



CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

**FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS**

GIOVANNI BADINE PIZZIGHINI

PORTFÓLIO DIGITAL

Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #6e1f975f2065b46eac90f75b055b5ef9bfe7cd1bb6a0919d985b70303f901ae5
<https://valida.ae/d5b34b4e3eaf6071082401e3d370a52267f1ed9558ab5ab20>

**LINS/SP
2º SEMESTRE/2023**





CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA

PAULA SOUZA

FACULDADE DE TECNOLOGIA DE LINS PROF. ANTONIO SEABRA
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS

GIOVANNI BADINE PIZZIGHINI

PORTFÓLIO DIGITAL

Trabalho de Conclusão de Curso (TCC)
apresentado à Faculdade de Tecnologia de
Lins para a obtenção do título de Tecnólogo
(a) em Análise e Desenvolvimento de
Sistemas.

Orientador: Prof. Me. Felipe Maciel
Rodrigues.

LINS/SP
2º SEMESTRE/2023

Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #6e1f975f2065b46eac90f75b055b5ef9bfe7cd1bb6a0919d985b70303f901ae5
<https://valida.ae/d5b34b4e3eaf6071082401e3d370a52267f1ed9558ab5ab20>





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #6e1f975f2065b46eac90f75b055b5ef9bfe7cd1bb6a0919d985b70303f901ae5
<https://valida.ae/d5b34b4e3eaf6071082401e3d370a52267f1ed9558ab5ab20>

Badine Pizzighini, Giovanni

B136p Portfólio Digital / Giovanni Badine Pizzighini. — Lins, 2023.

66f.

Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) — Faculdade de Tecnologia de Lins Professor Antonio Seabra: Lins, 2023.

Orientador(a): Me. Felipe Maciel Rodrigues

1. Portfólio. 2. Digital. 3. Desenvolvimento. 4. Tecnologia. 5. Sistemas.
I. Maciel Rodrigues, Felipe. II. Faculdade de Tecnologia de Lins Professor Antonio Seabra. III. Título.

CDD 004.21

Gerada automaticamente pelo módulo web de ficha catalográfica da FATEC Lins
mediante dados fornecidos pelo(a) autor(a).





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #6e1f975f2065b46eac90f75b055b5ef9bfe7cd1bb6a0919d985b70303f901ae5
<https://valida.ae/d5b34b4e3eaf6071082401e3d370a52267f1ed9558ab5ab20>

GIOVANNI BADINE PIZZIGHINI

PORTFÓLIO DIGITAL

Trabalho de Conclusão de Curso
apresentado à Faculdade de Tecnologia de
Lins como parte dos requisitos para obtenção
do título de Bacharel em Tecnologia em
Análise e Desenvolvimento de Sistemas sob
orientação do prof. Me. Felipe Maciel
Rodrigues

Data de aprovação: ___/___/___

Orientador (Prof. Me. Felipe Maciel Rodrigues)

Examinador 1 (Prof. Me. Júlio Fernando Lieira)

Examinador 2 (Prof. Me. Diego Henrique Emygdio Lazaro)





Escaneie a imagem para verificar a autenticidade do documento
Hash SHA256 do PDF original #6e1f975f2065b46eac90f75b055b5ef9bfe7cd1bb6a0919d985b70303f901ae5
<https://valida.ae/d5b34b4e3eaf6071082401e3d370a52267f1ed9558ab5ab20>



Este trabalho vai a todos que passam e superam uma vida constante de mudanças, à minha querida família, e aos meus poucos e grandes amigos.

Giovanni Badine Pizzighini



AGRADECIMENTOS

Agradeço primeiramente a minha família, que desde o início me apoiou nessa grande mudança, minha mãe principalmente.

Todos meus poucos e sinceros amigos, que depois de tanto tempo, aparecem como uma fortaleza frente as grandes dificuldades da vida.

Todos meus queridos professores pelos ensinamentos, dedicação, disposição e conhecimento que nos proporcionou para vencer mais uma etapa de nossas vidas.

Por fim, agradeço a vida como um todo. Às oportunidades e as portas abertas para uma mudança de carreira bem-sucedida para trilhar, mais uma vez, um possível caminho de felicidade e sabedoria.

