de Máquina de Estado: Visualizar Apuração 86
Figura 60 – Diagrama de Máquina de Estado: Gerenciar Vagas Ingressadas 87
Figura 61 – Diagrama de Máquina de Estado: Visualizar Vaga 87
Figura 62 – Diagrama de Máquina de Estado: Manter PCD 88
Figura 63 – Diagrama de Máquina de Estado: Fazer Login 89
Figura 64 – Diagrama de Máquina de Estado: Fazer cadastro 89
Figura 65 – Diagrama de Máquina de Estado: Relacionar PCD a vaga 90

Figura 66 – Diagrama de Máquina de Estado: Visualizar PCD 90
Figura 67 – Diagrama de Sequência: Manter Vaga 91
Figura 68 – Diagrama de Sequência: Gerenciar Empresa 92
Figura 69 – Diagrama de Sequência: Criar Empresa 93
Figura 70 – Diagrama de Sequência: Editar empresa 94
Figura 71 – Diagrama de Sequência: Gerenciar apuração 95
Figura 72 – Diagrama de Sequência: Gerenciar documento 95
Figura 73 – Diagrama de Sequência: Gerenciar mensagens 96
Figura 74 – Diagrama de Sequência: Gerenciar Visualizar PCD 97
Figura 75 – Diagrama de Sequência: Relacionar Vaga 97
Figura 76 – Diagrama de Sequência: Visualizar vaga 98
Figura 77 – Diagrama de Sequência: Gerenciar vagas ingressadas 99
Figura 78 – Diagrama de Sequência: Manter documento 100
Figura 79 – Diagrama de Sequência: Manter PCD 101
Figura 80 – Diagrama de Sequência: Editar PCD 102
Figura 81 – Diagrama de Sequência: Excluir PCD 102
Figura 82 – Diagrama de Sequência: Visualizar apuração 103
Figura 83 – Diagrama de Sequência: Gerenciar mensagens 104
Figura 84 – Diagrama de Sequência: Manter PCD 105
Figura 85 – Diagrama de Sequência: Editar PCD 105
Figura 86 – Diagrama de Sequência: Deletar PCD 106
Figura 87 – Diagrama de Sequência: Visualizar PCD 106
Figura 88 – Diagrama de Sequência: Gerenciar empresa 107
Figura 89 – Diagrama de Sequência: Editar Empresa 108
Figura 90 – Diagrama de Sequência: Consultar Empresa 108
Figura 91 – Wireframe de baixa fidelidade: Home do PCD 110
Figura 92 – Wireframe de alta fidelidade: Home do PCD 111
Figura 93 – Wireframe de baixa fidelidade: Login 112
Figura 94 – Wireframe de alta fidelidade: Login PCD 113
Figura 95 – Wireframe de baixa fidelidade: Seleção de cadastro 114
Figura 96 – Wirefreme de alta fidelidade: Seleção de criação de conta 114
Figura 97 – Wireframe de baixa fidelidade: Realização de cadastro 115
Figura 98 – Wireframe de alta fidelidade: Realização de cadastro 116
Figura 99 – Wireframe de baixa fidelidade: visualizar vagas disponíveis 116

Figura 100 – Wireframe de alta visualizar vagas disponíveis 117
Figura 101 – Wireframe de baixa entrar vaga 118
Figura 102 – Wireframe de alta fidelidade Entrar em Vagas 119
Figura 103 – Wireframe de baixa fidelidade: Visualizar vagas já candidatada 120
Figura 104 – Wireframe de alta fidelidade: visualizar processos realizados. 121
Figura 105 – Wireframe de baixa fidelidade: enviar documento do PCD para empresa 121
Figura 106 – Wireframe de alta fidelidade: enviar documento do PCD para empresa 122
Figura 107– Wireframe de baixa fidelidade: Atualizar documento 122
Figura 108 – Wireframe de alta fidelidade: Atualizar documento 123
Figura 109 – Wireframe de baixa fidelidade: Chat parte Portador de deficiência 124
Figura 110 – Wireframe de Alta fidelidade: Chat PCD 125
Figura 111 – Wireframe de baixa fidelidade: Visualização de perfil do PCD 126
Figura 112 – Wireframe de alta fidelidade: Visualização de perfil do PCD ... 126
Figura 113 – Wireframe baixa fidelidade: Editar Perfil PCD 127
Figura 114 – Wireframe de alta fidelidade: Editar perfil do PCD 128
Figura 115 – Wireframe de baixa fidelidade: criar vaga 129
Figura 116 – Wireframe de alta fidelidade: criar vaga 129
Figura 117 – Wireframe de baixa fidelidade: visualizar candidatos para a vaga 130
Figura 118 – Wireframe de alta fidelidade: visualizar candidatos para a vaga 131
Figura 119 – Wireframe de baixa fidelidade: Visualizar perfil PCD 132
Figura 120 – Wireframe de alta fidelidade: Visualizar perfil PCD 132
Figura 121 – Wireframe de baixa fidelidade: processo da empresa 133
Figura 122 – Wireframe de alta fidelidade: processo da empresa 134
Figura 123 – Wireframe de baixa fidelidade: Edição de vaga 135
Figura 124 – Wireframe de alta fidelidade: Edição de vagas 136
Figura 125 – Wireframe de baixa fidelidade: Visualizar candidatos que aceitaram a vaga 137

Figura 126 –Wireframe de alta fidelidade: Visualizar candidatos que aceitaram a vaga.....138
Figura 127 – Wireframe de baixa fidelidade: perfil empresa 138
Figura 128 – Wireframe de alta fidelidade: perfil empresa 139
Figura 129 – Wireframe de baixa fidelidade: Edição de perfil empresa.....140
Figura 130 – Wireframe de alta fidelidade: Editar perfil empresa.....141
Figura 131 – Wireframe de baixa: Home-APP 143
Figura 132 – Wireframe de alta fidelidade: Home-APP.....144
Figura 133 – Wireframe de baixa fidelidade: Início-APP 145
Figura 134 – Wireframe de alta fidelidade: Início-APP.....146
Figura 136 – Wireframe de Alta fidelidade: Login 148
Figura 137 – Wireframe de baixa fidelidade: Vagas Disponíveis 149
Figura 138 – Wireframe de Alta: Visualizar vagas 150
Figura 139 – Wireframe de baixa fidelidade: Vaga-APP 151
Figura 140 – Wireframe de alta fidelidade: Vaga-APP 152
Figura 141– Wireframe de baixa fidelidade: Candidaturas do PCD 153
Figura 142 – Wireframe de Alta fidelidade: Candidaturas já realizadas 154
Figura 143 – Wireframe de baixa fidelidade: Enviar documento 155
Figura 144 – Wireframe de Alta fidelidade: Enviar documento.....156
Figura 145 – Wireframe de baixa fidelidade: Documento-APP 157
Figura 146 – Wireframe de alta fidelidade: Documento-APP 158
Figura 147 – Wireframe de baixa fidelidade: Chat 159
Figura 148 – Wireframe de Alta fidelidade: Chat.....160
Figura 149– Wireframe de baixa fidelidade: Perfil Portador deficiência 160
Figura 150 – Wireframe de Alta fidelidade: Perfil PCD.....162
Figura 151 – Wireframe de baixa fidelidade: Editar documento 162
Figura 152 – Wireframe de Alta fidelidade: Editar documento 165
Figura 153 – Wireframe de baixa fidelidade: Load.....165
Figura 154 – Wireframe de Alta fidelidade: Load 167

Lista de tabelas

Tabela 1 – Descrição do caso de uso: Manter vagas.....64
Tabela 2 – Descrição do caso de uso: Gerenciar Empresa 64
Tabela 3 – Descrição do caso de uso: Gerenciar Apuração 65
Tabela 4 – Descrição do caso de uso: Gerenciar documento 66
Tabela 5 – Descrição de Caso de uso: Gerenciar Mensagens 67
Tabela 6 – Descrição do caso de uso: Visualizar Apuração.....67
Tabela 7 – Descrição do caso de uso: Gerenciar vagas ingressadas 68
Tabela 8 – Descrição do caso de uso: Visualizar vagas 68
Tabela 9 – Descrição do caso de uso: Manter PCD 69
Tabela 10 – Descrição do Caso de uso: Visualizar PCD 70
Tabela 11 – Descrição do caso de uso: Fazer login 70
Tabela 12 – Descrição do caso de uso: Relacionar PCD a vaga 72

Lista de abreviaturas e siglas

Cross-Origin Resource Sharing (CORS)

Cascading Style Sheets (CSS)

Diagrama Entidade Relacionamento (DER)

Document Object Model (DOM)

Chave Estrangeira (FK)

HyperText Markup Language (HTML)

Hypertext Transfer Protocol (HTTP)

JavaScript (JS)

JavaScript Object Notation (JSON)

Modelo Entidade Relacionamento (MER)

Não apenas SQL (NoSQL)

Node Package Manager (NPM)

Node Package Executor (NPX)

Pessoa Com Deficiência (PCD)

Chave Primaria (PK)

Sistema de Gerenciamento de Banco de Dados (SGBD)

Linguagem de Consulta Estruturada (SQL)

Unified Modeling Language (UML)

SUMÁRIO

1 INTRODUÇÃO 16
2 REFERENCIAL TEÓRICO 17
2.1 Problemas encontrados no mercado de Trabalho 17
2.2 Html 18
2.3 CSS 22
2.4 JavaScript 26
2.5 TailwindCSS 28
2.6 React 30
2.6.1 Vite 30
2.6.2 Axios 32
2.7 React Native 32
2.8 Expo 34
2.9 Node.JS 35
2.9.1 Express 37
2.9.2 Cors 37
2.9.3 Node Package Manager 37
2.9.4 Node Package Executor 37
2.10 Python 38
2.11 Flask 42
2.12 Jupyter 43
2.13 Machine Learning 43
2.14 Numpy 43
2.15 Pandas 44
2.16 Scikit-Learn 45
2.17 Banco De Dados 46
2.17.1 Diagrama Entidade Relacionamento 46

2.17.2 Modelo Entidade Relacionamento 47
2.17.3 Banco De Dados Não Relacional 47
2.18 Firebase 47
2.19 UML 48
2.19.1 Caso De Uso 48
2.19.2 Documentação de Caso de Uso 50
2.19.3 Diagrama de Atividade 52
2.19.4 Diagrama de Sequência 54
2.19.5 Diagrama de Máquina de Estado 56
2.20 Wireframe 57
2.20.1 Design Centrado No Usuário (DCU) 59
2.20.2 Experiência Do Usuário (UX) 59
3 DESENVOLVIMENTO 60
3.1 Diagrama de caso de uso 60
3.1.1 Diagrama de caso de uso 60
3.1.2 Diagrama de atividade 72
3.1.3 Diagrama de máquina de estado 82
3.1.4 Diagrama de sequência 91
3.2 Prototipação do sistema 109
3.2.1 Wireframes DaeLink Visão PCD 109
3.2.2 Wireframes DaeLink Visão Empresa 128
REFERÊNCIAS 170

1 INTRODUÇÃO

A integração de indivíduos com necessidades especiais (PCD) no mercado de trabalho através de plataformas de conectividade digital representa uma preocupação significativa e premente. Apesar da existência de numerosas iniciativas sociais destinadas a adaptação dos PCD na sociedade em geral, a quantidade de PCD empregados em empresas continua a ser relativamente baixo, como evidenciado pelo cumprimento insuficiente das quotas estabelecidas. Esta ausência persistente dos PCD no mercado de trabalho sublinha a necessidade contínua de medidas mais robustas para facilitar a sua inclusão (G1, 2019). Assim, este estudo tem como foco o desenvolvimento de um sistema que facilite essa conectividade de maneira mais eficiente, por meio de tecnologias web e aplicativos móveis, visando a integração mais efetivas entre empresas e os seus PCDs. Neste contexto, pessoas com deficiência de 14 anos ou mais apresentam taxas de participação no mercado de trabalho é significativamente menor (23,8%) e de formalização (34,3%) menores do que as pessoas sem deficiência, cujas taxas são de 66,3% e 50,9% (IBGE, 2022). Um dos principais fatores que limitam o acesso PCD ao mercado de trabalho é o preconceito. Estudos indicam que muitas empresas ainda não sentem confortáveis para contratar pessoas com deficiência, devido à falta de conhecimento sobre as capacidades desses profissionais (CNN Brasil, 2021).

Torna-se, portanto, identificar soluções que facilitem a integração entre as empresas e os PCDs, aumentando as oportunidades e a contratação desses profissionais através da utilização de novas tecnologias, com o objetivo final de promover uma sociedade mais inclusiva. Diante dessas considerações, torna-se pertinente indagar por que as cotas de PCD no mercado de trabalho continuam não sendo cumpridas e em que medida uma plataforma digital de profissionais pode facilitar a integração desses indivíduos nas empresas. Sendo assim a hipótese formulada é que o uso de um sistema digital voltado especificamente para a conexão entre empresas e PCD pode aumentar a taxa de preenchimento no setor de trabalho, contribuindo o processo de recrutamento e superando as barreiras atuais, como o preconceito. Para alcançar esse objetivo, este estudo é desenvolver uma plataforma digital que tenha como foco a integração de pessoas com necessidades especiais no setor de trabalho que irá promover um ambiente de trabalho mais inclusivo e aumento de preenchimento de vagas das cotas.

Foram estabelecidos como a primeira etapa uma revisão bibliográfica de inserção de pessoas com necessidades especiais no mercado de trabalho, com destaque para estudos de inclusão e ferramentas de inclusão para verificar como as plataformas digitais podem facilitar a contratação e preenchimento de vagas remanescentes. Com procedeu-se ao emprego de método quantitativo e estudo de caso para a coleta de todos os dados e análise de problemas específicos, empregando métodos indutivos e dedutivos, de acordo com a abordagem de Lakatos e Marconi (2017) junto de PEREIRA et al(2018) e GIL(2002). A pesquisa abordará vários autores seminais no campo, incluindo CNN (2021), que analisa a importância do uso da tecnologia para melhorar a inclusão. Os autores que serão referenciados ao longo do artigo para dar suporte teórico ao desenvolvimento da plataforma e aos seus potenciais soluções para o obstáculo da inclusão. Sendo o desenvolvimento dividido em três partes: Web, a aplicação móvel e o sistema de recomendação. O React Vite foi selecionado para o desenvolvimento web devido às suas capacidades simples de criação e processamento de componentes (SCHMITZ; GEORGII, 2015). O React Native foi utilizado para o Desenvolvimento da aplicação móvel devido à sua compatibilidade nativa e multiplataforma com Android e iOS (ESCUDELARIO; PINHO. 2020). O sistema de recomendação utilizou Python por seu extenso conjunto de ferramentas, incluindo Scikit-Learn para aprendizado de máquina (MENEZES, 2014).

2 REFERENCIAL TEÓRICO

Esse capítulo possuí como meta, abstrair todas as etapas de fundamentação teórica para a construção do projeto junto da apresentação de conceitos e tecnologias. Visando demonstrar todo o embasamento teórico da plataforma DAELink.

2.1 PROBLEMAS ENCONTRADOS NO MERCADO DE TRABALHO

De acordo com G1 (2023) cerca de 18,6 milhões de brasileiros com dois anos ou mais possuem algum tipo de deficiência, sendo a inclusão um desafio que persiste no Brasil devido à falta de acessibilidade e apoio adequado. Dados do Instituto Brasileiro de Geografia e Estatística (IBGE, 2022) comprova que essas pessoas enfrentam maiores dificuldades para entrar no mercado de trabalho cerca de (28,3%) deficientes e (66,3%) sem deficiência.

O Brasil estabelece que empresas com cem empregados ou mais são obrigadas a preencher de 2% a 5% com pessoas com deficiência, conhecida como Lei de Cotas, conforme o artigo 1º da Lei n º 8.213, de 24 de julho de 1991:

> Art. 1º A Previdência Social, mediante contribuição, tem por fim assegurar aos seus beneficiários meios indispensáveis de manutenção, por motivo de incapacidade, desemprego involuntário, idade avançada, tempo de serviço, encargos familiares e prisão ou morte daqueles de quem dependiam economicamente. (BRASIL, 1991.).

Apesar de estar em vigor há quase trinta anos, segundo UNICAMP (2020) ainda não é plenamente cumprida, ainda existe desafios para que a inclusão aconteça devido a pouca especificidade da legislação e a qualificação.

Para G1 (2017) a tecnologia está cada vez mais presente no cotidiano, muitas empresas não atendem a essa necessidade. Isso ocorre devido à falta de um sistema acessíveis, gerando uma lacuna na conexão de pessoas com deficiência. Portanto, há uma necessidade de criação de um sistema que facilite essa integração digital.

Uma solução de uma plataforma digital de conectividade pode promover a inclusão social e melhorar o acesso dessas pessoas ao mercado de trabalho.

2.2 HTML

A linguagem de marcação de hipertexto conhecida como *HTML*, sendo necessária para qualquer documento utilizado na *web* no quesito de sites utilizado para estruturar todo o conteúdo de uma página (SILVA, 2015).

Baseado em demarcações denominadas “*tags*”, que podem se referir a qualquer elemento presente na página, sendo interpretados pelo *browser,* mostrando sua representação visual (DUCKETT, 2016).

Figura 1 – Exemplo Html

```html
<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulário HTML</title>
</head>

> <body>···
</body>

</html>
```

Fonte: Autoria Própria, 2024.

Na figura acima, demonstra a estrutura básica para a criação de uma página *web* simples, e as principais *tags* que compõem essa estrutura são:

- “DOCTYPE *Html*” :O *Document Type Declaration* é utilizado para informar a versão do *HTML* que, consequentemente, ajuda o navegador a renderizar as informações.
- *<html>*:Esta *tag* é utilizada para definir o esqueleto do site, englobando todos os conteúdos de uma página na internet.
- <head>: É usado para a inclusão de recursos externos ao principal, como por exemplo *links*.
- <title>: Utilizado para definir o título da página, exibido na aba do navegador.
- <body>: A parte principal de um site, entre essas *tag* ficam os conteúdos carregados pelo navegador.
- <atributos>: São utilizados para definição de um comportamento na estrutura da página, como exemplo o atributo “*Class*” utilizado para definir um conjunto de elementos pertencentes a uma classe específica.

A imagem 2, demonstra o processo da criação de um formulário simples:

Figura 2 – Exemplo HTML 2

```html
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulário HTML</title>
</head>

<body>
    <div id="formstyle">
        <h2>Formulário de Contato</h2>

        <form action="#" method="POST" id="myForm">
            <label for="name">Nome</label><br>
            <input type="text" name="name" id="name" required> <br><br>


            <label for="name">Email</label><br>
            <input type="text" name="email" id="email" required> <br><br>

            <label for="name">Telefone</label><br>
            <input type="text" name="Telefone" id="Telefone" required> <br><br>


            <label for="name">Idade</label><br>
            <input type="date" name="Idade" id="Idade" required> <br><br>

            <label for="name">ETEC</label><br>
            <input type="text" name="ETEC" id="ETEC" required> <br><br>

            <label for="name">Curso</label><br>
            <input type="text" name="Curso" id="Curso" required> <br><br>

            <button type="submit">Enviar</button>
        </form>
    </div>
</body>
</html>
```

Fonte: Autoria Própria, 2024.

Para a criação de um formulário, foi utilizado algumas tags de estruturação, sendo elas:

- <h2> Utilizado para definir o tamanho de um texto em uma página *web*, podendo ir de h1 que é o maior, até h6 que é o menor.
- <form> É uma *tag* utilizada para a criação de formulários com intuito de extrair dados, como exemplo o cadastro de uma pessoa, onde é extraído nome, *email*, idade e senha.
- *<label>* É utilizada para definir um rótulo descritivo a um elemento de uma página *web.*
- *<Input>* É a *tag* principal para a utilização de formulários, ela é a caixa de texto que fica dentro dos formulários.
- <button> Utilizado para executar alguma ação dentro do sistema, como visto na imagem abaixo, a função dele vai ser “Cadastrar”.

O resultado desse código pode ser visto na figura 3 a seguir:

Figura 3 – Exemplo Formulario

Fonte: Autoria Própria, 2024.

2.3 CSS

A folha de estilo em cascata, o CSS é uma linguagem concebida para a estilização de páginas *web* mediante atributos pré-estabelecidos que se relacionam às características visuais (SCHEIDT, 2015).

Em que através das demarcações feitas pelo HTML, essas alterações estilísticas podem ocorrer, o restringindo apenas as marcações. (SILVA, 2008).

Figura 4 – Exemplo tag “link”

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Formulário HTML</title>
    <link href="css.css" rel="stylesheet">
</head>
```

Fonte: Autoria Própria, 2024.

Neste exemplo, é demonstrado como adicionar um código CSS externo ao arquivo Html.

- *<link> Tag* utilizada para vincular propriedades de outro arquivo, através do “
- <href> (que é seguido do local armazenado do arquivo) e o *rel* para se definir o tipo de relacionamento.

Figura 5 – Exemplo CSS parte 1

```
index.html    # css.css
# css.css > form
1   * {
2       padding: 0;
3       margin: 0;
4       font-size: 14px;
5       line-height: 1.5;
6   }
7
8   html,body{
9       overflow-x: hidden;
10  }
11
12  body {
13      font-family: system-ui, -apple-system;
14      color: #c9d1d9;
15      display: flex;
16      flex-direction: column;
17      align-items: center;
18  }
19
20  #formstyle {
21      display: flex;
22      flex-direction: column;
23      gap: 16px;
24      margin-top: 30px;
25  }
26
27  #title {
28      font-size: 24px;
29      letter-spacing: -0.5px;
30      text-align: center;
31      color: #0d1117;
32  }
```

Fonte: Autoria Própria, 2024.

Na figura 6, através da anexação de atributos às classes ou via *id* se restringindo a uma unidade da respectiva classe *("#formstyle"*). É feita a adição de atributos que afetam toda a página (*html, body,* *), ou elementos específicos trocando fonte, cores, posicionamento horizontal e vertical e de outros detalhes gráficos do formulário.

Figura 6 – Exemplo CSS parte 2

```
index.html    # css.css
# css.css > form
form {
    display: flex;
    flex-direction: column;
    gap: 14px;
    background-color: #e7e7e7;
    border-radius: 6px;
    padding: 40px;
    border: 1px solid #21262d;
}

.input-field {
    display: flex;
    flex-direction: column;
    gap: 8px;
}

.input-field div {
    display: flex;
    justify-content: space-between;
}

.input-field label {
    font-weight: bold;
    color: #0d1117;

}

.form-control {
    all: unset;
    padding: 5px 12px;
    line-height: 20px;
    color: #c9d1d9;
    vertical-align: middle;
    background-color: #0d1117;
    border: 1px solid #30363d;
    border-radius: 6px;
    box-shadow: none;
}
```

Fonte: Autoria Própria, 2024.

Continuando no código mudamos através das classes (‘.’ Previamente da classe), ou acessando diretamente as tags a mudando em toda a página.

Figura 7 – Exemplo CSS parte 3

```
73  .form-control:focus-visible, .form-select:focus-visible {
74      border-color: #58a6ff;
75      outline: none;
76  }
77
78  button {
79      background-color: #233a86;
80      color: #ffffff;
81      border: 1px solid rgba(240,246,252,0.1);
82      display: inline-block;
83      padding: 5px 16px;
84      font-weight: 500;
85      line-height: 20px;
86      cursor: pointer;
87      border-radius: 6px;
88      transition: background-color 80ms;
89  }
90
91  button:hover{
92      background-color: #2e98a0;
93  }
```

Fonte: Autoria Própria, 2024.

Ao final do código *CSS*, todos os componentes passam pelo processo de indexação de atributos. Assim realizando a estilização do formulário e dos outros elementos que compõe uma página *web*.

Figura 8 – Exemplo Formulário Estilizado

Fonte: Autoria Própria, 2024.

2.4 JavaScript

Originado pela Netscape, e sendo documentado pela Sun Microsystems (Oracle), o Javascript é popularmente conhecido como uma linguagem de programação da *web* tornando-se específica para comportamentos do *HTML* e *CSS* (FLANAGAN ,2012).

Para Groner (2019) a linguagem Javascript é extremamente eficaz, se tornando uma das linguagens mais populares mundialmente, passando a ser usada tanto para *front-end* quanto *back-end* resultando em uma linguagem multifuncional.

Na Figura 9 veremos o funcionamento do Javascript dentro do formulário mostrado anteriormente, exemplificando o envio de componentes digitados dentro das caixas de textos.

Figura 9 – Exemplo de formulário com javascript

```
cadastro.html > html > body
2   <html lang="en">
8   <body>
10      <form id="myForm" action="#" method="post">
17          <label for="Telefone">Telefone</label><br>
18          <input type="text" id="Telefone" name="Telefone" required><br><br>
19
20          <label for="idade">Idade</label><br>
21          <input type="date" id="idade" name="idade" required><br><br>
22
23          <label for="Escola">ETEC</label><br>
24          <input type="text" id="escola" name="escola" required><br><br>
25
26          <label for="Curso">Curso:</label><br>
27          <input type="number" id="curso" name="curso" required><br><br>
28
29          <button type="submit">Enviar</button>
30      </form>
31
32      <script>
33          document.getElementById('myForm').onsubmit = function(event) {
34              event.preventDefault();
35              if (confirm('Quer testar o formulário? Se sim click em ok! ')) {
36
37                  alert('Formulário enviado com sucesso!');
38                  this.submit();
39              } else {
40                  alert('Envio cancelado.');
41              }
42          };
43      </script>
44  </body>
45  </html>
```

Fonte: Autoria Própria, 2024.

Depois do </form> deve-se implementar, a *tag* <Script> possui como função incorporar ou referenciar código Javascript em um documento HTML.

- *Document*: Utilizado como forma de representação do documento HTML carregado no navegador.
- *GetElementById*: É reconhecido como um método do objeto '*document*' que retorna o elemento que possui um atributo ID, no caso '*myForm*'.
- *Onsubmit*: É um evento que ocorre quando um formulário é enviado. Acompanhado com uma função a este evento permitindo que seja executado o código quando o usuário tentar enviar.
- *Function(event):* Possui como função sempre ser chamada quando se utilizar o evento '*onsubmit*'. Ela recebe um argumento 'Event', que é um objeto contendo detalhes sobre o evento de submissão.
- *Confirm* ('quer testar o formulário? Se sim, clique em ok!'): '*confirm*' é uma função que exibe uma caixa de diálogo com uma mensagem e dois botões: OK e CANCELAR. Retorna '*true*' se o usuário clicar em 'OK' e '*FALSE*' se clicar em cancelar. Isso é usado para confirmar como o usuário proceder com o envio do formulário.
- Alert: possui como objetivo exibir uma caixa de diálogo com uma mensagem para usuário. Neste caso, é usado para informar que o formulário foi enviado com sucesso, já o outro exemplo informa ao usuário que o envio foi formulário foi cancelado.

Contudo, ao enviar o resultado, o *alert* deverá ser apresentado demonstrando as duas opções, testar o formulário ou cancelar.

Figura 10 – Exemplo alerta de teste

Fonte: Autoria Própria, 2024.

Na figura 11 a seguir, aparecerá o resultado se for escolhida a opção OK, demonstrando o sucesso do envio do formulário.

Figura 11 – Exemplo javascript sucesso

Fonte: Autoria Própria, 2024.

Caso seja escolhida a opção CANCELAR, deverá aparecer o envio cancelado do formulário.

Figura 12 – Exemplo Javascript CANCELAR

Fonte: Autoria Própria, 2024.

2.5 TailwindCSS

O framework de estilo CSS código aberto, cujo é denominado Tailwind tem o intuito de oferecer uma abordagem única para projetos *web*, dentro do código HTML, gerando estilo eficaz sem a precisão de escrever comando CSS. (CLASEN, 2023).

O tailwind tem como objetivo eliminar a necessidade do uso de CSS personalizado, através de biblioteca de classes pré-definidas para estilos de elementos específicos, como cores, tipografia e muitos outros. (MATOS, 2023; KLIMM, 2021).

Na figura 13, será demonstrado a realização de um código utilizando tailwind CSS em um formulário semelhante ao mostrado anteriormente.

Figura 13 – Exemplo Tailwind Css

```
<html lang="pt-br">
<body class="h-screen flex items-center justify-center bg-black">
    <div class="w-full max-w-md p-8 space-y-4 bg-white rounded-lg shadow">
        <form action="" class="space-y-6">
            <h1 class="text-3xl font-extrabold tracking-tight text-gray-900">Formulário de Contato</h1>
            <div>
                <label for="nome" class="block text-sm font-medium text-gray-700">Nome:</label>
                <input class="w-full border border-gray-300 rounded-md px-5 py-3 focus:outline-none focus:ring-pink-500 focus:border-pink-500" id="nome" type="text"
                placeholder="Digite seu Nome">
            </div>
            <div>
                <label for="tel" class="block text-sm font-medium text-gray-700">Telefone:</label>
                <input class="w-full border border-gray-300 rounded-md px-5 py-3 focus:outline-none focus:ring-pink-500 focus:border-pink-500" id="tel" type="Number"
                placeholder="Digite seu Telefone">
            </div>
            <div>
                <label for="etec" class="block text-sm font-medium text-gray-700">Etec:</label>
                <input class="w-full border border-gray-300 rounded-md px-5 py-3 focus:outline-none focus:ring-pink-500 focus:border-pink-500" id="etec" type="text"
                placeholder="Digite sua Etec">
            </div>
            <div>
                <label for="idade" class="block text-sm font-medium text-gray-700">Idade:</label>
                <input class="w-full border border-gray-300 rounded-md px-5 py-3 focus:outline-none focus:ring-pink-500 focus:border-pink-500" id="idade" type="date"
                placeholder="Digite sua Idade">
            </div>
            <div>
                <label for="curso" class="block text-sm font-medium text-gray-700">Curso:</label>
                <input class="w-full border border-gray-300 rounded-md px-5 py-3 focus:outline-none focus:ring-pink-500 focus:border-pink-500" id="curso" type="text"
                placeholder="Digite seu Curso">
            </div>
            <button type="submit" class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-pink-600
            hover:bg-pink-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-pink-500">Enviar</button>
        </form>
    </div>
</body>
</html>
```

Fonte: Autoria Própria, 2024.

Em seguida, será apresentado o formulário no aspecto visual, mostrando o resultado após a aplicação das estilizações descritas.

Figura 14 – Tailwind formulário

Fonte: Autoria Própria, 2024.

2.6 React

O React é uma biblioteca JavaScript utilizada para a criação de interfaces de forma particionada, que podem ser combinadas em componentes. Abrangendo desde sites a aplicativos móveis (REACT).

Concebida com o intuito de simplificar e agilizar a criação de interfaces, criada por Jordan Walke, um engenheiro do Facebook, em 2011. Se tornando atualmente a biblioteca mais popular de JavaScript (SILVA, 2021).

2.6.1 Vite

Vite vem do francês que significa “Rápido”, demonstrado em sua proposta de ser uma ferramenta que permite a criação de projetos *front-end* de forma acessível, sendo leve e prática, trazendo conceitos criativos para as páginas *web* (REACT VITE).

Para Schmitz e Georgii (2015), o React utiliza a implementação de camadas para a visualização de uma página *web,* utilizando como base componentes para a sua criação.

Utilizando-se Document Object Model (DOM), garantindo um desempenho superior a outras bibliotecas do gênero. E a criação de uma página nesta linguagem pode ser vista na imagem a seguir (REACT VITE).

Figura 15 – Exemplo Vite

```
1   import { useState } from 'react'
2   import reactLogo from './assets/react.svg'
3   import viteLogo from '/vite.svg'
4   import './App.css'
5
6   function App() {
7     const [count, setCount] = useState(0)
8
9     return (
10      <>
11        <div>
12          <a href="https://vitejs.dev" target="_blank">
13            <img src={viteLogo} className="logo" alt="Vite logo" />
14          </a>
15          <a href="https://react.dev" target="_blank">
16            <img src={reactLogo} className="logo react" alt="React logo" />
17          </a>
18        </div>
19        <h1>Vite + React</h1>
20        <div className="card">
21          <button onClick={() => setCount((count) => count + 1)}>
22            count is {count}
23          </button>
24          <p>
25            Edit <code>src/App.jsx</code> and save to test HMR
26          </p>
27        </div>
28        <p class="read-the-docs">
29          Click on the Vite and React logos to learn more
30        </p>
31      </>
```

Fonte: Autoria Própria, 2024.

E o resultado da criação de uma tela básica feita em React pode ser observado na figura 16 a seguir.

Figura 16 – React Vite exemplo

Fonte: Autoria Própria, 2024.

Utilizando desses conceitos, por utilizar como base o *HTML* para criação de páginas, para fazer um formulário não se torna muito distinto, como visto na imagem a seguir.

Figura 17 – Exemplo React

```
import './App.css'

function App(){

  return(
    <>
    <div className="login-box">
      <h2>
        Login
      </h2>

      <form action="">
        <div className="user-box">
          <input type="text" name="" id="" />
          <label htmlFor="">Username</label>
        </div>
        <div className="user-box">
          <input type="password" name="" id="" />
          <label htmlFor="">Password</label>
        </div>
        <a href="">
          Submit
        </a>
      </form>
    </div>
    </>
  )
}
export default App
```

Fonte: Autoria Própria, 2024.

A principal diferença da criação de uma página *HTML* para uma React são os atributos como o "*Class*" que fica "ClassName" no React.

Com base na imagem anterior, foi realizada a criação de um formulário simples que pode ser visto na figura 18 abaixo.

Figura 18 – React Formulario

Fonte: Autoria Própria, 2024.

2.6.2 AXIOS

O Axios é um framework de NodeJs e de linguagens de front-end como o React, sua principal função é de gerenciar as requisições com o protocolo Hypertext Transfer Protocol (HTTP) dentro de um ambiente *web* (AXIOS, 2024).

2.7 React Native

De acordo com Escudelario e Pinho (2020), o React Native é uma plataforma baseada em React, viabilizando a criação de aplicativos híbridos, executados no iOS (Apple), quanto no Android, sendo criada pelo Facebook em 2013.

O React Native pode ser definido como um framework de código aberto que visa a criação de aplicações nativas, ou seja, não há uma camada *web* como interface, mas sim o próprio aplicativo nativo. (CASA GRANDE; TANAKA, 2023).

Baseado na criação de sua estrutura em componentes, o React Native utiliza dos mesmos conceitos de criação de tela do React, podendo ser visto na figura 19 abaixo.

Figura 19 – React Native Exemplo