Giovanni Badine Pizzighini





RESUMO

Este portfólio tem por objetivo demonstrar os conhecimentos e atividades feitas durante os 3 anos de aprendizado na FATEC de Lins. Os capítulos estão divididos em projetos semestrais, com exceção dos capítulos 6 e 7, que representam projetos desenvolvidos no quinto semestre do curso de Análise e Desenvolvimento de Sistemas. No primeiro capítulo, tem-se a introdução descrevendo os principais projetos e as habilidades encontradas e aprendidas para efetivamente elaborá-los. Seguidamente, os projetos em ordem por semestre, sendo eles o Sistema de Gestão Escolar (IHC) no Figma, Livraria usando arquivos em C++, *website* usando *HTML* e *Bootstrap*, Agenda telefônica para *Mobile* usando *JQUERY* e Criação do Banco de Dados DocumentDB, respectivamente para o primeiro, segundo, terceiro e quarto semestres. Estes dois últimos foram resultado do adiantamento da disciplina de “Trabalho de Graduação II” ainda no quinto semestre. No último capítulo, conclui-se o trabalho explicando de forma geral as tecnologias aplicadas e os conhecimentos desenvolvidos em cada projetos, ressaltando os desafios e métodos aplicados para a finalização deste.

Palavras-Chave: desenvolvimento, projeto, portfólio, *website*, banco de dados, mobile





ABSTRACT

This portfolio aims to demonstrate the knowledge and activities carried out during the 3 years of learning at FATEC de Lins. The chapters are divided into semester projects, except for chapters 6 and 7, which represent projects developed in the fifth semester of the Systems Analysis and Development course. In the first chapter, there is an introduction describing the main projects and the skills found and learned to effectively develop them. Then, the projects in order by semester, being the School Management System (IHC) in Figma, Bookstore system using files with C++, website using HTML and Bootstrap, Phonebook for Mobile using JQUERY and Creation of the DocumentDB Database, respectively for the first, second, third and fourth semesters. These last two were the result of the advancement of the “Trabalho de Graduação II” subject in the fifth semester. In the last chapter, the work concludes by explaining in general terms the technologies applied and the knowledge developed in each project, highlighting the challenges and methods researched to complete it.

Keywords: development, project, portfolio, website, database, mobile





LISTA DE ILUSTRAÇÕES

Figura 2.1 – Contexto de uso e interface com o usuário.....	13
Figura 2.2 – Visão Geral da Plataforma.....	15
Figura 2.3 – Menu Lateral para Interação de Páginas no Figma.....	16
Figura 2.4 – Menu à direita com seleção de notas.....	17
Figura 2.5 – Tela de Login.....	17
Figura 2.6 – Tela de Apresentação.....	18
Figura 3.1 – Modelo de Structs.....	23
Figura 3.2 – Função ‘Cadastrar Livro’.....	24
Figura 3.3 – Função ‘Cadastrar Cliente’.....	24
Figura 3.4 – Função ‘Localizar Cliente’.....	25
Figura 3.5 – Função <i>Main</i>	28
Figura 4.1 – Exemplo de código em HTML.....	32
Figura 4.2 – Exemplo de código em CSS.....	33
Figura 4.3 – Página <i>Home</i> da aplicação <i>web</i>	34
Figura 4.4 – Exemplo da página 2 com imagem incorporada.....	35
Figura 4.5 – Exemplo da página 2 com <i>link</i> no rodapé.....	35
Figura 4.6 – Exemplo da página 3 com imagem incorporada na totalidade do <i>grid</i>	36
Figura 4.7 – Página 4 com exemplo de formulário.....	37
Figura 5.1 – Função #btn-add [ADICIONAR] da Agenda.....	42
Figura 5.2 – Função [EDITAR] da Agenda.....	43
Figura 5.3 – Função [ATUALIZAR] da Agenda	44
Figura 5.4 – Função [EXCLUIR] da Agenda.....	45
Figura 5.5 – Funções [ATUALIZAR] e [MOSTRAR] da Agenda.....	46
Figura 5.6 – Exibição da aplicação de Agenda Telefônica.....	47
Figura 6.1 – Interface do <i>IDE Cloud9</i> na <i>AWS</i>	50
Figura 7.1 – <i>Landing page</i> do Portfólio.....	55
Figura 7.2 – Seção “Sobre mim”.....	56
Figura 7.3 – Seção “Conhecimentos”.....	57





Figura 7.4 – Seção “Experiência”	57
Figura 7.5 – Seção “Portfólio”	58
Figura 7.6 – Modal Interface da Seção “Portfólio”	59
Figura 7.7 – Seção “Contato”	59





LISTA DE ABREVIACÕES

API - *Application Programming Interface*
AWS - *Amazon Web Services*
BJSON - *Binary JavaScript Object Notation*
CDN - *Content Delivery Network*
CRUD - *Create, Read, Update e Delete*
CSS - *Cascading Style Sheets*
DOM - *Document Object Model*
EC2 - *Elastic Compute Cloud*
HTML - *HyperText Markup Language*
IDE - *Integrated Development Environment*
IHC - *Interação Humano-Computador*
JSON - *JavaScript Object Notation*
NoSQL - *Not Only 'SQL' (Structured Query Language)*
SQL - *Structured Query Language*
UI - *User Interface*
URL - *Universal Resource Locator*
UX - *User Experience*
VPC - *Virtual Private Cloud*