```
C: > Users > endri > Desktop > App-ReactNative > animes-app > src > App.js > styles > container
1   import { StyleSheet, Text, View, ImageBackground, Button, Alert,
2   } from 'react-native';
3
4   export default function HomeScreen({ navigation }) {
5     return (
6       <View style={styles.container}>
7         <Text style={styles.logo}>Animes</Text>
8         <Text style={styles.sub}>Melhor app de animes do Brasil</Text>
9         <ImageBackground
10        source={require('../assets/X.jpg')}
11        style={{width: 410, height: 600, top: 130, zIndex: -1}}
12        />
13
14          <Button
15          title="Go to Details"
16          onPress={() => navigation.navigate('Details')}
17          style={{zIndex: 1, color: '#fff'}}
18        />
19
20
21      </View>
22    );
23  }
```

Fonte: Autoria Própria, 2024.

Os principais componentes utilizados na criação dessa tela são:

- *Import*: utilizado para a importação de bibliotecas do React para facilitar a criação de telas com componentes prontos.
- *Export default function*: É a função principal que envia os componentes para a tela principal para ser interpretada.
- *View style*= {}: É um componente utilizado para representar a tela da aplicação, e o *style* é a estilização desse componente.
- *Text*: É basicamente um componente para a criação de textos na tela.
- *ImageBackground*: É uma imagem utilizada como plano de fundo do aplicativo.
- *Source*: Local onde a imagem que vai ser utilizada como plano de fundo.
- *Button*: Serve para a criação de um botão na aplicação.
- *Title:* Representa o nome que esse botão vai ter.
- *OnPress:* Representa qual vai ser a função que o botão vai executar quando for clicado.

Figura 20 – React Native 2

```
C: > Users > endri > Desktop > App-ReactNative > animes-app > src > App.js > ...
25
26  const styles = StyleSheet.create({
27    container: {
28      flex: 1,
29      justifyContent: 'center',
30      backgroundColor: '#011039',
31    },
32    title: {
33      textAlign: 'center',
34      marginVertical: 8,
35    },
36    logo: {
37      textAlign: 'center',
38      zIndex:'1',
39      fontSize: '70',
40      color: '#fff',
41      fontWeight: '800'
42    },
43    sub: {
44      textAlign: 'center',
45      zIndex:'1',
46      fontSize: '25',
47      color: '#fff',
48      fontWeight: '400',
49    },
50    fixToText: {
51      flexDirection: 'row',
52      justifyContent: 'space-between',
53    },
54
55
56  });
```

Fonte: Autoria Própria, 2024.

Na figura 20 utiliza os mesmos conceitos de estilização abordados no *CSS*, que a principal diferença é a sua estruturação que utiliza uma constante para definir o estilo de aplicativo.

2.8 EXPO

O Expo assim como o pacote do react "*create-react-app*" fornece a criação de toda a estrutura necessária para desenvolver uma aplicação, fornecendo um ambiente para a criação facilitada de aplicativos (ESCUDELARIO; PINHO, 2020).

Sendo assim o Expo é uma ferramenta utilizada na criação mobile com React Native que permite facilitar o acesso a algumas API's nativas do dispositivo sem precisar da instalação de qualquer dependência ou alterar código nativo (ROCKETSEAT,2020).

Na Figura 21 será demonstrado um formulário dentro do react-native com expo nas suas dependências.

Figura 21 – Exemplo Expo

```
MeuApp > App.js > App
You, 2 seconds ago | 1 author (You)
1  import React, { useState } from 'react';
2  import { StyleSheet, Text, TextInput, View, Button, Alert } from 'react-native';
3
4  export default function App() {
5    const [name, setName] = useState('');
6    const [email, setEmail] = useState('');
7    const [message, setMessage] = useState('');
8
9    const handleSubmit = () => {
10     Alert.alert('Formulário enviado', `Nome: ${name}\nEmail: ${email}\nMensagem: ${message}`);
11   };
12
13   return (
14     <View style={styles.container}>
15       <Text style={styles.title}>Formulário de Contato</Text>
16       <TextInput
17         style={styles.input}
18         placeholder="Nome"
19         value={name}
20         onChangeText={text => setName(text)}
21       />
22       <TextInput
23         style={styles.input}
24         placeholder="Email"
25         value={email}
26         onChangeText={text => setEmail(text)}
27         keyboardType="email-address"
28       />      You, 1 second ago • Uncommitted changes
29       <Button title="Enviar" onPress={handleSubmit} />
30     </View>
31   );
32 }
```

Fonte: Autoria Própria, 2024.

2.9 Node.JS

Node.js é um ambiente JavaScript que permite a execução dos comandos de forma externa, exercendo um papel de *back-end* aos moldes *non-blocking* (não bloqueante) realizando múltiplas execuções ao mesmo tempo, evitando atraso (PEREIRA, 2013).

Possuindo também como qualidades para sua atuação, junto de ser compatível com diversas plataformas, e sendo impulsionado pelas bibliotecas feitas pelos usuários, tornando-se uma ferramenta prática e acessível. (DUARTE JÚNIOR, 2022).

Figura 22 – Exemplo Node

```
1   const express = require('express');
2   const app = express();
3
4   app.get('/', function(req,res) {
5       res.send("Tela Principal")
6   });
7
8   app.get('/Tela2', function(req,res) {
9       res.sendFile("Tela 2")
10  });
11
12  app.get('/TelaParametros/:nome', function(req,res) {
13      res.send("ola "+ req.params.nome);
14  });
15
16
17  app.listen(1600, function(){
18      console.log("Servidor funcionando")
19  })
```

Fonte: Autoria Própria, 2024.

No exemplo acima pode ser visto as funcionalidades básicas do node de criar um servidor utilizando o Express em que os principais comandos são:

- *App*: Tem como intuito carregar todas as funcionalidades do Express dentro de si.
- Métodos: Dentro do node existem métodos para diversas funções em uma página web, no exemplo foi utilizado o *Get*.
- Req, Res: O *Request* e o Respons são responsáveis pelo envio de informações entre páginas.
- /Tela: É o local onde as ações dentro da função serão executadas.
- /TelaParametros/:nome: Tem a mesma funcionalidade do tópico anterior com diferença que para ser executado precisa de uma informação que no exemplo acima foi o “Nome”.

2.9.1 Express

O Express é um framework de NodeJs que tem como principal função facilitar o desenvolvimento de aplicações de forma prática, utilizando de middleware e roteamento, renderizado em telas HTML dinâmicas (BARSOTI; GIBERTONI, 2020).

2.9.2 Cors

O Cross-Origin Resource Sharing é responsável pelo compartilhamento de informações em uma página web utilizando do HTTP para a sua distribuição, sendo utilizado como framework de NodeJs (MOZILLA, 2024).

2.9.3 Node Package Manager

NPM é o gerenciador de pacotes do Node, que permite a instalação de diversos módulos que abrangem diversos aspectos como *back-end* e pagamento online, sendo eles realizados e disponibilizados pela comunidade (POWERS, 2019)

2.9.4 Node Package Executor

O Npx é um comando incorporado ao npm, realizando a execução de módulos sem instalá-los, servindo para requisições rápidas, através da remoção automática desses módulos após seu uso, economizando memória. (MORAES, 2021).

2.10 Python

Segundo Matthers (2016) Python é uma linguagem de programação extremamente eficientes pelo fato de os programas conterem menos linhas de códigos, ajudando a construção de um código “limpo”, obtendo uma compreensão e depuração rápida.

Para Menezes (2014) Python vem crescendo em várias áreas da computação, sendo elas bancos de dados, biotecnologia e jogos. Tornado assim uma linguagem de fácil aprendizagem e obtenção de resultados em pouco tempo através de suas bibliotecas.

Para Menezes (2019) a versatilidade, a simplicidade fazem do Python uma das linguagens de programação mais populares do mundo, o que a torna uma escolha valiosa para qualquer programador

Conforme Matthers (2016) a linguagem Python possui um interpretador que executa em uma janela de terminal, o que permite a possibilidade de teste de códigos sem a necessidade de salvar e executar um programa completo.

Na figura 23 será demonstrado um modelo de código exemplificando as funcionalidades da linguagem Python onde poderá ser inserido uma senha e o programa verificará se a senha é segura ou não.

Figura 23 – Exemplo teste de senha Python

Fonte: Autoria Própria, 2024.

- Função: refere-se a um bloco de código que realiza uma tarefa específica, sendo definida com o uso da palavra *def*, tendo então como objetivo organizar e reutilizar um código, promovendo uma eficiência.
- def: possui como função iniciar a definição de uma função chamada ‘verificar_senha’, que recebe como parâmetro(referência) ‘senha’.
- Len(senha): é uma função que retornará o número de caracteres que existem na senha fornecida. A expressão ‘>= 8’ compara esse número para ver se a senha tem pelo menos 8 caracteres.
- *Any*(char,isdigit() for char in senha): verifica se há pelo menos um digito na senha. O método ‘char,isdigit()’ avaliará se cada caractere na senha é um número. ‘*any*’ é uma função que retorna *‘true’* se qualquer caractere for um digito, caso contrário retorna ‘false’.
- *Any*(char.isupper() for char in senha): Possui uma similaridade anterior, esta linha usa ’*any*’ para determinar se há alguma letra maiúscula na senha. ‘char.isupper() ’é um método que verifica se o caractere ‘char’ é uma letra maiúscula.
- Estrutura Condicional *(If):* é usado para executar uma rotina apenas se uma condição específica for verdadeira.
- *Return* “Senha segura!”: Se a condição do ‘*if*’ for verdadeira, a função retorna a *String* “Senha segura!”.
- *Input* (“Digite sua senha para verificação:): Esta função incorporada exibe uma mensagem e espera que o usuário insira uma *string* através do teclado. Sendo assim a *string* inserida é então atribuída a variável ‘senha’.
- *Print* (resultado): é usado para exibir o valor de ‘resultado’, que informará ao usuário sobre a segurança de sua senha.

Para McKinney, (2018) O python muitas vezes poderá ser instalado pacotes adicionais, não incluídos na distribuição do Anaconda, sendo um deles o *pip* que é um programa que gerencia e cuida da instalação de pacotes Python.

Nesse sentido existem dois tipos: o conda e *pip* ambos com propósitos diferentes, o conda oferece um gerenciamento de pacotes geral para uma grande variedade de linguagens no ambiente conda e *pip* oferece serviços especificamente para Python. (MUELLER, 2020).

Na figura 24 demonstrará um exemplo de código que usará um pacote que não estão dentro do python então será realizado a instalação.

Figura 24 – Exemplo Python com importe

```
ProjetosDTeste > Projeto4.py > DecidaPorMim > iniciar
1   import random
2   from emoji import emojize
3
4   class DecidaPorMim:
5       def __init__(self):
6           self.respostas = [
7               'faça isso com certeza :thumbs_up:',
8               'não faça isso com certeza :thumbs_down:',
9               'não sei, você quem sabe :man_shrugging:',
10              'não recomendo, mas você pode tentar :thinking_face:',
11              'é uma boa pergunta :question:'
12          ]
13
14      def iniciar(self):
15          input('Faça uma pergunta para o nosso sistema! ')
16          print(emojize(random.choice(self.respostas)))
17
18  decida = DecidaPorMim()
19  decida.iniciar()
20
21
```

Fonte: Autoria Própria, 2024.

- *Import random*" Esta linha importa o módulo '*random*' do Python, usados para gerar números aleatórios. Esse módulo é utilizado no código para selecionar aleatoriamente uma resposta da lista de possíveis respostas."
- *From emoji import emojize*" Esta linha importa a função 'emojize' do pacote '*emoji*', que converte *strings* contendo códigos de texto de *emoji* em *emojis* visuais."
- Class DecidaPorMim" que permite a criação de objetos em Python com atributos (dados) e métodos (funções) associados."
- def__init__(Self): Este método especial é o construtor da classe, chamado quando uma nova instância da classe é criada, inicializando os atributos do objeto."
- self.resposta=[...]: "Dentro do construtor o atributo 'resposta' é definido é definido como uma lista de *strings*, cada uma podendo incluir códigos de *emoli*. "
- Decida = DecidaPorMim(): "cria uma instância da classe DecidaPormim. O construtor *init* é chamado para iniciar o novo objeto decida."
- Decida.iniciar(): "é chamado, iniciando o processo em que o usuário é solicitado a fazer uma pergunta e, em seguida, uma resposta aleatória é exibida."

Figura 25 – Exemplo de instalação de pacotes

```
PS C:\Users\danil\OneDrive\Área de Trabalho\Cursos\python> pip install emoji
Collecting emoji
  Downloading emoji-2.11.1-py2.py3-none-any.whl.metadata (5.3 kB)
Downloading emoji-2.11.1-py2.py3-none-any.whl (433 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 433.8/433.8 kB 6.8 MB/s eta 0:00:00
Installing collected packages: emoji
Successfully installed emoji-2.11.1
PS C:\Users\danil\OneDrive\Área de Trabalho\Cursos\python>
```

Fonte: Autoria Própria, 2024.

Na figura 25 é usado o *pip install emoji* para a instalação dos pacotes de *emojis*, devido não serem incluídos dentro do Python.

Figura 26 – Exemplo de emoji com Python

```
Faça uma pergunta para o nosso sistema! teste
não recomendo, mas você pode tentar 🤔
PS C:\Users\danil\OneDrive\Área de Trabalho\Cursos\python>
```

Fonte: Autoria Própria, 2024.

A figura 26 acima, demonstra a pergunta realizada pelo sistema com a resposta do usuário, gerando assim a função do sistema, pegando uma frase dentro da lista e respondendo junto com o *emoji*.

2.11 Flask

Um pequeno framework projetado para ser extensível, promovendo um núcleo sólido com os serviços básicos. O flask tem três dependências principais, os subsistemas de roteamento, depuração e WSGI (Web Server Gateway Interface). (GRINBERG, 2018).

O Flask é uma ferramenta que foi desenvolvida em 01 de abril de 2010, sendo muito utilizada para criar aplicações *web*, possuindo diversas vantagens, devido á flexibilidade que a ferramenta proporciona para a criação de *APIs*. (SILVA, 2019.)

2.12 Jupyter

Um ambiente de desenvolvimento de manipulação dados, o Jupyter Notebook é utilizado principalmente por conta da sua simplicidade e facilidade de uso e pelo suporte a linguagens de marcação e códigos (VASCONCELOS; GUERRA, 2023).

Figura 27 – Exemplo Jupyter

Fonte: Autoria Própria, 2024.

2.13 Machine Learning

O Machine Learning utiliza-se da filtragem de dados para a criação de novas informações, gerando resultados significativos, possibilitando a tomada de decisões inteligente através dos dados gerados (KNEUSEL, 2024).

A tecnologia está em constante evolução, e o aprendizado de máquina se torna crucial para o avanço de diversas áreas comerciais, sendo adotada pelas maiores empresas da atualidade como a Netflix (DOMINGOS, 2017).

2.14 Numpy

O NumPy é uma biblioteca da linguagem Python que visa primordialmente a possibilidade da utilização de computação numérica em projetos, criado em 2005 como uma biblioteca de código aberto (NUMPY).

Segundo Pereira (2020), o NumPy possui funções fundamentais para o gerenciamento de dados pelo fato de sua manipulação de matrizes e vetores com funções matemáticas de alto nível.

2.15 Pandas

Sendo utilizado como ferramenta para análise de dados, a biblioteca Pandas tem como função primordial o acesso à manipulação de dados utilizando de planilhas ou dados da internet (CHEN, 2018).

Para McKinney (2018), o Pandas obtém uma série de funções para a manipulação de dados sendo essencial para criações de análise de dados e machine learning.

E as principais funcionalidades do Pandas podem ser vistas na imagem a seguir:

Figura 28 – Exemplo Pandas