SUMÁRIO

1 INTRODUÇÃO	10
2 SISTEMA DE GESTÃO ESCOLAR (SIGA)	12
3 LIVRARIA USANDO ARQUIVOS EM C++	20
4 WEBSITE USANDO HTML E BOOTSTRAP	30
5 AGENDA TELEFÔNICA PARA MOBILE USANDO JQUERY	37
6 CRIAÇÃO DE BANCO DE DADOS AWS DOCUMENTDB.....	46
7 MANUAL DO PORTFÓLIO	54
7.1 LANDING PAGE (SEÇÃO PRINCIPAL).....	55
7.2 SOBRE MIM.....	55
7.3 CONHECIMENTOS.....	56
7.4 EXPERIÊNCIA.....	57
7.5 PORTFÓLIO.....	58
7.6 CONTATO.....	59
8 CONSIDERAÇÕES FINAIS.....	68
REFERÊNCIAS BIBLIOGRÁFICAS.....	79





1 INTRODUÇÃO

Em grande parte dos estudos de ensino superior, é possível determinar os conhecimentos adquiridos por meio de Trabalhos de Conclusão de Curso (TCCs), que visam identificar as competências e habilidades do aluno frente aos desafios do mundo profissional. Entre diversas opções, destaca-se, para um conteúdo mais técnico e objetivo, o modelo de Portfólio, aceito pela instituição em questão FATEC de Lins “Prof. Antônio Seabra”, escolhido para este trabalho a fim de explorar e destacar as atividades desenvolvidas durante o curso.

O objetivo deste portfólio é demonstrar o conhecimento adquirido por este discente no curso de Análise e Desenvolvimento de sistemas nos seus 3 anos de curso, destacando as competências técnicas, disciplinas cursadas e projetos desenvolvidos em cada disciplina.

A motivação do portfólio em questão será ter um registro das atividades e competências desenvolvidas em curso como “porta de entrada” para futuros empregadores e estudantes, para que tenham compreensão do que foi aprendido na jornada de graduação.

Justifica-se, além da proposta do Portfólio como trabalho de graduação, a necessidade de avaliar e demonstrar o conhecimento adquirido, bem como compreender se o conteúdo proposto pela instituição abrange a maior parte do mundo técnico da informática e análise de sistema na conjuntura atual do mercado de trabalho e nas pesquisas desenvolvidas na área.

A metodologia utilizada será, como abarcado acima, no modelo de Portfólio, contendo um este trabalho escrito e complementado com o Portfólio digital e a organização do trabalho escrito compreenderá na divisão das atividades e habilidades desenvolvidas do 2º Semestre letivo ao 6º semestre letivo.

Deste modo, a ideia é que o trabalho resulte em um item importante no passo a passo de uma carreira de sucesso, a começar pela visibilidade e exposição das competências técnicas, tornando-se um grande diferencial na vida de qualquer profissional.

Abaixo destacam-se as atividades em questão de forma ascendente na cronologia da grade curricular:

No contexto do segundo semestre, foram desenvolvidos nas disciplinas de Sistemas de Informação um projeto de prototipação para um portal do aluno





por meio do software online *Figma*; e para a disciplina de Interação Humano Computador, o desenvolvimento de um fluxograma para a estruturação de um website.

Já no terceiro semestre, na disciplina de Estrutura de Dados, foram desenvolvidas competências nas linguagens C++ e C, em que foram construídas um banco de dados de arquivo, e um catálogo de biblioteca para acessar o acervo.

No quarto semestre, na disciplina de Programação para web, desenvolvemos um website utilizando as linguagens de *HyperText Markup Language (HTML)* e *Cascading Style Sheets (CSS)* no *Netbeans®*.

Em seguida, no quinto semestre foram desenvolvidas duas atividades, nas disciplinas de Programação para dispositivos móveis e Laboratório de banco de dados. Na disciplina de Programação para dispositivos móveis, foi desenvolvida uma aplicação para celular constituindo uma agenda telefônica utilizando a biblioteca *JQUERY* do *JavaScript* juntamente com *HTML*. Já na disciplina de banco de dados, foi criado um banco de dados não-relacional *DocumentDB* em ambiente de nuvem e inserindo registros para uma apresentação prática em sala de aula.

Para a última parte deste trabalho, há um capítulo do manual do usuário do Portfólio Digital, ou seja, o objetivo final do trabalho de conclusão de curso e descreve como foi construído os detalhes de cada seção desenvolvida.