```
1   import pandas as pd
2
3   data_dict = {'Name': ['Alice', 'Bob', 'Charlie'],
4                'Age': [25, 30, 35],
5                'City': ['New York', 'Paris', 'London']}
6
7
8   df = pd.DataFrame(data_dict)
9
10  print(df)
11
12
```

Fonte: Autoria Própria, 2024.

Para a criação do exemplo acima foi utilizado o comando “DataFrame”, mas existem diversos comandos para analisar base de dados diferentes como:

- Read_csv: Lê dados de determinado arquivo ou URL para manipular dados.
- Read_table: Visualiza dados de determinada tabela de um arquivo ou URL.
- Read_fwf: Lê arquivos de uma coluna especifica com tamanho fixo.
- Read_clipboard: Utilizado para conversão de tabelas para páginas web, tendo muita similaridade com o “read_table”.
- Read_excel: Analisa dados de um arquivo Excel com extensões de XLS ou XLSX
- Read_html: Lê dados de uma tabela em um documento HTML.
- Read_json: Capta informações em um JavaScript Object Notation (JSON).

- Read_sql: Lê resultados de uma inquisição SQL.

2.16 Scikit-Learn

Utilizado como biblioteca de linguagem de máquina, o Scikit-Learn é um software de código livre que tem como intuito a análise de dados com base em algoritmos de classificação, agrupamento e entre outros (INHESTA, 2022).

E com a utilização de uma base de dados simples, pode ser feito a análise como apresentado na figura abaixo.

Figura 29 – Exemplo Sckit Learn