Assim, para cada disciplina cursada nos semestres supracitados, haverá uma descrição de como o projeto foi desenvolvido, as etapas em destaque, as ferramentas utilizadas, a utilidade dentro do mundo de tecnologia e por fim, uma breve explicação de como utilizar a aplicação/modelo (se for o caso), ou então um detalhamento conclusivo do projeto para fins mais teóricos.





2 SISTEMA DE GESTÃO ESCOLAR (SIGA)

Na disciplina de *Interação Humano Computador* (IHC), fomos introduzidos os conceitos de métodos e formas de como nós, seres humanos e principalmente usuários, interagimos com as aplicações e *softwares* disponíveis no mundo da computação e da web. Ao longo do curso, experienciamos diversas formas de como podemos interagir com os mais diversos recursos disponíveis como websites, aplicações móveis (interfaces em celular) e métodos atuais de como pretendemos e atuamos ao utilizar os recursos.

Para isso, estudamos também sobre *User Interface* (UI) e *User Experience* (UX), que respectivamente, tratam sobre como os produtos estão disponíveis – no caso, sua apresentação gráfica - para o uso e como interagir com ele; e sobre como vamos interagir com o produto, ou seja, ter uma experiência satisfatória ao utilizar o produto.

Ao final da disciplina, fomos requeridos a apresentar cerca de 2 a 4 páginas de uma aplicação / site com um sistema de gestão escolar, muito parecido com o SIGA, da Fatec-SP, onde os alunos podem consultar notas, faltas, solicitar documentos, fazer matrícula etc.

2.1 CONCEITUAÇÃO

Em uma definição breve, podemos conceituar as interações humano-computador da seguinte forma, de acordo com Hewett:

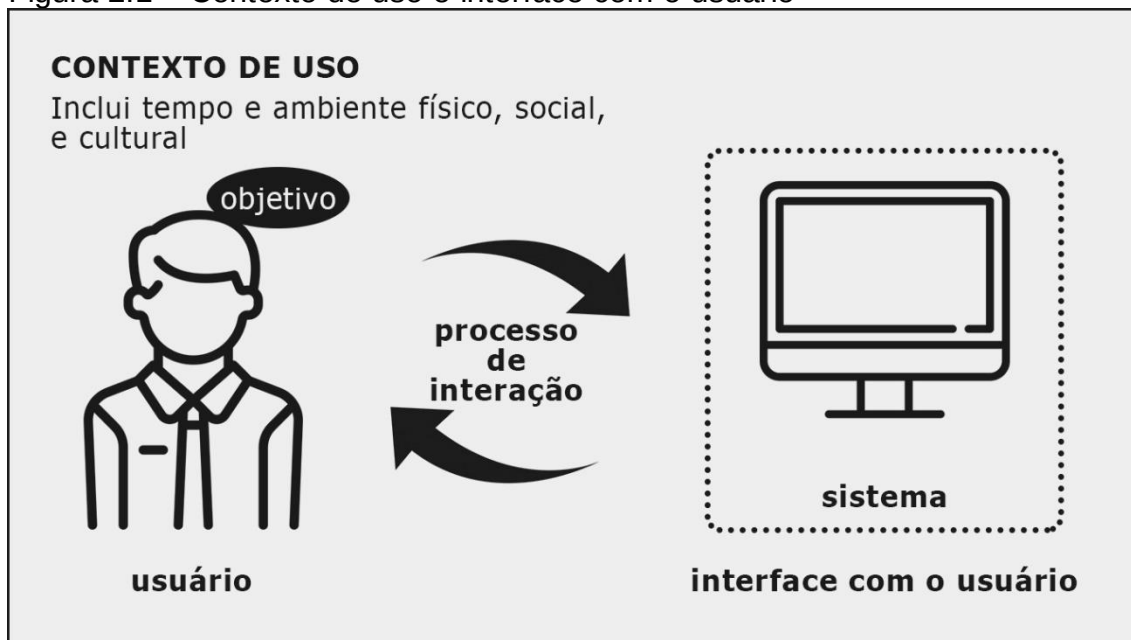
Interação Humano-Computador (IHC) é uma disciplina voltada para o projeto, avaliação e implementação de sistemas computacionais interativos para uso humano e com o estudo de fenômenos importantes que os rodeiam. (HEWETT et al., 1992, p. 5).