```
cossine_df = pd.DataFrame(rec_df['X-Men'].sort_values(ascending=False))
cossine_df.columns = ['Recomendações']
cossine_df.head(20)
```

[26]

title	Recomendações
X-Men	1.000000
X2: X-Men United	0.687475
Spider-Man	0.629953
Fifth Element, The	0.595723
Spider-Man 2	0.594701
Men in Black (a.k.a. MIB)	0.586287
X-Men: The Last Stand	0.585717
Star Wars: Episode VI - Return of the Jedi	0.583115
Matrix Reloaded, The	0.582737
Star Wars: Episode I - The Phantom Menace	0.580186
Shrek	0.580160
Gladiator	0.571908
Star Wars: Episode V - The Empire Strikes Back	0.571208
Matrix, The	0.568441
Indiana Jones and the Temple of Doom	0.555896

Fonte: Autoria Própria, 2024.

As três linhas de código demonstram ao Scikit-Learn analisando uma base de dados de filmes em que:

- Pd.Dataframe: é o comando onde prepara a base de dados para fazer a análise de recomendação
- Rec_df['x-man']: analisa o filme "x-men" na base de dados com base na avaliação da base de dados

- sort_values(ascending=False): Ordena os valores de uma coluna e com o atributo “ascending”, colocando os de ordem decrescente.

2.17 Banco De Dados

Para Cayres (2015) Banco de dados são coleções de dados que podem ser inseridos, atualizados e recuperados, sendo visto como uma representação de fatos, conceitos de maneira normalizada. Essa afirmação ressalta a importância do banco de dados, esclarecendo a possibilidade de armazenamento de informações tanto em sistemas simples quanto em sistemas complexos (ELMASRI; NAVATHE, 2011).

Segundo Caiut (2015) o Sistema de Gerenciamento de Banco de Dados (SGBD), que consiste na finalidade de gerenciar os Bancos de dados, facilitando assim o processo de definição, construção e manejo de bancos de Dados armazenados.

Seu surgimento iniciou na década de 1960, quando Charles Bachman desenvolveu a base do modelo de dados de rede; mais tarde, no final dessa década, a IBM lançou seu SGBD. (RAMAKRISHNAN; GEHRKE, 2008, p.24).

2.17.1 Diagrama Entidade Relacionamento

O Der são utilizados para visualizar e estruturar as relações entre os dados de um sistema de Banco de Dados, orientando os desenvolvedores e projetistas a entenderem o domínio do problema e a projetar uma base estrutural eficiente. (RAMAKRISHNAN; GEHRKE, 2008, p.24).

Figura 30 – Exemplo Der

Fonte: Autoria Própria, 2024.

Na figura 30 teremos a representação da sala cujo intuito é demonstrar os espaços físicos disponíveis para o uso. Já os alunos representam as pessoas que utilizam as salas, no contexto, os alunos que ocupam a sala para estudo. O relacionamento será estabelecido por Locação que ligará as salas aos alunos.

2.17.2 Modelo Entidade Relacionamento

Modelo de entidade relacional (MER) representa o banco como uma coleção de relações, onde cada equivale uma tabela de valores de registros lineares, chamados de tuplas, composto por atributos. (SILBERSCHATZ; SUDARSHAN, 2006).

Figura 31 – Exemplo Mer

Fonte: Autoria Própria, 2024.

Com a demonstração da figura acima é notório as seguintes pautas. No modelo de dados, temos duas entidades principais: Sala e Aluno. A entidade Sala é identificada pelo atributo. Num_Sala (PK), que serve como chave primária, indicando unicamente cada sala e sua capacidade máxima no atributo Quantidade_Alunos. Na entidade Aluno, Id_Aluno (PK) serve como identificador único, enquanto Nome e Ano_Aluno detalham as informações pessoais e acadêmicas. O atributo num_Sala (FK) em Aluno estabelece uma conexão direta com a Sala, mostrando onde cada aluno está alocado e garantindo a eficiência na gestão de espaço.

2.17.3 Banco De Dados Não Relacional

Para o termo NoSQL refere-se a um banco de dados que não segue normas de tabelas presente no banco de dados relacional, sendo ideais para lidar com grandes volumes de dados, solucionando problemas de escalabilidade (CAYRES,2015).

2.18 Firebase

Sendo eficaz e versátil para a criação de banco de dados, o Firebase faz uso de JSON para permitir atualizações em tempo real, junto a funções de armazenamento em nuvem, lhe tornando completo para diversos projetos (FIREBASE)

Figura 32 – Exemplo de banco de dados com o Firebase

Fonte: Autoria Própria, 2024.

2.19 UML

Sendo uma representação visual utilizada para auxiliar a compreensão do sistema em sua parte física e lógica, O Unified Modeling Language, tornando-se um padrão internacional aceito para a documentação de um software (GUEDES, 2018).

Utilizado para a realização de um projeto completo, e preciso que possa ser alterado posteriormente e permitindo um melhor entendimento entre os clientes e os desenvolvedores (PEREIRA, 2011).

Em termos gerais, através dos diagramas que constituem a UML, o projeto todo é abordado de diferentes formas técnicas para obter um melhor resultado em sua conclusão (GUEDES, 2018).

2.19.1 Caso De Uso

O caso de uso é o primeiro e mais simples diagrama de um sistema, com intuito de demonstrar todas as suas funcionalidades de forma compreensível entre profissionais e leigos (GUEDES, 2018).

Sendo considerado o diagrama mais importante que um sistema possuí, por ser a base dos demais diagramas e servindo como base para o software com a aplicabilidade da análise de sistemas (PRESSMAN; MAXIM, 2021).

Na Figura 33, pode ser vista um exemplo simples de um diagrama de caso de uso.

Figura 33 – Exemplo de Diagrama de Caso de Uso

Fonte: Guedes, 2018.

O exemplo abordado acima retrata o diagrama de caso de uso de um sistema veterinário que tem como principais elementos:

- Atores: São representados por um boneco palito que são pessoas ou sistemas externos que interagem diretamente com o sistema, no exemplo foi utilizado os atores Secretária, Veterinário e Cliente.
- Levantamento de Requisitos: São as principais funcionalidades que o sistema precisa ter, sendo dividido entre Requisito Funcional, Requisito não Funcional e as Regras de Negócio.
- Caso de Uso: São os reflexos dos requisitos funcionais que são representados por uma elipse com verbo no infinitivo que no exemplo acima foi utilizado “Gerencias Veterinários” e “Marcar Consulta”.
- Associação: Serve para demonstrar a interação entre Ator e Caso de Uso, é representado por uma reta.
- Generalização: Conhecido como Herança, demonstra o processo das heranças que pode ser utilizado entre Ator para Ator ou Caso de uso para Caso de uso, representado por uma reta com uma seta.
- Inclusão: Representado por uma reta pontilhada com “<<Include>>” em cima, representa quando uma ação do sistema é considerada obrigatório.

- Extensão: Representado por uma reta pontilhada com “<<Extend>>” em cima, pode ser utilizado para definir validações, Consequências ou Decisões podendo ser opcional.

2.19.2 Documentação de Caso de Uso

A documentação de um caso de uso tem como função descrever, por meio de informações em linhas gerais do caso de uso, como os atores interagem com ele e suas etapas para serem executadas pelo ator e sistema (GUEDES,2011).

Desta maneira não se existe um formato específico de documentação para os casos de uso definido pela UML, porém há formatos correspondentes a ela, deste modo o caso de uso é flexível para realização de sua documentação (GUEDES,2018).

Na figura 34 demonstra um exemplo de documentação de um caso de uso chamado Encerrar conta onde fornecerá características.

Figura 34 – Exemplo Documentação Caso de Uso

Nome do Caso de Uso	**UC05 – Realizar Depósito**
Ator Principal	Funcionário
Atores Secundários	Clientes
Resumo	Descreve os passos necessários para um cliente depositar algum valor em uma determinada conta
Pré-condições	
Pós-condições	
Cenário Principal	
Ações do Ator	**Ações do Sistema**
1. Informar o número da conta	
	2. Verificar se a conta existe
3. Fornecer o valor a ser depositado	
	4. Somar o valor depositado ao saldo da conta
	5. Executar caso de uso “Registrar Movimento”
Restrições/Validações	1. A conta precisa existir e estar ativa
Cenário de Exceção – Conta não encontrada	
Ações do Ator	**Ações do Sistema**
	1. Comunicar ao cliente que o número da conta informada não foi encontrado

Fonte: Guedes, 2018.

- Nome do Caso de Uso: Identifica claramente o objetivo do caso de uso no exemplo "Encerrar Conta".
- Ator Principal: Indica quem inicia o processo “Cliente”.
- Atores Secundários: Mostrar quem auxilia no processo “Funcionário”.
- Resumo: Tem como objetivo descrever brevemente o objetivo do caso de uso.

Exemplo: Este caso de uso descreve as etapas necessárias para que um cliente encerre uma conta.

- Pré-Condições: Especifica as condições que devem ser verdadeiras antes de iniciar o caso de uso “É necessário existir uma conta ativa”.
- Pós-Condições: Descreve o estado esperado após a conclusão do caso de uso no exemplo “A conta será encerrada”.
- Fluxo Principal: Sua função é detalhar os passos padrão que descrevem como o caso de uso é executado normalmente.

  Exemplo:

  Cliente solicita encerramento da conta fornecendo o número da conta.

  Sistema emite saldo.

  Sistema encerra a conta.
- Restrições/Validações: Defini regras que limitam como o caso de uso pode ser executado.

  “A conta só pode ser encerrada pelo seu titular e o saldo deve estar zerado”.
- Fluxos Alternativos: Demonstra variações do fluxo principal para diferentes cenários.

Exemplo:

- Saldo Positivo: Cliente deve realizar saque antes do encerramento.
- Saldo Negativo: Cliente deve fazer depósito para zerar o saldo.
- Manutenção do Cadastro do Cliente: Se for a única conta do cliente, o cadastro deve ser atualizado para inativo.

2.19.3 Diagrama de Atividade

Dentre todos da UML, o Diagrama de Atividade, é utilizado para a representação da lógica seguida para a realização de um procedimento, permitindo processos em separado demonstrado todo fluxo de trabalho (FOWLER; TORTELLO, 2005)

Na figura 35, possuí um exemplo de um diagrama de atividade e explicação de seus elementos.

Figura 35 – Exemplo Diagrama de Atividade

Fonte: FOWLER; TORTELLO, 2005.

Neste exemplo, é exemplificado a atividade “Realizar Pedido” que é constituída pela separação de processos após o nó de ação “Receber Pedido”.

Depois processos em paralelos são executados em ordem aleatória até a junção respectiva no final.

Na 1 parte, ocorre o “Preenchimento do Pedido” e uma decisão simples de ordem de prioridade sendo o tipo da entrega selecionada, que possuí uma intercalação e após isso se junta ao outro processo.

Enquanto na 2 parte, ocorre apenas o “Envio da Fatura” e o “Receber Pagamento” antes de se juntarem em um único fluxo.

Ao final do diagrama a atividade se termina, no “Nó de atividade Final”.

Os elementos utilizados comummente no diagrama de atividade são:

- Atividade: É a denominação geral dos processos a serem descritos, escrito de forma sucinta.
- Nó Inicial: Utilizado para demarcar o começo da atividade, e seu fluxo de processos.
- Nó de Atividade Final: Utilizado para demarcar o fim dos processos da atividade.
- Nó de Ação: Serve para a representação de uma etapa, consistindo em apenas um processo unitário.
- Fluxo de Controle: São os elementos visuais que mostram o caminho a ser seguido após determinado processo.
- Nós de Decisão: Representa a decisão entre dois ou mais processos, em que certa condição impacta no que deve ser executado.
- Intercalação: Utilizado para interconectar fluxos após um nó de decisão, que possuem a mesma continuação
- Separação ou Nó de Bifurcação: Utilizado para representar a separação de um Nó de Ação, em vários de forma paralela.
- Nó de Junção: Complemento do Nó de Bifurcação, por determinar o ponto de conexão entre os processos que antes estavam paralelos prosseguindo para um mesmo fluxo.

2.19.4 Diagrama de Sequência

Segundo Guedes (2011), um diagrama de sequência é um diagrama comportamental que ilustra processos, demonstrando a ordem em que eventos ocorrem com base em caso de uso específico.

Após a identificação dos eventos pelo caso de uso, o modelador cria um diagrama de sequência, que representará como esses eventos provocam o fluxo de um objeto para o outro ao longo dos processos. (PRESSMAN; MAXIM, 2021).

Tratando-se de um processo iniciado por um ator, geralmente dependendo do diagrama de classe, devido os objetos descritos no diagrama de sequência. Sendo uma abordagem excelente para completar o diagrama de classes. (GUEDES,2018).

A seguir na figura 36 será demonstrado um exemplo de caso de sequência mostrando um processo de emissão de saldo do sistema de controle bancário.

Figura 36 – Exemplo Diagrama de Sequência

Fonte: Guedes, 2018.

Nesse exemplo são utilizados alguns conceitos para a composição do caso de sequência sendo eles:

- **Atores:** representam entidades externas que interagem com o sistema e solicitam serviços, iniciando o processo, eles são representados por um boneco magro semelhante com os utilizados em diagrama de casos de uso.
- **Lifelines:** descrevem objetos ou entidades que participam da interação. Geralmente instancias de classes, com uma linha tracejada vertical indicando a duração de sua participação na interação, mostrando na visualização do fluxo dos eventos e ações.
- **Classes de Boundary e de Controle:** As lifelines podem representar instâncias de diferentes tipos de classes, como boundary (responsáveis pela interface com o usuário) e controle (responsáveis pela lógica de controle do sistema). Isso é indicado pelos estereótipos <<boundary>> e <<control>>.
- **Métodos chamados e retornos:** As mensagens trocadas entre as lifelines representam chamadas de métodos e retornos de valores. Por exemplo, o método consultarConta é chamado na lifeline da classe ControleEmitirSaldo e, se bem-sucedido, retorna um valor verdadeiro.
- **Fluxo de interação:** O diagrama de sequência detalha o fluxo de interação entre os participantes, mostrando a sequência de ações realizadas durante o processo de emissão de saldo em um sistema bancário.
- **Retorno de métodos:** Os retornos dos métodos podem ser valores específicos, como verdadeiro ou um saldo bancário representado por um valor numérico. Esses retornos são representados nas mensagens trocadas entre as lifelines.
- **Estereótipos:** Além dos estereótipos padrão para as classes (<<boundary>>, <<controll>>, <<entity>>), o diagrama também pode usar estereótipos específicos para indicar o tipo de lifeline, como <<entity>> para lifelines de classes de entidade.

2.19.5 Diagrama de Máquina de Estado

Criado para representar o comportamento de um sistema com o funcionamento das funções de cada elemento que o compõe, o Diagrama de Máquina de Estado, serve para representar funções individuais e coletivas de um sistema (GUEDES, 2018).

Na figura 37, é demonstrado a representação de um diagrama de máquina de estado de “Realizar Deposito” com seus devidos processos:

Figura 37 – Exemplo de Diagrama de Máquina de Estado

Fonte: Guedes, 2018.

Este diagrama é composto por diversos elementos gráficos em sua composição, e os principais dele são:

- Estado: Tem como função representar a situação de um elemento de um sistema, demonstrando os possíveis períodos e processos aplicados sobre ele.
- Transições: Representado por uma seta ligando dois estados, ele demonstra um evento e a mudança de estado de um determinado objeto.
- Estado Inicial: Determina onde que a função do sistema se inicia, representado por um círculo completamente preenchido.
- Estado Final: tem o mesmo objetivo do Estado inicial com a particularidade de determinar o fim do sistema, sendo representado por um círculo parcialmente preenchido.

- Atividades Internas: São atividades que determinam o tempo em que o Estado irá fazer a sua tarefa, sendo dividido por:
    - Entry: Executa a ação no primeiro momento em contato com o estado.
    - Exit: Executa a ação no final de contato com o estado.
    - Do: Executa a ação em todo o momento.
- Pseudoestado de Escolha: É utilizado como método de validação, condição ou escolha sobre possíveis ações que um estado pode fazer, representado por um losango.
- Barra de Bifurcação/União: Representa quando uma ação dentro de um sistema ocorre de forma paralela, é utilizado uma barra preta para a sua representação visual.

2.20 Wireframe

O wireframe é um esboço simplificado de estrutura de uma interface que ilustra as funções do produto. Isso permite estimar o esforço necessário de dados até a modelagem de teste de front-end conforme a proposta (TEIXEIRA, 2014). Para Lucidchart o termo “wireframe” descreve uma representação visual de desenvolvimento de páginas *web* e aplicativos *moveis*, sendo utilizados no início da criação de estrutura básica de uma tela antes de acrescentar o design final.

Desta maneira os wireframes são definidos em baixa fidelidade que se refere ao rascunho do produto, sendo simples e com poucos detalhes, utilizado para explicar a ideia inicial para seu time, clientes. (MIRO).

Enquanto os de Alta fidelidade são uma representação detalhada do produto, incluindo cores e elementos gráficos atraentes, demonstrando uma visão mais precisa da interface final, junto de suas funções. (TEIXEIRA, 2014).

Na figura 38 a seguir será demonstrado um exemplo de wireframes de baixa fidelidade que representa um protótipo de site e seus componentes, mostrando uma pré-visualização da página *web*.

Figura 38 – Exemplo Wireframe de Baixa Fidelidade

Fonte: Autoria Própria, 2024.

Na figura 39 demonstrará o wireframe de alta fidelidade do exemplo anterior com componentes mais detalhados, mostrado uma evolução do design anterior representando uma tela mais próxima do produto.

Figura 39 – Exemplo Wireframe Baixa 2

Fonte: Autoria Própria, 2024.

2.20.1 Design Centrado No Usuário (DCU)

O design Centrado no Usuário surgiu da Interação humano-computador (IHC) e consiste em uma metodologia de design de software para desenvolvedores ajudando a criação de aplicativos que atendam às necessidades de seus usuários. (LOWDERMILK, 2013, p.26)

2.20.2 Experiência Do Usuário (UX)

De acordo Teixeira (2014) tem como função ser uma disciplina do design focada na experiência emocional e sensorial dos usuários ao interagir com produtos, colocando o Usuário como centro no processo de criação, evitando decisões baseadas em suposições

De acordo com Miro, Focando a Experiência do usuário (UX), envolve a análise de como um usuário navega por um produto ou serviço, ilustrando os caminhos percorridos e as interações realizadas pelos usuários enquanto tentam completar uma tarefa.

3 DESENVOLVIMENTO

Neste capítulo, serão apresentados e demonstrados os processos de desenvolvimento do DaeLink desde seu planejamento até sua realização. Começando com a parte teórica, sendo elas diagramas da UML, Wireframes até seus códigos e demonstração de seus resultados por meio das páginas.

3.1 Diagrama de caso de uso

A seguir será demonstrado o caso de uso do sistema, expressando como a pessoa com deficiência e a empresa realizaram suas ações dentro do sistema DaeLink.

3.1.1 Diagrama de caso de uso

Em nosso sistema, os requisitos funcionais são as funcionalidades e serviços que um sistema deve oferecer que serão executados após a interação de um determinado usuário, assim utilizado como denotação para a criação de telas do sistema.

Tratando-se de requisitos Não Funcionais são aqueles que seu papel não está diretamente ligado a execuções de usuários, sendo específicos da infraestrutura do sistema responsável por partes como segurança, desempenho ou estética do sistema

As regras de negócios, são declarações que definem os aspectos de funcionamento do sistema ligados diretamente a diretrizes da empresa delimitando o que pode ou não ocorrer em determinada situação.

Requisito não funcional

Sistema

- O sistema deve ter compatibilidade com diversos tipos de dispositivo
- A aplicação deverá ser realizada para android e IOS
- O sistema deve ter uma interface fácil que passe confiabilidade do sistema