Dentre os elementos interativos, existem outros conceitos fundamentais na disciplina que demonstram as funções de cada um desses elementos, e, principalmente, o **contexto de uso**. Abaixo, pode-se observar um quadro que representa o contexto de uso:





Figura 2.1 – Contexto de uso e interface com o usuário



Fonte. Chief of Design. 2023

No caso acima, representação a interação do usuário por ter um objetivo ao decidir usar o produto ou sistema, e o processo de interação se dá por meio da interface do usuário (UI). A partir do início dessa interação, o usuário busca atingir os objetivos de uso, e para que isso aconteça, também surge o sentido de como será sua experiência (UX), ou seja, a resposta emocional do usuário ao sistema, para que ela seja satisfatória e eficiente.

De acordo com Nielsen, a usabilidade é um aspecto fundamental da UI e da UX. Segundo ele a usabilidade é um atributo de qualidade que se relaciona à facilidade de uso e aprendizado de um produto ou sistema (NIELSEN, 1993).

Deste modo, a UI deve ser projetada de forma a facilitar a interação do usuário com o sistema, reduzindo a necessidade de aprendizado e minimizando erros e frustrações.

Hewett et al. (1992), por sua vez, destaca a importância de uma abordagem centrada no usuário para o desenvolvimento de sistemas interativos.

De acordo com o autor, a melhor forma de criar sistemas que atendam às necessidades dos usuários é por meio de um processo iterativo de design que envolve a participação ativa dos usuários (HEWETT et al., 1992).

Aplicando os conceitos de UI e UX, deu-se início ao protótipo de construção do sistema de gestão com orientação do professor Felipe Maciel Rodrigues, por meio da ferramenta de prototipação Figma, disponível na rede.





Definindo acima, prototipação é um processo utilizado no design e desenvolvimento de produtos, serviços ou sistemas, que envolve a criação de representações iniciais do produto final para validar conceitos, explorar soluções e obter feedback dos usuários. Esses protótipos podem variar em termos de fidelidade, desde protótipos de baixa fidelidade, que são rápidos e simples, até protótipos de alta fidelidade, que se assemelham mais ao resultado em termos de aparência e funcionalidade.

Existem diferentes tipos de protótipos comumente utilizados:

- *Rabiscos* ou *Sketches*: são protótipos de baixa fidelidade que envolvem esboços rápidos e rudimentares para visualizar conceitos iniciais.
- *Wireframes*: são representações estruturais de uma interface, geralmente em preto e branco, que mostram a organização dos elementos, a hierarquia visual e o fluxo de interação do usuário.
- *Mockups*: são protótipos de média a alta fidelidade que se assemelham visualmente ao produto final, incluindo cores, tipografia, imagens e layout. Eles fornecem uma visão mais precisa de como o produto será.
- Protótipos interativos: são protótipos funcionais que permitem a interação do usuário, simulando o comportamento do produto final. Podem incluir animações, transições e elementos interativos.

Neste contexto, o Figma é um editor gráfico de vetor e prototipagem de projetos de design baseado principalmente no navegador web, com ferramentas *offline* adicionais para aplicações desktop para *GNU/Linux*, *macOS* e *Windows* (*Wikipedia*. Figma. Disponível em: <https://en.wikipedia.org/wiki/Figma>. Acesso em: 08 de maio 2023).

É um software focado no desenvolvimento de sistemas de design gráfico, prototipagem de interface gráfica de usuário e desenvolvimento de *UI/UX* permitindo também o desenvolvimento colaborativo em tempo real com outros usuários remotamente (GONZALEZ, WIRED, 2020).