- O sistema deverá utilizar do sistema em nuvem do firebase para conectar o sistema
- O wesite deverá ser feita com a lógica de programação do javascript e python junto com a biblioteca React e os frameworks flask, skitc-learn, axios entre outros.

- A aplicação Mobile deverá ser desenvolvida com a biblioteca React Native junto com a ferramenta Expo da linguagem javascript
- A aplicação deve ser compatível com a versão 7.0 do Android e a 17.3.0 do IOS
- O sistema deverá ser modelado com base na UML
- O sistema deverá ter um sistema de recomendação com base em similaridade em vagas de trabalho

Requisito funcionais:

Requisitos funcionais Portador de Deficiência:

- RF01: O portador de deficiência poderá visualizar a vaga em processo.
- RF02: O portador de deficiência poderá entrar em processo para uma vaga específica.
- RF03: O portador de deficiência cadastra seus documentos.
- RF04: Portador de deficiência deleta seus documentos.
- RF05: Portador de deficiência atualiza seus documentos.
- RF06: Portador de deficiência visualiza seus documentos.
- RF07: Portador de deficiência poderá realizar o seu cadastro
- RF08: Portador de deficiência poderá realizar login.
- RF09: Portador de deficiência poderá visualizar as informações de sua apuração.
- RF10: Portador de deficiência poderá visualizar seu processo de apuração.

Requisitos funcionais administrador:

- RF01:O administrado deleta os documentos do Portador de Deficiência.
- RF02:O administrado visualizar os documentos do Portador de Deficiência.
- RF03:O administrado aprova os documentos do Portador de Deficiência.
- RF04:O administrado deleta os documentos da empresa.
- RF04:O administrado deleta empresa.
- RF05:O administrador atualiza empresa.
- RF05:O administrador consulta informações da empresa.

- RF06:O administrado aprova os documentos da empresa.
- RF07:O administrado realiza login.

Requisitos funcionais empresa

- RF01: a Empresa realiza seu cadastro.
- RF02:A empresa realiza login.
- RF03:A empresa cadastra seus documentos/vaga.
- RF04:A empresa visualiza seus documentos.
- RF05:A empresa visualiza o perfil da pessoa com deficiência.
- RF06:A empresa contatará Pessoa com deficiência para a vaga.
- RF07:A empresa visualiza documentos.
- RF08:Empresa poderá visualizar as informações de sua apuração situação pessoa com deficiência.

Regras de negócio

- As empresas que poderão procurar para preencher suas vagas remanescentes
- Supervisão de Ações pelo Administrador
- Restrição de Modificação de Documentos Após Análise
- Autenticação Necessária para Funcionalidades Avançadas
- O perfil de empresa está focado exclusivamente no ambiente web

Pode-se observar o caso de uso do sistema com a figura abaixo, que demonstra a interação entre o PCD, Administrador e a Empresa em relação ao sistema Daelink com todas suas funções respectivas, conforme as diretrizes da UML, segue a documentação dos casos de uso apresentado anteriormente separando-se entre: Requisitos Funcionais, Requisitos Não Funcionais e Regra de Negócio.

Figura 40 – Diagrama de Caso de Uso Geral do DaeLink

Fonte: Autoria Própria, 2024.

Logo após, pode ser visto a documentação de cada caso de uso presente em nosso sistema.

Tabela 1 – Descrição do caso de uso: Manter vagas

Nome do Caso de Uso	Manter Vagas
Ator Principal	Empresa
Ator Secundário	
Resumo	Esse caso de uso representa as ações que podem ser realizadas em relação as vagas de emprego disponibilizadas.
Pré-condições	Efetuar Login na tela de Login
Pós-condições	Pode-se fazer os processos do caso de uso "Realizar Cadastro".
Fluxo Principal	
Ações da Empresa	Ações do sistema
1.O caso de uso de uso começa quando a Empresa acessa a página de login.	
	2.O Sistema abre uma tela de gerenciamento das Vagas.
3.A Empresa pode realizar ações nesse registro.	
3.1 A Empresa seleciona a opção de Exclusão. 3.2 A Empresa seleciona a opção de consultar. 3.3 A Empresa seleciona a opção de atualizar as informações das Vagas. 3.4 A Empresa seleciona a opção de criar uma Vaga.	

Fonte: Autoria Própria, 2024.

Tabela 2 – Descrição do caso de uso: Gerenciar Empresa

Nome do Caso de Uso	Gerenciar Empresa
Ator Principal	Administrador
Ator Secundário	Empresa
Resumo	Este caso de Uso descreve as etapas em que o gerenciamento dos dados da empresa será realizado.
Pré-condições	Efetuar Login na tela de Login.
Pós-condições	Pode-se fazer os processos de cadastrar uma conta.
Fluxo Principal	
Ações do Administrador	Ações do sistema
1.O caso de uso de uso começa quando o Administrador acessa a página de login.	

	2.O Sistema abre uma tela de gerenciamento dos registros das Empresas.
3.O Administrador pode realizar ações nesse registro.	
3.1 O Administrador seleciona a opção de Exclusão. 3.2 O Administrador seleciona a opção de consultar. 3.3 O Administrador seleciona a opção de atualizar os dados da Empresa.	
Ações da Empresa	Ações do sistema
1. O caso de uso de uso começa quando a Empresa acessa a página de login.	
	2.O Sistema abre uma tela de gerenciamento dos registros das Empresas.
3. A Empresa pode realizar ações nesse registro.	
3.1 A empresa seleciona a opção de consultar. 3.2 A empresa seleciona a opção de atualizar os dados da Empresa.	

Fonte: Autoria Própria, 2024.

Tabela 3 – Descrição do caso de uso: Gerenciar Apuração

Nome do Caso de Uso	Gerenciar apuração
Ator Principal	Empresa
Ator Secundário	PDC
Resumo	Este caso de uso representa a consulta do status da apuração dos documentos.
Pré-condições	Já ter ocorrido a apuração das informações do PCD.
Pós-condições	
Fluxo Principal	
Ações da Empresa	Ações do sistema
1. A Empresa seleciona a opção para validar as condições do PCD.	
	2. O Sistema demonstra os documentos enviados pelo candidato (PCD).
3. A Empresa apura os documentos e fica a critério deles a validação.	
	4. A Empresa apura os documentos, e valida no Sistema.
Fluxo Alternativo	

Ações do PCD	Ações do sistema
1. O caso de uso começa quando a Empresa escolhe verificar a condição do PCD apurado.	
	2. O Sistema mostra os status da apuração do PCD.
3.O caso de uso começa quando o PCD escolhe verificar a apuração feita pela empresa.	
	2. O Sistema abre uma tela que mostra os status da sua apuração.

Fonte: Autoria Própria, 2024.

Tabela 4 – Descrição do caso de uso: Gerenciar documento

Nome do Caso de Uso	Gerenciar Documento
Ator Principal	PCD
Resumo	Este caso de uso representa as ações que podem ser realizadas em relação aos documentos enviados pelo candidato (PCD).
Pré-condições	Já estar em um processo de seleção. "Visualizar vagas ingressadas"
Pós-condições	Pode-se fazer os processos do caso de uso "Visualizar Documentos", "Gerenciar Apuração"
Fluxo Principal	
Ações do PCD	Ações do sistema
1.O caso de uso de uso começa quando o PCD seleciona a tela de gerenciamento de documentos.	
	2. O Sistema abre uma tela de gerenciamento de Documentos.
3.O PCD pode realizar ações nesses Documentos.	
3.1 O PCD seleciona a opção de Exclusão. 3.2 O PCD seleciona a opção de consultar. 3.3 O PCD seleciona a opção de atualizar as informações do documento.	
Fluxo Alternativo	
Ações doEmpresa	Ações do sistema
1.O caso de uso começa quando a empresa seleciona o gerenciamento de documento	
	2. O sistema demonstra os PCDs que se cadastraram na vaga e seus documentos
3. A empresa seleciona a opção de Visualização do documento.	

4. A empresa seleciona entrar em contato com a empresa	

Fonte: Autoria Própria, 2024.

Tabela 5 – Descrição de Caso de uso: Gerenciar Mensagens

Nome do Caso de Uso	Gerenciar Mensagens
Ator Principal	PCD
Ator Secundário	
Resumo	Este caso de uso representa o gerenciamento de mensagens do sistema.
Pré-condições	O pcd deverá estar autenticada no sistema (Fazer login)
Pós-condições	
Fluxo Principal	
Ações Empresa	Ações do sistema
1. O caso de uso começa quando a empresa entra no gerenciamento de mensagem.	
	2. O Sistema abre uma página de contato onde a empresa e o pcd gerenciam as mensagens
Ações PCD	Ações do Sistema
1. O caso de uso começa quando o PCD decide gerenciar suas mensagens	
	2. O Sistema abre uma tela que mostra os as mensagens digitadas por ele ou pela empresa

Fonte: Autoria Própria, 2024.

Tabela 6 – Descrição do caso de uso: Visualizar Apuração

Nome do Caso de Uso	Visualizar apuração
Ator Principal	PCD
Ator Secundário	Empresa
Resumo	Este caso de uso representa a visualização da apuração tanto por via empresa ou PCD.
Pré-condições	O pcd deverá ter gerenciado suas vagas ingressadas. A empresa deve ter gerenciado suas apurações
Pós-condições	
Fluxo Principal	
Ações Empresa	Ações do sistema
1. O caso de uso começa quando o PCD entra para visualizar suas apurações.	

	2. O Sistema abre uma página onde mostrará se foi ou não apurado.
Fluxo alternativo	
Ações Empresa	Ações do Sistema
1. O caso de uso começa quando a empresa entra para visualizar a sua apuração ao candidato selecionado	
	2. O Sistema abre uma tela que mostra os as apurações feitas ou não.
3. A empresa escolhe se deve ou não apurar o candidato.	

Fonte: Autoria Própria, 2024.

Tabela 7 – Descrição do caso de uso: Gerenciar vagas ingressadas

Nome do Caso de Uso	Gerenciar vagas ingressadas
Ator Principal	PCD
Resumo	Esse caso de uso descreve as etapas em que o Portador de Deficiência realizará ao Ingressar em uma vaga em processo
Pré-condições	Ator ter realizado Login
Pós-condições	O sistema registra a inscrição da Pessoa com Deficiência no processo seletivo.
Fluxo Principal	
Ações do cliente	Ações do sistema
1. O caso de uso começa quando a Pessoa com Deficiência seleciona a opção "Ingressar na vaga" para uma vaga específica.	
	2. O sistema exibir uma tela para entrar em um processo seletivo de uma empresa escolhida pelo ator.
3. Ator envia suas informações pessoais.	
	4. O sistema exibe uma mensagem de confirmação da inscrição.

Fonte: Autoria Própria, 2024.

Tabela 8 – Descrição do caso de uso: Visualizar vagas

Nome do Caso de Uso	Visualizar vagas
Ator Principal	PCD
Resumo	Esse caso de uso descreve as etapas que um Portador de Deficiência realiza ao visualizar Vagas em Processo de uma empresa.

Pré-condições	Cliente deverá fazer login e entrar na área de Visualização de Vaga para poder realizar esse caso de uso.
Pós-condições	O ator visualiza o processo da vaga da empresa desejada
Fluxo Principal	
Ações do cliente	Ações do sistema
1. O caso de uso começa quando o a Pessoa com deficiência acessar área de visualização de processo de vagas.	
	2. O sistema exibe uma tela de processo da vaga
3. Ator selecionar detalhes para visualizar.	
	4- O sistema exibi detalhes do processo.

Fonte: Autoria Própria, 2024.

Tabela 9 – Descrição do caso de uso: Manter PCD

Nome do Caso de Uso	Manter PCD
Ator Principal	PCD
Ator Secundário	Administrador
Resumo	Esse caso de uso representa as ações que podem ser feitas em relação ao registro de um portador de deficiência no banco de dados.
Pré-condições	Efetuar Login na tela de Login
Pós-condições	
Fluxo Principal	
Ações do Administrador e PCD	Ações do sistema
1.O caso de uso de uso começa quando o Administrador acessa a página de login.	
	2.O Sistema abre uma tela de gerenciamento dos registros dos PCDs.
3.O Administrador e PCD pode realizar ações nesse registro.	
3.1 Selecionar a opção de Exclusão. 3.2 Selecionar a opção de consultar. 3.3 Selecionar a opção de atualizar os dados do PCD.	
1.O caso de uso de uso começa quando o PCD acessa a página de login.	

	2.O Sistema abre uma tela de gerenciamento dos registros dos PCDs.
3.O PCD pode realizar ações nesse registro.	
3.1 O PCD seleciona a opção de Exclusão. 3.2 O PCD seleciona a opção de consultar. 3.3 O PCD seleciona a opção de atualizar os dados do PCD.	

Fonte: Autoria Própria, 2024.

Tabela 10 – Descrição do Caso de uso: Visualizar PCD

Nome do Caso de Uso	Visualizar PCD
Ator Principal	Empresa
Ator Secundário	
Resumo	Este caso de uso descreve como uma empresa pode consultar o perfil de portador e deficiência
Pré-condições	A empresa deverá estar autenticada no sistema (Fazer login)
Pós-condições	
Fluxo Principal	
Ações Atores	Ações do sistema
1. A empresa acessa sua conta na plataforma.	
2. O cliente navegará até o perfil desejado.	
	3. O sistema verifica se o cliente está autenticado.
	4. O sistema exibe o perfil da pesquisa da empresa, demonstrando detalhes sobre o perfil.

Fonte: Autoria Própria, 2024.

Tabela 11 – Descrição do caso de uso: Fazer login

Nome do Caso de Uso	Fazer Login
Ator Principal	PCD, Empresa, Administrador
Ator Secundário	
Resumo	Esse caso de uso descreve as etapas em que ocorrerá a "autenticação com o banco de Dados".
Pré-condições	Preexistir as informações no Banco de Dados.
Pós-condições	Após realizar a autenticação, será permitido a conclusão de outros casos de uso

Fluxo Principal	
Ações Atores	Ações do sistema
1- Ator irá inserir as informações de login (Email e Senha).	
	2- O sistema irá autenticar com o Banco de Dados se as informações existem e estão corretas.
3- Se as informações estiverem corretas, o sistema permite que o cliente prossiga para outros casos de uso.	
Fluxo Alternativo	
Ações do Cliente	Ações do sistema
1.O Cliente vai inserir as informações de login (Email e Senha).	
	2. O Sistema irá autentificar com o Banco de Dados se as informações existem e estão corretas.
	3. Se não existir ou estiver incorreto, irá apresentar um aviso como “Email ou Senha incorretos”.

Fonte: Autoria Própria, 2024.

Tabela 12 – Descrição do caso de uso: Relacionar PCD a vaga

Nome do Caso de Uso	Relacionar PCD a vaga
Ator Principal	Empresa
Ator Secundário	PCD
Resumo	Este caso de uso descreve as etapas que irão ser realizado para se contatar o pcd para a vaga
Pré-condições	A empresa deverá estar autenticada no sistema (Fazer login) A Portador com Deficiência deverá estar autenticada no sistema (Fazer login)
Pós-condições	
Fluxo Principal	
Ações Empresa	Ações do sistema
1. O caso de uso começa quando a empresa contatará com o PCD para preenchimento de sua vaga.	
	2. O Sistema abre uma tela que mostra os candidatos que a empresa se interesse.
3. A empresa seleciona uma opção de envio ao candidato.	
	4. O sistema enviará ao candidato para preencher vaga

Fonte: Autoria Própria, 2024.

3.1.2 Diagrama de atividade

A seguir será demonstrado os diagramas de atividade sendo usado para a modelagem do sistema DaeLink, tendo como foco descrever o fluxo de atividades ou ações do sistema, determinando os processos e ajudado na orientação e visualização do cenário de funcionamento do sistema, mostrando ações em sequência.

Figura 41 – Diagrama de Atividade: Manter Vagas

Fonte: Autoria Própria, 2024.

Figura 42 – Diagrama de Atividade: Gerenciar empresa

Fonte: Autoria Própria, 2024.

Figura 43 – Diagrama de Atividade: Gerenciar apuração

Fonte: Autoria Própria, 2024.

Figura 44 – Diagrama de Atividade: Gerenciar Documento

Fonte: Autoria Própria, 2024.

Figura 45 – Diagrama de Atividade: Gerenciar mensagens

Fonte: Autoria Própria, 2024.

Figura 46 – Diagrama de Atividade: Visualizar Apuração

Fonte: Autoria Própria, 2024.

Figura 47 – Diagrama de Atividade: Gerenciar vagas Ingressadas

Fonte: Autoria Própria, 2024.

Figura 48 – Diagrama de Atividade: Visualizar Vaga

Fonte: Autoria Própria, 2024.

Figura 49 – Diagrama de Atividade: Manter PCD

Fonte: Autoria Própria, 2024.

Figura 50 – Diagrama de Atividade: Visualizar PCD

Fonte: Autoria Própria, 2024.

Figura 51 – Diagrama de Atividade: Relacionar PCD a Vaga

Fonte: Autoria Própria, 2024.

Figura 52 – Diagrama de Atividade: Fazer Login

Fonte: Autoria Própria, 2024.

Figura 53 – Diagrama de Atividade: Fazer cadastro

Fonte: Autoria Própria, 2024.

3.1.3 Diagrama de máquina de estado

Tratando agora da diagramação de máquina de estado ou também conhecido como de diagrama de estado, possui como ideia trazer em uma prototipação de um projeto os estados pelos quais um objeto passa durante seu ciclo de vida, bem como as transições que ocorrem entre esses estados. Sendo utilizado no sistema DaeLink como forma de descrever como um objeto responde a eventos com base em seu estado atual e como esse estado afetará o próprio estado do objeto.

Figura 54 – Diagrama de Máquina de Estado: Manter Vagas

Fonte: Autoria Própria, 2024.

Figura 55 – Diagrama de Máquina de Estado: Gerenciar Empresa

Fonte: Autoria Própria, 2024.

Figura 56 – Diagrama de Máquina de Estado: Gerenciar Apuração

Fonte: Autoria Própria, 2024.

Figura 57 – Diagrama de Máquina de Estado: Manter Documentos

Fonte: Autoria Própria, 2024.

Figura 58 – Diagrama de Máquina de Estado: Gerenciar Mensagens

Fonte: Autoria Própria, 2024.

Figura 59 – Diagrama de Máquina de Estado: Visualizar Apuração

Fonte: Autoria Própria, 2024.

Figura 60 – Diagrama de Máquina de Estado: Gerenciar Vagas Ingressadas

Fonte: Autoria Própria, 2024.

Figura 61 – Diagrama de Máquina de Estado: Visualizar Vaga

Fonte: Autoria Própria, 2024.

Figura 62 – Diagrama de Máquina de Estado: Manter PCD

Fonte: Autoria Própria, 2024.

Figura 63 – Diagrama de Máquina de Estado: Fazer Login

Fonte: Autoria Própria, 2024.

Figura 64 – Diagrama de Máquina de Estado: Fazer cadastro

Fonte: Autoria Própria, 2024.

Figura 65 – Diagrama de Máquina de Estado: Relacionar PCD a vaga

Fonte: Autoria Própria, 2024.

Figura 66 – Diagrama de Máquina de Estado: Visualizar PCD

Fonte: Autoria Própria, 2024.

3.1.4 Diagrama de sequência

Continuando com a diagramação teremos agora o de sequência que possui como objetivo representar a linha cronológica de ações dentro do sistema, ou seja, demonstrará a interação entre dois objetos ou componentes de um sistema junto de seu tempo de execução de funcionalidades especificas, demonstrando assim o comportamento dinâmico do DaeLink.

Figura 67 – Diagrama de Sequência: Manter Vaga

Fonte: Autoria Própria, 2024.

Figura 68 – Diagrama de Sequência: Gerenciar Empresa

Fonte: Autoria Própria, 2024.

Figura 69 – Diagrama de Sequência: Criar Empresa

Fonte: Autoria Própria, 2024.

Figura 70 – Diagrama de Sequência: Editar empresa

Fonte: Autoria Própria, 2024.

Figura 71 – Diagrama de Sequência: Gerenciar apuração

Fonte: Autoria Própria, 2024.

Figura 72 – Diagrama de Sequência: Gerenciar documento

Fonte: Autoria Própria, 2024.

Figura 73 – Diagrama de Sequência: Gerenciar mensagens

Fonte: Autoria Própria, 2024.

Figura 74 – Diagrama de Sequência: Gerenciar Visualizar PCD

Fonte: Autoria Própria, 2024.

Figura 75 – Diagrama de Sequência: Relacionar Vaga

Fonte: Autoria Própria, 2024.

Figura 76 – Diagrama de Sequência: Visualizar vaga

Fonte: Autoria Própria, 2024.

Figura 77 – Diagrama de Sequência: Gerenciar vagas ingressadas

Fonte: Autoria Própria, 2024.

Figura 78 – Diagrama de Sequência: Manter documento

Fonte: Autoria Própria, 2024.

Figura 79 – Diagrama de Sequência: Manter PCD

Fonte: Autoria Própria, 2024.

Figura 80 – Diagrama de Sequência: Editar PCD

Fonte: Autoria Própria, 2024.

Figura 81 – Diagrama de Sequência: Excluir PCD

Fonte: Autoria Própria, 2024.

Figura 82 – Diagrama de Sequência: Visualizar apuração

Fonte: Autoria Própria, 2024.

Figura 83 – Diagrama de Sequência: Gerenciar mensagens

Fonte: Autoria Própria, 2024.

Figura 84 – Diagrama de Sequência: Manter PCD

Fonte: Autoria Própria, 2024.

Figura 85 – Diagrama de Sequência: Editar PCD

Fonte: Autoria Própria, 2024.

Figura 86 – Diagrama de Sequência: Deletar PCD

Fonte: Autoria Própria, 2024.

Figura 87 – Diagrama de Sequência: Visualizar PCD

Fonte: Autoria Própria, 2024.

Figura 88 – Diagrama de Sequência: Gerenciar empresa

Fonte: Autoria Própria, 2024.

Figura 89 – Diagrama de Sequência: Editar Empresa

Fonte: Autoria Própria, 2024.

Figura 90 – Diagrama de Sequência: Consultar Empresa

Fonte: Autoria Própria, 2024.

3.2 Prototipação do sistema