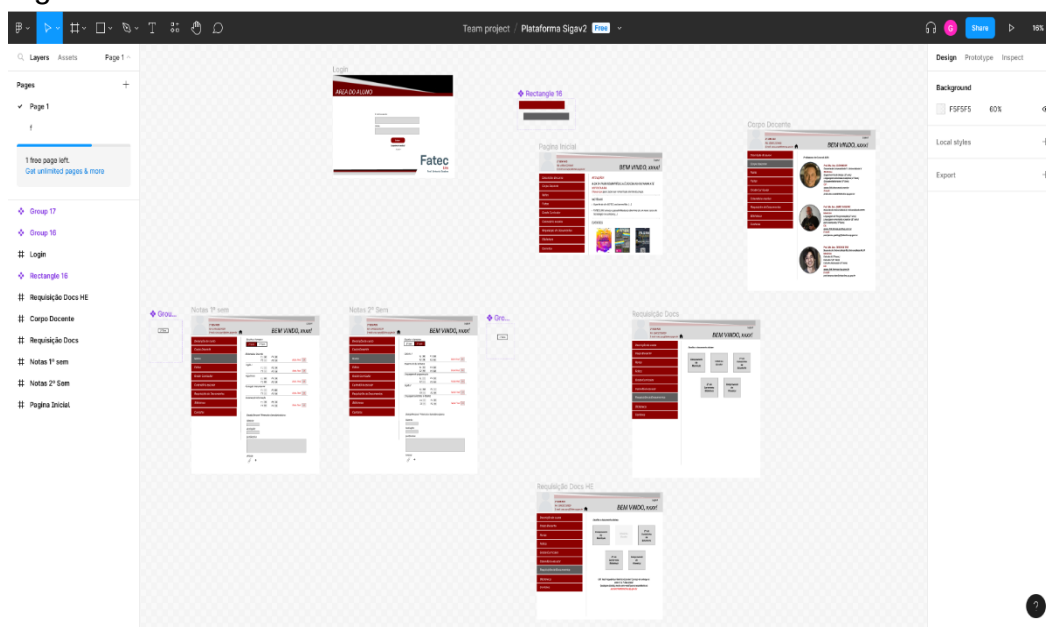
2.2 DESENVOLVIMENTO

Aplicando parte do conhecimento adquirido em sala de aula, juntamente com testes na aplicação web Figma, deu-se início à prototipação do trabalho da disciplina.

Como primeiros passos, deveríamos definir quais as páginas, qual o formato (neste caso, foi usado o layout para Tablet) e usabilidade das ferramentas que iríamos produzir, pensando sempre em *UI/UX* para garantir o contexto de uso e atingimento de objetivos pelo suposto usuário da aplicação.

Deu-se início definindo que as páginas de interação seriam a de *login* do usuário, a consulta de notas, a consulta do corpo docente e a requisição de documentos. No total, foram confeccionadas 7 páginas de interação conforme figura 2.2:

Figura 2.2 – Visão Geral da Plataforma



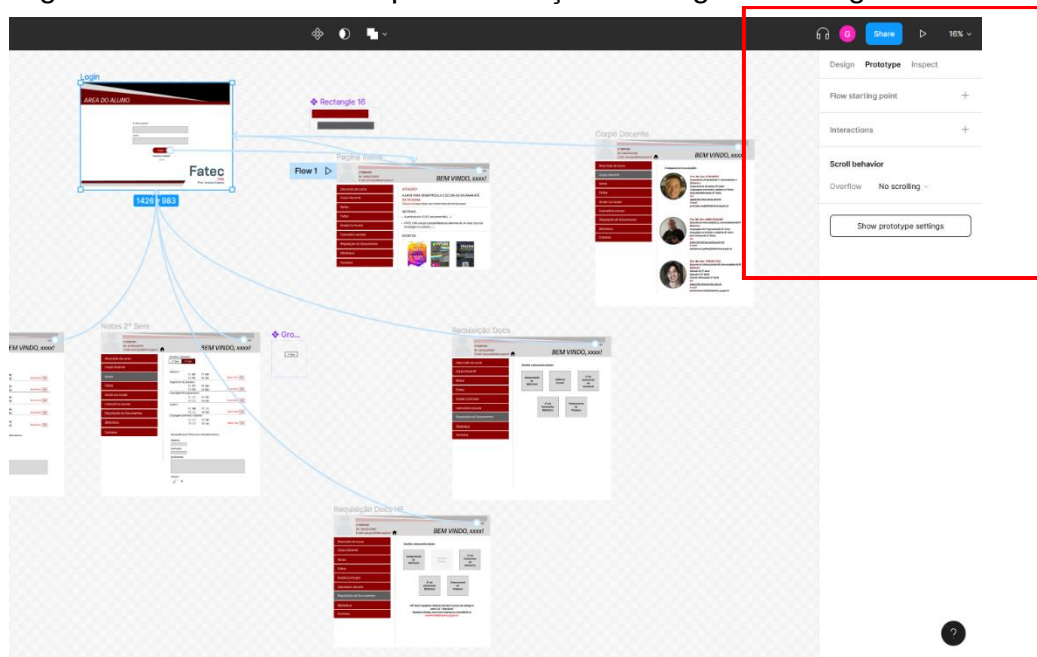
Fonte. Elaborado pelo autor, 2023

A ferramenta Figma permite edição de cada página em separado e a interação entre elas por meio de “*canvas de relacionamento*”, ou de “*interação*”, para que fosse possível conectar as páginas de acordo com a sequência de *login* para as outras páginas que poderiam ser interagidas pelo usuário.





Figura 2.3 – Menu Lateral para Interação de Páginas no Figma



Fonte. Elaborado pelo autor. 2023

Para a criação de cada página, foram considerados elementos vistos em sala de aula, além dos imprescindíveis *UI/UX*. Como objetivo principal era levar uma experiência eficiente para o suposto aluno-usuário da aplicação, onde poderia encontrar os elementos fáceis e com uma usabilidade fluida e simples. Para tal, usou-se itens de interação como botões e animações de troca de páginas, com um menu lateral com mudança de cor:





Figura 2.4 – Menu à direita com seleção de notas.