Esta etapa envolve a criação das interfaces para a prototipação do website fazendo uso de wireframes de baixa e alta fidelidade. Possuindo como propósito apresentar e demonstrar cada funcionalidade das páginas do sistema, que estão divididas entre os dois usuários do sistema PCD, Empresas.

O primeiro Wireframe abordará a página inicial de visualização do Pcd contendo uma apresentação do sistema e suas funcionalidades, em que o logotipo principal é exibido de forma destacada no cabeçalho da página, acompanhada por uma barra de navegação. O menu de navegação será composto por links para as seções de Processos, Vagas, Empresas e Perfil.

Cada página do site será projetada para oferecer uma experiência de usuário intuitiva e acessível, garantindo que todas as funcionalidades sejam de fácil acesso e uso. Além disso, os wireframes de alta fidelidade incluirão detalhes visuais mais refinados, como cores, tipografia e ícones, proporcionando uma visão mais realista de como o site final será apresentado aos usuários. Este processo de prototipação é crucial para identificar e resolver problemas de usabilidade antes do desenvolvimento final, assegurando que o produto atenda às necessidades de todos os usuários de forma eficaz.

3.2.1 Wireframes DaeLink Visão PCD

Consequentemente, apresentamos a Home do PCD em baixa fidelidade. Esta página incluirá informações sobre o DaeLink, e na barra de navegação estarão disponíveis *links* para páginas onde o usuário poderá acessar e realizar suas funções dentro do sistema. A home do Daelink na visão da empresa terá a mesma estrutura da página *web*, tendo como diferencia algumas informações e funções de paginação diferentes.

Figura 91 – Wireframe de baixa fidelidade: Home do PCD

Fonte: Autoria Própria, 2024.

Agora será apresentado o wireframe de alta fidelidade, que mostrará o resultado da página inicial com suas cores e ideias implementadas, vista pelo portador de necessidade especiais. Caso o usuário tenha entrado como empresa as informações seriam outra.

Figura 92 – Wireframe de alta fidelidade: Home do PCD

Fonte: Autoria Própria, 2024.

Consequentemente, o wireframe de baixa fidelidade abordará a página de *login*, onde o usuário irá autenticar sua conta em nosso sistema. Nesta página, o usuário poderá inserir seu e-mail e senha nos campos apropriados o permitindo acessar suas funções dentro do sistema.

Figura 93 – Wireframe de baixa fidelidade: Login

Fonte: Autoria Própria, 2024.

Em seguida, é apresentado o wireframe de alta fidelidade da tela de Login, onde o PCD ou a empresa poderá se autenticar no sistema. Para acessar, é necessário ter uma conta registrada no site dito anteriormente. Este design incorpora as cores e imagens estabelecidas para garantir uma experiência visual atraente.

Figura 94 – Wireframe de alta fidelidade: Login PCD

Fonte: Autoria Própria, 2024.

Agora, vamos abordar a ideia de que um usuário com necessidades especiais (PcD) ou empresa não possua uma conta ele poderá registrar-se ao clicar no botão "Registrar-se" na tela de login. Quando o usuário acessar, será redirecionado para uma página onde mostrará qual tipo de conta o usuário pretenderá criar ou seja ele poderá gerar uma conta de empresa se ele assim preferir ou conta de portador de necessidades especiais.

Figura 95 – Wireframe de baixa fidelidade: Seleção de cadastro

Fonte: Autoria Própria, 2024.

A seguir, apresentamos o wireframe de alta fidelidade da área de seleção do tipo de conta a ser criado. Onde ao escolher uma determinada opção o sistema redirecionará para uma determinada página onde o usuário irá realizar seu cadastro.

Figura 96 – Wirefreme de alta fidelidade: Seleção de criação de conta

Fonte: Autoria Própria, 2024.

Consequentemente após o usuário (PCD) escolher a opção de cria sua conta tipo usuário, será redirecionado para a página de cadastro onde o portador de necessidades especiais poderá submeter suas informações de cadastro e assim fazer parte da comunidade DaeLink. Mas caso escolha empresa será redirecionado para uma página semelhante do usuário, tendo somente como diferencial algumas informações de cadastro para a companhia .

Figura 97 – Wireframe de baixa fidelidade: Realização de cadastro

Fonte: Autoria Própria, 2024.

A seguir, apresentamos o wireframe de alta fidelidade para o cadastro de Pessoas com Deficiência (PCD). Caso o usuário ainda não possua uma conta, ele deverá fornecer algumas informações específicas para concluir o cadastro no sistema, e para que não fiquei muita informação em uma só tela foi realizado a separação dos registros em etapas facilitando ao usuário a submissão de suas informações.

Figura 98 – Wireframe de alta fidelidade: Realização de cadastro

Fonte: Autoria Própria, 2024.

Na sequência do wireframe de baixa fidelidade, teremos a visualização das vagas disponíveis para o PCD realizar sua candidatura. Mostrando a possibilidade de entrar na vaga da empresa desejada que mais se encaixa com sua profissão.

Figura 99 – Wireframe de baixa fidelidade: visualizar vagas disponíveis

Fonte: Autoria Própria, 2024.

No próximo wireframe de alta fidelidade, serão demonstradas as vagas para as quais o portador de deficiência poderá se candidatar, com a possibilidade de entrar e esperar por seu processo.

Figura 100 – Wireframe de alta visualizar vagas disponíveis

Fonte: Autoria Própria, 2024.

Logo após entrar na vaga desejada, teremos um wireframe de baixa fidelidade que possui como intuito demonstrar o cadastro que o portador de deficiência realizara para se candidatar-se na vaga que deseja.

Figura 101 – Wireframe de baixa entrar vaga

Fonte: Autoria Própria, 2024.

O wireframe abaixo se refere a de alta fidelidade, onde tem como objetivo demonstrar a página já construída para melhor representação, mostrando ao portador de necessidades especiais se a vaga está ou aberta para a realização de sua candidatura.

Figura 102 – Wireframe de alta fidelidade Entrar em Vagas

Fonte: Autoria Própria, 2024.

Na sequência do wireframe de baixa fidelidade, teremos a visualização das vagas cujas inscrições já foram realizadas, mostrando a possibilidade de envio do documento para a empresa, entrar em contato e visualizar a apuração ou situação.

Figura 103 – Wireframe de baixa fidelidade: Visualizar vagas já candidatada

Fonte: Autoria Própria, 2024.

No próximo wireframe de alta fidelidade, serão demonstradas as vagas para as quais foram realizadas inscrições, com informações essenciais e uma melhor visualização.

Figura 104 – Wireframe de alta fidelidade: visualizar processos realizados

Fonte: Autoria Própria, 2024.

Após o usuário acessar a área de envio de documentos, ele encontrará diversos campos que devem ser preenchidos com as informações solicitadas. Esses dados permitirão que as empresas analisem a candidatura do usuário e decidam sobre a confirmação de sua vaga para o PcD em sua companhia.

Figura 105 – Wireframe de baixa fidelidade: enviar documento do PCD para empresa

Fonte: Autoria Própria, 2024.

A seguir apresentaremos o wireframe de alta fidelidade, que abordará o envio de documentos para a empresa dito anteriormente com mais detalhes de visualização.

Figura 106 – Wireframe de alta fidelidade: enviar documento do PCD para empresa

Fonte: Autoria Própria, 2024.

Tratando agora da funcionalidade de edição de documento poderemos perceber o wireframe de baixa fidelidade onde o portador de necessidade especial, poderá realizar a edição de seu documento após o seu envio dito anteriormente.

Figura *107*– Wireframe de baixa fidelidade: Atualizar documento

Fonte: Autoria Própria, 2024.

Entrando no wireframe de alta fidelidade é notório com maior precisão a página de edição de documento do portador de necessidade especial, demonstrando assim o envio ou a função de deletar esse documento.

Figura *108* – Wireframe de alta fidelidade: Atualizar documento

Fonte: Autoria Própria, 2024.

O wireframe a ser tratado agora é um de baixa fidelidade onde apresenta a funcionalidade de contatar a empresa para ter informações relevantes ao PCD sendo representado esse contato através de um chat. O chat poderá ser realizado tanto pela parte do PCD ou empresa, tendo em vista que ambos terão a possibilidade de entrarem em contato.

Figura 109 – Wireframe de baixa fidelidade: Chat parte Portador de deficiência

Fonte: Autoria Própria, 2024.

Entrando no wireframe de alta do chat é notório algumas mensagens tanto da empresa quanto do PCD na qual foi selecionada pela empresa junto de suas próprias mensagens onde são demonstrado essa interação entre ambos.

Figura 110 – Wireframe de Alta fidelidade: Chat PCD

Fonte: Autoria Própria, 2024.

Tratando do wireframe de baixa fidelidade do perfil do PCDs, será possível visualizar e editar as informações registradas após ter feito o cadastro, desta maneira será capaz de perceber dois botões onde o primeiro se remete a opção de visualização de suas informações, a segunda opção se remeterá a configuração onde o usuário poderá sair de sua conta ou editar seu perfil.

Figura 111 – Wireframe de baixa fidelidade: Visualização de perfil do PCD

Fonte: Autoria Própria, 2024.

O wireframe de alta fidelidade a seguir apresenta a visualização detalhada do perfil do PCDs. Esta tela exibe todas as informações registradas durante o cadastro, permitindo ao usuário visualizar e editar seus dados conforme necessário.

Figura 112 – Wireframe de alta fidelidade: Visualização de perfil do PCD

Fonte: Autoria Própria, 2024.

Ao usuário clicar na opção de edição de perfil ele será redirecionado para a página onde apresentará algumas informações suas, o possibilitando a troca de seus dados junto de senha, imagens de perfis e outras informações relevantes.

Figura 113 – Wireframe baixa fidelidade: Editar Perfil PCD

Fonte: Autoria Própria, 2024.

Consequentemente teremos o wireframe em alta fidelidade onde será apresentado com maiores detalhes de como será a página de edição de perfil.

Figura 114 – Wireframe de alta fidelidade: Editar perfil do PCD

Fonte: Autoria Própria, 2024.

3.2.2 Wireframes DaeLink Visão Empresa

O wireframe abaixo tem como objetivo mostrar o processo de acesso da empresa para criar uma vaga que precisa ser preenchida em sua companhia. Nesta tela, haverá diversos campos de entrada de dados que devem ser preenchidos para criar uma vaga completa e atrativa.

Figura 115 – Wireframe de baixa fidelidade: criar vaga

Fonte: Autoria Própria, 2024.

A seguir, o wireframe de alta fidelidade apresentará a criação de vagas. Nesta seção, a empresa poderá estabelecer vagas para PCD (Pessoas com necessidades especiais) e, assim, preencher suas posições de maneira mais eficiente.

Figura 116 – Wireframe de alta fidelidade: criar vaga

Fonte: Autoria Própria, 2024.

Se a empresa desejar visualizar ou pesquisar usuários que melhor se encaixam em suas vagas através do sistema de recomendação, ela poderá fazer isso através da tela mostrada no wireframe de baixa fidelidade a seguir.

Figura 117 – Wireframe de baixa fidelidade: visualizar candidatos para a vaga

Fonte: Autoria Própria, 2024.

Como citado anteriormente no wireframe de baixa fidelidade, a figura abaixo apresentará a pesquisa de candidatos que melhor se encaixam na vaga, caso a empresa opte por utilizar essa funcionalidade.

Figura 118 – Wireframe de alta fidelidade: visualizar candidatos para a vaga

Fonte: Autoria Própria, 2024.

O próximo wireframe de baixa fidelidade apresentará a página que será exibida caso a empresa opte por visualizar um candidato ao clicar em “Visualizar”, conforme mencionado anteriormente. Nessa tela, serão exibidas as informações do perfil do candidato. Além disso, a empresa terá a possibilidade de entrar em contato com o candidato ou adicioná-lo à vaga desejada, por meio das opções disponíveis na seção de ações.

Figura 119 – Wireframe de baixa fidelidade: Visualizar perfil PCD

Fonte: Autoria Própria, 2024.

A seguir teremos o wireframe de alta fidelidade onde a empresa poderá realizar as funções dito anteriormente, e caso queira poderá visualizar outros candidatos que que tenha similaridade com o perfil pesquisado, podendo ser vista essa função no lado esquerdo da tela

Figura *120* – Wireframe de alta fidelidade: Visualizar perfil PCD

Fonte: Autoria Própria, 2024.

Na página inicial (Home), a empresa também poderá visualizar as vagas que criou e os candidatos que se candidataram a essas vagas. Ou seja, ela poderá visualizar suas vagas e quem se candidatou em sua vaga ou caso queira ela poderá editar suas vagas acrescentando informações adicionais a sua vaga, mas caso não esteja satisfeito com sua vaga a companhia poderá excluir seu registro de maneira eficiente.

Figura 121 – Wireframe de baixa fidelidade: processo da empresa

Fonte: Autoria Própria, 2024.

Agora, no wireframe de alta fidelidade, apresentaremos uma tela onde serão exibidas as vagas fornecidas pela empresa. Ao clicar em um botão, será possível visualizar os candidatos que aceitaram a vaga, permitindo assim avaliações detalhadas dito anteriormente. Ou editar ou até mesmo a exclusão desta vaga

Figura 122 – Wireframe de alta fidelidade: processo da empresa

Fonte: Autoria Própria, 2024.

O wireframe de baixa fidelidade apresenta o fluxo de edição de vagas pelas empresas. Caso a empresa opte por editar uma vaga já cadastrada, ela deverá clicar na opção de edição de vaga previamente fornecida na interface. Ao selecionar essa opção, a empresa será redirecionada para uma nova página dedicada exclusivamente à edição da vaga. Nessa nova interface, todos os campos relevantes serão exibidos de forma clara e organizada, permitindo que a empresa atualize facilmente as informações da vaga de maneira direta e eficiente, sem distrações ou informações desnecessárias. O design minimalista e funcional deste wireframe garante que o processo de edição seja simples e intuitivo.

Figura 123 – Wireframe de baixa fidelidade: Edição de vaga

Fonte: Autoria Própria, 2024.

Na versão de alta fidelidade, conseguimos visualizar com maior clareza como a interface final será implementada. A tela de edição de vagas apresenta uma organização intuitiva, onde as informações essenciais da vaga poderão ser facilmente modificadas. Além das alterações básicas, a interface também oferece opções avançadas, como a possibilidade de definir se a vaga está aberta ou fechada para novos candidatos. Caso a vaga já tenha sido preenchida, a empresa poderá fechá-la com apenas alguns cliques, simplificando o gerenciamento das vagas. Esse design prático e eficiente visa facilitar o processo para ambos os lados: as empresas, que terão controle total sobre suas vagas, e os candidatos, que encontrarão informações atualizadas e precisas de maneira mais rápida.

Figura 124 – Wireframe de alta fidelidade: Edição de vagas

Fonte: Autoria Própria, 2024.

Após acessar os candidatos interessados em uma vaga, a empresa terá a possibilidade de visualizar os candidatos que se inscreveram na vaga, incluindo os documentos enviados. Além disso, a empresa poderá decidir se deseja aprovar ou rejeitar o candidato para a vaga, permitindo uma avaliação criteriosa sobre a compatibilidade do candidato com os requisitos do cargo, caso deseje empresa poderá entrar em contato através do chat fornecida pelo sistema.

Figura 125 – Wireframe de baixa fidelidade: Visualizar candidatos que aceitaram a vaga

Fonte: Autoria Própria, 2024.

Agora, no wireframe de alta fidelidade, será estabelecida a exibição dos candidatos após a empresa acessar a área de visualização dos candidatos que aceitaram a vaga.

Figura 126 –Wireframe de alta fidelidade: Visualizar candidatos que aceitaram a vaga

Fonte: Autoria Própria, 2024.

O wireframe de baixa fidelidade a seguir apresenta as informações do perfil do usuário, com opções para editar o perfil ou realizar algumas configurações do perfil.

Figura 127 – Wireframe de baixa fidelidade: perfil empresa

Fonte: Autoria Própria, 2024.

Após a explicação do wireframe de baixa fidelidade, apresentaremos o wireframe de alta fidelidade, que mostrará com mais detalhes o perfil da empresa. Este wireframe incluirá informações sobre a empresa, além de um botão para editar o perfil ou realizar o logout na parte de “config”.

Figura 128 – Wireframe de alta fidelidade: perfil empresa

Fonte: Autoria Própria, 2024.

Quando a empresa decidir alterar algumas de suas informações dentro do nosso sistema ela poderá acessar a área de configuração dita anteriormente onde a companhia será redirecionada para a página de edição de suas informações normalmente.

Figura 129 – Wireframe de baixa fidelidade: Edição de perfil empresa

Fonte: Autoria Própria, 2024.

Com o wireframe de alta fidelidade, é possível compreender como a empresa poderá visualizar e editar suas informações de forma clara e eficiente. A interface final é desenhada para proporcionar uma experiência intuitiva, onde todos os dados essenciais da empresa estarão organizados e acessíveis. O layout permite que a companhia veja de maneira rápida e precisa suas informações cadastrais, facilitando a edição de qualquer campo conforme necessário.

Figura 130 – Wireframe de alta fidelidade: Editar perfil empresa

Fonte: Autoria Própria, 2024.

A seguir, vamos discutir os wireframes do aplicativo, com o objetivo de garantir que pessoas com deficiência possam acessar suas funcionalidades de maneira mais tranquila e eficiente.

O wireframe abaixo ilustra uma interface em baixa fidelidade que visa proporcionar uma experiência de usuário mais intuitiva e acessível para pessoas com deficiência (PcD). Após o login, o aplicativo exibirá a foto de perfil do usuário e disponibilizará funcionalidades que são acessadas diretamente por meio dos seguintes botões:

Vagas: Visualizar as vagas disponíveis de maneira simples e rápida.

Documentos: Editar facilmente os documentos enviados às empresas.

Processo: Visualizar o status do seu processo de candidatura e enviar os documentos necessários.

Perfil: Acesse e edite seu perfil pessoal registrado no sistema.

Empresas: Navegar pelas empresas cadastradas no sistema para conhecer mais sobre elas.

Sair: Faça logout com segurança e eficiência.

Figura 131 – Wireframe de baixa: Home-APP

Fonte: Autoria Própria, 2024.

Agora, passaremos a discutir o wireframe de alta fidelidade, que apresenta o design com maior precisão. Este estágio inclui as cores e detalhes visuais que estavam representados de forma simplificada nos wireframes de baixa fidelidade.

Figura 132 – Wireframe de alta fidelidade: Home-APP

Fonte: Autoria Própria, 2024.

A seguir será demonstrado o wireframe da tela inicial do aplicativo onde demonstrará uma tela de primeira impressão do aplicativo.

Figura 133 – Wireframe de baixa fidelidade: Início-APP

Fonte: Autoria Própria, 2024.

Agora poderá ser notado o wireframe de alta onde é mostrado a tela com mais qualidade e detalhes.

Figura *134* – Wireframe de alta fidelidade: Início-APP

Fonte: Autoria Própria, 2024.

No próximo wireframe, será detalhada a área de login do sistema, que permite os portadores de deficiência realizarem o acesso ao aplicativo. Vale ressaltar que, para entrar no aplicativo, é necessário ter previamente criado uma conta no site DaeLink.

Figura 135 – Wireframe baixa fidelidade: Login

Fonte: Autoria Própria, 2024.

Agora, no wireframe de alta fidelidade, apresentaremos de maneira mais detalhada a página de login discutida anteriormente. Este wireframe mostrará o design final, incluindo todas as cores, tipografias, e elementos visuais que compõem a interface.

Figura 135 – Wireframe de Alta fidelidade: Login

Fonte: Autoria Própria, 2024.

No instante em que a pessoa com necessidades especiais manifestar interesse em visualizar as vagas disponíveis no sistema, será possível utilizar a funcionalidade para identificar oportunidades compatíveis com seu perfil e, dessa forma, realizar a candidatura.