The image shows a web interface for a student's grade selection. On the left, a vertical menu lists various options: 'Descrição do curso', 'Corpo Docente', 'Notas' (highlighted with a red box), 'Faltas', 'Grade Curricular', 'Calendário escolar', 'Requisição de Documentos', 'Biblioteca', and 'Contatos'. The main content area is titled 'BEM VINDO, xxxx!' and features a 'Logout' link. Below the title, there's a section for 'Escolha o Semestre:' with radio buttons for '1ª Sem' and '2ª Sem'. The '1ª Sem' option is selected. Below this, there are tables for various subjects, each showing two semesters (T1, T2) and two periods (P1, P2) with their respective grades. The subjects listed are Matemática Discreta, Inglês I, Algoritmos, Português Instrumental, and Sistemas de Informação. Each subject has a 'Nota Final' (Final Grade) displayed in a red box. At the bottom, there's a section for 'Deseja Recorrer? Preencha o formulário abaixo:' with fields for 'Materia:', 'Avaliação:', 'Justificativa:', and 'Anexos:'.

Matemática Discreta	T1	P1	T2	P2	Nota Final
	6,0	8,0	-	9,0	7,5

Inglês I	T1	P1	T2	P2	Nota Final
	-	3,0	-	4,0	3,0

Algoritmos	T1	P1	T2	P2	Nota Final
	8,0	9,0	8,5	9,0	8,5

Português Instrumental	T1	P1	T2	P2	Nota Final
	-	10,0	-	9,0	9,5

Sistemas de Informação	T1	P1	T2	P2	Nota Final
	7,0	3,0	5,5	7,5	6,5

Fonte. Elaborado pelo autor, 2023

Assim, partindo da página inicial abaixo e em seguida acessando as outras páginas por meio do menu lateral, é possível navegar pela plataforma fazendo as devidas requisições nas páginas de acordo com os objetivos do usuário:

Figura 2.5 – Tela de Login

The image shows the login page for the Fatec Lins student area. The header features the text 'AREA DO ALUNO' in white on a dark red background. Below the header, there are two input fields for 'E-mail ou usuário' and 'Senha'. A red 'Entrar' button is positioned below the password field. Below the button, there are links for 'Esqueceu a senha?' and 'Ajuda?'. The footer of the page displays the 'Fatec Lins' logo and the name 'Prof. Antonio Seabra'.

Fonte. Elaborado pelo autor, 2023





Figura 2.6 – Tela de Apresentação



Fonte. Elaborado pelo autor, 2023

2.3 CONCLUSÃO

Deste modo, usando os conhecimentos em aula, os testes na ferramenta e principalmente, manuseando como aprendizado o Figma, foi possível criar uma prototipação de sistema *web* de gestão escolar que atendesse a atividade proposta pelo professor e com os objetivos determinados pela disciplina de IHC.

Primeiramente, levando em consideração toda a construção do racional de como um sistema de gestão escolar deve ser feito para garantir eficiência de uso e simplicidade para o aluno-usuário, com alguns dos elementos necessários disponíveis em uma universidade ou escola para acesso dos alunos.

Adicionalmente, levar em consideração a estética e a aparência visual da *UI* também são importantes, inclusive para a *UX*. Elementos como cores, fontes e layout devem ser cuidadosamente selecionados de forma a criar uma identidade visual atraente e coerente. No entanto, é importante lembrar que a estética não deve ser priorizada em detrimento da funcionalidade e da usabilidade.

Em suma, uma *UI* bem projetada e uma *UX* satisfatória são fundamentais para o sucesso de um software ou sistema interativo. Isso requer uma





abordagem centrada no usuário, com foco na usabilidade, na estética e na experiência emocional do usuário.





3 LIVRARIA USANDO ARQUIVOS EM C++

Na disciplina de Estrutura de Dados, no terceiro semestre letivo, tivemos os componentes em C/C++ sobre como criar estrutura de agrupamento de variáveis, mais conhecidas como *struct*, para posteriormente criar um simples banco de dados de uma Livraria usando essas estruturas e manipulando entradas e saídas dos arquivos registrados, completando assim o trabalho.

Mesmo com outros artifícios hoje em termos de banco de dados e registros, a programação em C desses elementos se vê útil no aprendizado pois mostra componentes teóricos importantes em linguagem de médio nível e garante um conhecimento prévio para estruturas mais complexas à nível de banco de dados, manipulação de dados, entradas e saídas etc.