Figura 136 – Wireframe de baixa fidelidade: Vagas Disponíveis

Fonte: Autoria Própria, 2024.

Posteriormente, será apresentado o wireframe completo, abordando as cores e os componentes discutidos anteriormente, porém com mais detalhes, de modo a demonstrar um entendimento aprofundado da estrutura da tela.

Figura 137 – Wireframe de Alta: Visualizar vagas

Fonte: Autoria Própria, 2024.

Ao selecionar a opção de visualização da vaga, será exibida a página correspondente, cujo objetivo é permitir que o candidato insira as informações necessárias para se candidatar à vaga selecionada, iniciando, assim, o processo de inscrição.

Figura 138 – Wireframe de baixa fidelidade: Vaga-APP

Fonte: Autoria Própria, 2024.

O wireframe de alta fidelidade mostra de como ficará a página ao pcd no aplicativo ocasionando uma maior interação com o usuário.

Figura 139 – Wireframe de alta fidelidade: Vaga-APP

Fonte: Autoria Própria, 2024.

Seguindo para o próximo wireframe de baixa fidelidade, veremos a área de processos, que o usuário acessa para gerenciar suas inscrições. Nessa página, a pessoa com deficiência (PcD) será redirecionada para visualizar as vagas em que se inscreveu, com detalhes específicos sobre cada uma. Além disso, a interface fornecerá a opção de enviar documentos ao clicar em Documentos, entrar em contato com a empresa ou visualizar sua apuração caso já tenha enviado seu documento.

Figura 140– Wireframe de baixa fidelidade: Candidaturas do PCD

Fonte: Autoria Própria, 2024.

Tratando agora sobre o de alta fidelidade teremos uma interface finalizada, demonstrado as funcionalidades citadas anteriormente.

Figura 141 – Wireframe de Alta fidelidade: Candidaturas já realizadas

Fonte: Autoria Própria, 2024.

Quando o usuário clicar em "Documentos" tratado anteriormente, ele será redirecionado para uma nova página, onde o documento será emitido para a análise da empresa. A empresa avaliará se o candidato atende aos requisitos da vaga e tomará a decisão de aceitá-lo ou não em sua equipe, mas caso o usuário já tenha enviado o documento isso não será realizado pois será redirecionado para a tela de edição e não de emissão de documento.

Figura 142 – Wireframe de baixa fidelidade: Enviar documento

Fonte: Autoria Própria, 2024.

Desta maneira podemos notar essas funcionalidades de maneira mais eficiente no wireframe de alta fidelidade a seguir.

Figura 143 – Wireframe de Alta fidelidade: Enviar documento

Fonte: Autoria Própria, 2024.

Caso já tenha um documento registrado, o sistema redirecionará para uma tela onde será permitido alterar as informações já cadastradas, modificando e permitindo atualização.

Figura *144* – Wireframe de baixa fidelidade: Documento-APP

Fonte: Autoria Própria, 2024.

Logo a seguir temos o wireframe de alta fidelidade que possui como objetivo tratar com mais eficácia as cores e resultado da página móvel.

Figura *145* – Wireframe de alta fidelidade: Documento-APP

Fonte: Autoria Própria, 2024.

No wireframe a seguir terá como intuito representar o chat de baixa fidelidade demonstrando como seria o chat para que a pessoa com necessidades especiais possa entrar em contato com a empresa.

Figura 146 – Wireframe de baixa fidelidade: Chat

Fonte: Autoria Própria, 2024.

Neste wireframe a seguir abordará o chat em alta fidelidade mostrando o funcionamento do chat dentro no aplicativo.

Figura 147 – Wireframe de Alta fidelidade: Chat

Fonte: Autoria Própria, 2024.

Dando continuidade às funções disponíveis na página inicial (Home), o botão "Perfil" redireciona o usuário para sua página de perfil, onde as informações cadastradas no site serão exibidas. Caso o usuário deseje modificar qualquer dado, ele poderá fazê-lo ao selecionar a opção "Editar Perfil", a qual será explorada em detalhes posteriormente.

Figura 148─ Wireframe de baixa fidelidade: Perfil Portador deficiência

Fonte: Autoria Própria, 2024.

No wireframe de alta fidelidade, a página de perfil será apresentada com todos os detalhes visuais, incluindo a foto de perfil e um plano de fundo personalizado. Além disso, todas as informações anteriormente citadas. Este wireframe proporciona uma visão completa da aparência final do perfil do usuário no aplicativo.

Figura 149 – Wireframe de Alta fidelidade: Perfil PCD

Fonte: Autoria Própria, 2024

O wireframe de baixa fidelidade para a tela de edição de perfil exibe as informações do perfil da pessoa com deficiência (PcD) em caixas de texto editáveis. Esse formato possibilita ao usuário modificar os dados e reenviar as alterações para o sistema. Esta abordagem garante que as edições possam ser realizadas de forma prática e eficiente.

Figura 150 – Wireframe de baixa fidelidade: Editar documento

Fonte: Autoria Própria, 2024.

No próximo wireframe pode-se notar a tela finalizada, onde a pessoa com necessidade especial, poderá alterar seu documento já enviado para empresa.

Figura 151 – Wireframe de Alta fidelidade: Editar documento

Fonte: Autoria Própria, 2024

A seguir será demonstrado uma tela de carregamento, onde será exibida sempre que o sistema estiver processando dados extensos. O modelo de wireframe abaixo demonstra essa tela de carregamento, que serve para notificar o usuário de que o sistema está realizando o processamento necessário para renderizar as informações.

Figura 152 – Wireframe de baixa fidelidade: Load

Fonte: Autoria Própria, 2024

O wireframe a seguir tem como intuito de determinar a página de carregamento de informações dentro do aplicativo. Sendo o foco do wireframe demonstrar com maiores detalhes a tela.

Figura 153 – Wireframe de Alta fidelidade: Load

Fonte: Autoria Própria, 2024

CONSIDERAÇÕES FINAIS

O objetivo deste trabalho, consiste na integração social de um grupo marginalizado (Pessoas com Deficiência) por meio da empregabilidade, em que a plataforma serve de método comunicativo para o processo seletivo e de contratação. Sendo constituído para atender a demanda empresarial de cotas estabelecidas, junto da necessidade social das pessoas com deficiência. Tornando-se um sistema comunicativo, que visa a realização desses dois meios fundamentais.

Enquanto estava sendo realizada a criação do projeto, foram levantando aspectos funcionais para que a comunicação funcionasse de forma adequada como: Utilização por parte da empresa, adequadação as necessidades especiais de cada pessoa, e analisar o critério jurídico e encargos da plataforma para delimitar as funções do sistema. Assim com a análise desses pontos, foi desenvolvido um sistema que se adeque a eles, somado a detalhes e funcionalidades específicas que são os destaque e diferenciais do sistema.

Conforme abordado anteriormente, durante a realização do sistema foi abordada delimitações que estão diretamente associados a uma amplitude de ações, fazendo que ele se torne exclusivo apenas para contato profissional entre partes distintas. Fazendo que embora cumpra com seu objetivo final, possua um caminho a ser seguido para uma evolução tecnológica adicionando mais funcionalidades para abordar esses quesitos.

Havendo esse rumo de atualizações pré-estabelecido de forma consensual, conceitos e funcionalidades intercomunicativos entre os próprio PCD (Pessoa com Deficiência) ao exemplo de redes sociais, ou elementos analíticos para as empresas otimizarem vagas conforme a necessidade.

Mesmo com esses diversos pontos debatidos, o sistema não possui resultados gerais devido a ele não haver sido aplicado. Entretanto essa análise trouxe como possível resultado a diminuição do descumprimento dessas vagas, fazendo que desta forma, ocorra a inclusão social por meio do trabalho e de forma digna.

Com todos esses pontos levantados, junto da perspectiva gerada durante o desenvolvimento do projeto, o grupo em consenso acredita que o projeto possa ajudar na inclusão social das pessoas com deficiência devido a ele se adequar a padrões

atuais de mercado. E o desejo interno de que o projeto possa se tornar uma referência para outros projetos que visem inclusões sociais por meio da tecnologia. ￼

REFERÊNCIAS

AXIOS. **Introdução** [S.l], Axios, 2024. Disponível em: https://axios-http.com/ptbr/docs/intro. Acessado em: 07 jun 2024.

BARSOTI, N.; GIBERTONI, D. Impacto que o Sequelize traz para o desenvolvimento de uma API construída em Node.js com Express.js. Revista Interface Tecnológica, v. 17, n. 2, p. 231–243, 18 dez. 2020.

BRASIL. Lei nº 8.213, de 24 de julho de 1991. **Dispõe sobre os planos de benefícios da previdência social e dá outras providências. Da finalidade e dos princípios básicos da previdência social.** Disponível em: https://www.planalto.gov.br/ccivil_03/leis/l8213cons.htm. Acesso em: 7 jun. 2024.

CASA GRANDE, C.; TANAKA, S. Comparação entre o desempenho de aplicações para smartphones desenvolvidas em Flutter e React Native: **uma análise utilizando algoritmos de ordenação. Revista Terra & Cultura**: Cadernos de Ensino e Pesquisa, v. 39, n. especial, p. 7-17, 2023. Disponível em: http://periodicos.unifil.br/index.php/Revistateste/article/view/2796/2559. Acesso em: 05 maio 2024.

CAIUT, Fabio. **Administeação de Banco de Dados**. Rio de Janeiro: Escola Superior de Redes Rnp, 2015. 186 p.

CAYRES, Paulo Henrique. Modelagem de Banco de Dados. Rio de Janeiro: Escola Superior de Redes Rnp, 2015. 186 p.

CHEN, Daniel Y. **Análise de dados com Python e Pandas**. São Paulo: Novatec Editora, 2018. 432 p.

CLASEN, Beatriz. TailwindCSS e Next.js: **dominando o desenvolvimento web com estilo.** Disponível em: https://blog.rocketseat.com.br/tailwindcss-e-next-js-dominando-o-desenvolvimento-com-estilo/. Acesso em: 30 abr. 2024.

CNN BRASIL. **Lei de cotas para pessoas com deficiência faz 30 anos neste sábado.** Disponível em: https://www.cnnbrasil.com.br/nacional/lei-de-cotas-para-pessoas-com-deficiencia-faz-30-anos-neste-sabado/. Acesso em: 15 jun. 2024.

DOMINGOS, Pedro. **O Algoritmo Mestre**: como a busca pelo algoritmo de machine learning definitivo recriará nosso mundo. São Paulo: Novatec, 2017. 344 p.

DUARTE JÚNIOR, Luiz Fernando. **Programação Web com Node.js**. 2. ed. Gravataí, Rs: Edição do Autor, 2022. 394 p.

DUCKETT, Jon**. HTML e CSS**: projete e construa websites. São Paulo: Alta Books, 2016. 512 p.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistema de banco de dados**. 6. ed. Universida do Texas em Arlington: Pearson Universidades, 2011. 788 p.

ESCUDELARIO, Bruna de Freitas; PINHO, Diego Martins de. **React Native**: desenvolvimento de aplicativos mobile com react. São Paulo,Sp-Brasil: Casa do Código, 2020. 189 p.

EXPRESS. **Framework web rápido, flexível e minimalista para Node.js**. [S.l.]: Express, 2024. Disponível em: https://expressjs.com/pt-br/. Acesso em: 7 jun. 2024.

FIREBASE. **Aprenda os fundamentos.** [S.l]. Firebase, 2024. Disponível em: https://firebase.google.com/docs?hl=pt-

br&_gl=1*18pchen*_up*MQ.&gclid=504d29a01bc01aa2d6394f92b0c2ad51&gclsrc=3p.ds. Acesso em 07 jun 2024.

FLANAGAN, David. **Javascript**: o guia definitivo. 6. ed. Porto Alegre: Bookman, 2012. 1080 p.

FOWLER, Martin; TORTELLO, João. **UML Essencial**: um breve guia para a linguagem-padrão de modelagem de objetos. 3. ed. Porto Alegre: Bookman, 2005.

G1. **Brasil tem 18,6 milhões de pessoas com deficiência, cerca de 8,9% da população**, segundo IBGE. 2023. Disponível em: https://g1.globo.com/economia/noticia/2023/07/07/brasil-tem-186-milhoes-de-pessoas-com-deficiencia-cerca-de-89percent-da-populacao-segundo-ibge.ghtml. Acesso em: 14 maio 2024.

G1. **O desafio de usar tecnologia na inclusão de pessoas com deficiência. Pequenas Empresas & Grandes Negócios**, 11 dez. 2017. Disponível em: https://g1.globo.com/economia/pme/pequenas-empresas-grandes-negocios/noticia/2017/12/o-desafio-de-usar-tecnologia-na-inclusao-de-pessoas-com-deficiencia.html. Acesso em: 13 maio 2024.

G1. **Deficientes enfrentam dificuldades para ocupar vagas de trabalho reservadas por lei**. G1, 11 set. 2019. Disponível em: https://g1.globo.com/pe/pernambuco/noticia/2019/09/11/deficientes-enfrentam-dificuldades-para-ocupar-vagas-de-trabalho-reservadas-por-lei.ghtml. Acesso em: 20 maio 2024.

GIL, Antonio Carlos. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2002. 175p.

GRANDE, Carine Casa; TANAKA, Simone Sawasaki. Comparaçao entre o desempenho de aplicações para smartphones desenvolvidas em Flutter e React Native: uma análise utilizando algoritmos de ordenação. **Revista Terra & Cultura: Cadernos de Ensino e Pesquisa**, v. 39, n.T especial, p. 7-17, 16 fev. 2023. Disponível em: http://publicacoes.unifil.br/index.php/Revistateste/article/view/2796/2559. Acesso em: 05 maio 2024.

GRINBERG, Miguel. **Desenvolvimento web com Flask**: desenvolvendo aplicações web com python. São Paulo,Sp-Brasil: Novatec Editora, 2018. 312 p.

GRONER, Loiane. **Estrutura de dados e algoritmos em javaScript**: escreva um código javascript complexo e eficaz usando a mais recente ecmascript. 2. ed. São Paulo, Sp-Brasil: Novatec Editora, 2019. 408 p. Tradução: Lúcia A. Kinoshita.

GUEDES, Gilleanes T. A. **UML 2**: uma abordagem prática. 2. ed. São Paulo: Novatec Editora, 2011. 488 p.

GUEDES, Gilleanes T. A. **UML 2**: uma abordagem prática. 3. ed. São Paulo: Novatec Editora, 2018. 496 p.

INHESTA, Danielle Christina Fernandes. **Uso de aprendizado de máquina para determinar as melhores práticas de implementação de chatbots**. 2022. Trabalho de Conclusão de Curso (MBA) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2022. Disponível em: https://bdta.abcd.usp.br/directbitstream/5c19ea8d-736e-47c7-927e-9aecf5e80609/Danielle%20Christina%20Fernandes%20Inhesta_Monografia_207392

.TCC.Danielle%20Inhesta_Final%20Version_22092022.Danielle.pdf. Acesso em: 28 abr. 2024.

IBGE – **Instituto Brasileiro de Geografia e Estatística. Desemprego e informalidade são maiores entre as pessoas com deficiência**. Agência de Notícias IBGE, 22 nov. 2022. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/34977-desemprego-e-informalidade-sao-maiores-entre-as-pessoas-com-deficiencia. Acesso em: 15 maio 2024.

KNEUSEL, Ronald T. **Como a Inteligência Artificial Funciona**: da magia à ciência. São Paulo: Novatec Editora Ltda, 2024. 256 p.

LAKATOS, Eva Maria; MARCONI, Marina de Andrade. **Fundamentos de Metodologia Científica.** 8. ed. São Paulo: Atlas, 2017. 368 p.

LOWDERMILK, Travis. **Design Centrado no Usuário: um guia para o desenvolvimento de aplicativos amigáveis.** São Paulo: Novatec Editora, 2013. 184 p.

LUCIDCHART. **O que é wireframe?** Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-wireframe. Acesso em: 18 maio 2024.

MATOS, Nicole Fernandes Freitas. **Desenvolvimento e validação de Protótipos de Aplicativo para o Autocuidado de Gestantes.** 2023. 53 f. Trabalho de Conclusão de Curso (Graduação) – Curso de Engenharia da Computação, Universidade Federal do Ceará, Sobral, 2023. Disponível em: https://repositorio.ufc.br/handle/riufc/76303. Acesso em: 04 maio 2024.

MATTHERS, Eric. **Curso intensivo de python**: uma introdução prática e baseada em projetos à programação. São Paulo, Sp-Brasil: Novatec Editora, 2016. 656 p. Tradução: Lúcia A. Kinoshita.

MCKINNEY, Wes. **Python para análise de dados:** tratamento de dados com pandas, numpy e ipython. São Paulo: Novatec Editora, 2018. 616 p.

MENEZES, Nilo Ney Coutinho. **Introdução à Programação com Python**: algoritmos e lógica de programação para iniciantes. 2. ed. São Paulo, Sp-Brasil: Novatec Editora, 2014. 328 p.

MENEZES, Nilo Ney Coutinho. **Introdução à Programação com Python**: algoritmos e lógica de programação para iniciantes. 3. ed. São Paulo, Sp-Brasil: Novatec Editora, 2019. 328 p.

MIRO. **O que é um wireframe?** Como criar? Tipos, exemplos e modelos. Disponível em: https://miro.com/pt/wireframe/o-que-e-wireframe. Acesso em: 18 maio 2024.

MOZILLA. **Cross-Origin Resource Sharing** (CORS). Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/HTTP/CORS. Acesso em: 18 jun. 2024.

MORAES, William Bruno. **Construindo Aplicações com NodeJS**. 3. ed. São Paulo: Novatec, 2021. 272 p.

MUELLER, John Paul. **Começando a Programar em Python**: para leigos. 2. ed. Rio de Janeiro Rj-Brasil: Alta Books, 2020. 391 p.

NUMPY. **Quem Somos.** [s.l.]: NumPy, 2024. Disponível em: https://numpy.org/pt/about/. Acesso em: 07 jun. 2024.

PEREIRA, Luiz Antônio Demoraes. **Análise e Modelagem de Sistemas com a UML**: com dicas e exercícios resolvidos. Rio de Janeiro: Edição do Autor, 2011. 282 p.

PEREIRA, Caio Ribeiro. **Aplicações web real-time com Node.js**. [s.l.]: Casa do Código, 2013. 2 p.

PEREIRA, Adriana Soares et al. Metodologia da pesquisa científica. [s.l.]: [s.n.], 2018. 119 p.

PEREIRA, Eduardo. **Computação Evolucionária**: aplique os algoritmos genéticos com python e numpy. Flórianopolis: Casa do Código, 2020. 166 p.

POWERS, Shelley. **Aprendendo Node**: usando javascript no servidor. 2. ed. São Paulo: Novatec Editora, 2019. 312 p. Tradução em português autorizada da edição em inglês da obra Learning Node, 2nd Edition, ISBN 9781491943120 © 2016 Shelly Powers. Esta tradução é publicada e vendida com a permissão da O'Reilly Media, Inc., detentora de todos os direitos para publicação e venda desta obra.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia se Software**: uma abordagem profissional. 9. ed. Porto Alegre: Amgh Editora Ltda, 2021. 704 p. Tradução de: Francisco Araújo da Costa.

RAMAKRISHNAN, Raghu; GEHRKE, Johannes. **Sistema de Gerenciamento de Banco de Dados**. 3. ed. São Paulo,Sp-Brasil: Amgh, 2008. 905 p.

REACT. **Introdução** [S.l], React, 2024. Disponível em: https://pt-br.legacy.reactjs.org/docs/getting-started.html. Acessado em: 07 jun 2024.

REACT VITE. Visão geral. [s.l.]: **React Vite**, 2024. Disponível em: https://pt.vitejs.dev/guide/. Acesso em: 07 jun. 2024.

RNP **– Rede Nacional de Ensino e Pesquisa.** Tecnologia e acessibilidade. Disponível em:https://esr.rnp.br/temas-diversos/tecnologia-e-acessibilidade/ Acessado em 07 jun 2024.

ROCKETSEAT**. Expo e React Native**: a união que está transformando o desenvolvimento mobile. Blog Rocketseat, 09 out. 2020. Disponível em: https://blog.rocketseat.com.br/expo-react-native/. Acesso em: 09 jun. 2024.

SANTOS NETO. **Difícil inserção de pessoas com deficiência no mercado de trabalho. Campinas,** 23 set. 2020. Disponível em: https://unicamp.br/unicamp/ju/noticias/2020/09/23/dificil-insercao-de-pessoas-com-deficiencia-no-mercado-de-trabalho/. Acessado em: 14 maio 2024.

SCHEIDT, F. A. **Fundamentos de CSS: criando design para sistemas web**. [s.l.] Outbox Livros Digitais, 2015. 6 p.

SCHMITZ, Daniel; GEORGII, Daniel Pedrinha. **React – Guia do Iniciate**: domine a biblioteca javascript utilizada pelo facebook e instagram. São Paulo: Leanpub, 2015. 51 p.

SILBERSCHATZ, Abraham; SUDARSHAN, S. Korth Henry F**. Sistema de Banco de Dados**. 5. ed. Rio de Janeiro Rj-Brasil: Editora Campus, 2006. 781 p

SILVA, M. S. **Criando Sites com HTML: Sites de Alta Qualidade com HTML e CSS**. [s.l.] Novatec Editora, 2008. 213 p.

SILVA, Maurício Samy. **Fundamentos de HTML5 e CSS3**. São Paulo: Novatec, 2015. 304 p.

SILVA, Tiago. **Flask de A a Z**: crie aplicações web mais completas e robustas em python. São Paulo,Sp-Brasil: Casa do Código, 2019. 353 p.

SILVA, Mauricio Samy. **React Aprenda Praticando:** desenvolva aplicações web reais com uso da biblioteca react e de seus módulos auxiliares. São Paulo: Novatec Editora Ltda, 2021. 240 p.

TEIXEIRA, Fabricio. **Introdução e Boas Práticas em UX Design.** São Paulo: Casa do Código, 2014. 217 p.

VASCONCELOS, Davi R.; GUERRA, Paulo T. **Ensinando Teoria da Computação com Jupyter Notebook.** In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI), 31., 2023, João Pessoa/PB. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2023. p. 9-19. ISSN 2595-6175. DOI: https://doi.org/10.5753/wei.2023.229496.