Ao final da disciplina, fomos requeridos a direcionar o código para o professor Júlio Leria, em criamos as *structs*, criamos métodos de leitura, cadastro de livros, manipulação de entradas e saídas, cadastro de clientes, manipulação dos dados destes clientes, e finalizando com uma função *Main* de execução do código.

3.1 CONCEITUAÇÃO

A estrutura de dados é uma parte essencial da programação, que permite organizar e armazenar dados de forma eficiente. A linguagem C oferece recursos poderosos para lidar com estruturas de dados, como a declaração de *structs* e a manipulação de arquivos. Neste capítulo, esses tópicos serão explorados, fornecendo uma introdução à utilização de *structs*, a criação e manipulação de arquivos em C, bem como a interação com eles.

Inicialmente, faz-se necessário uma apresentação breve sobre a linguagem de programação trabalhada neste semestre, a linguagem C. Define-se abaixo:

C é uma linguagem de programação compilada de propósito geral, estruturada, imperativa, procedural, padronizada pela Organização Internacional para Padronização (ISO), criada em 1972 por Dennis Ritchie na empresa *AT&T Bell Labs* para desenvolvimento do sistema operacional Unix (originalmente escrito em *Assembly*). (Livinginternet.com. 2023)





Além disso, destaca-se a importância no mundo da programação da linguagem C:

C é uma das linguagens de programação mais populares e existem poucas arquiteturas para as quais não existem compiladores para C. C tem influenciado muitas outras linguagens de programação (por exemplo, a linguagem Java). (Cprogramminglanguage.com. 2023)

Avançando na linha do tempo, a linguagem C evolui para C++, primeiramente esta última sendo criada como uma extensão da primeira, em 1980. As diferenças entre as linguagens C e C++ são poucas, mas tiveram um grande avanço, principalmente na característica de programação orientada a objetos, como indica Calle (2017):

Um objetivo chave do projeto C++ era manter a compatibilidade com o C. A ideia era preservar a integridade de milhões de linhas de programas já escritos e depurados, a integridade de muitas bibliotecas C existentes e a utilidade de ferramentas C já desenvolvidas. Devido ao alto grau de sucesso na obtenção desse objetivo, a transição do C para o C++ é muito simples. A melhoria mais significativa da linguagem C++ é seu suporte à programação orientada para objetos (abreviadamente OOP). Neste caso, muda-se a abordagem na solução de problemas para aproveitar-se de todos os benefícios de C++, isto é, utiliza-se outra filosofia de programação (CALLE, 2017, p. 3).

Neste caso, foi-se usado a linguagem C para apresentação final do trabalho, mas as diferenças entre C e C++ são poucas, com o avanço de se usar esta última com o propósito de utilizar-se de conteúdo orientado a objeto e bibliotecas adjacentes, não usadas neste trabalho.

3.2 DESENVOLVIMENTO

Iniciando a produção do código, importante ressaltar o conceito do que é um arquivo e como foi estruturado o código para a criação da aplicação da livraria.

Arquivos são uma maneira de armazenar dados permanentemente em um sistema de arquivos. Em C, podemos criar, abrir, fechar, ler e escrever em arquivos usando funções específicas da biblioteca padrão.

Para que sejam feitas modificações no arquivo, chama-se de manipulação a parte que envolve várias operações, como criação, leitura, gravação, exclusão e busca de dados. Podemos criar um novo arquivo usando a função "*fopen*" com o modo de abertura "w" (escrita). Para ler dados de um arquivo, usamos a função





"fscanf" ou "fread". Para escrever dados em um arquivo, usamos a função "fprintf" ou "fwrite". Também podemos buscar dados específicos em um arquivo e realizar operações de atualização ou exclusão.

Todos estes métodos estiveram presentes na maioria das funções no código, pois são comandos básicos dentro do assunto de Arquivos na linguagem de programação.

O ponto de partida neste trabalho foi criar as estruturas de variáveis, chamadas de *structs*. Devemos entender o conceito de *struct*. Em um breve artigo da Escola Politécnica da PUC-RS, temos o conceito de *struct*:

Structs, também conhecidas como Registros, definem tipos de dados que agrupam variáveis sob um mesmo tipo de dado.
A ideia de usar uma struct é permitir que, ao armazenar os dados de uma mesma entidade, isto possa ser feito com uma única variável. Por exemplo, se for preciso armazenar a altura, o peso e a idade de uma pessoa, pode-se criar uma struct chamada 'Pessoa' e agrupar os dados em um único tipo de dado(...).
(PUCRS. 2023)

Portanto, as *structs* são úteis para representar entidades complexas, como registros, objetos ou estruturas de dados personalizadas. A declaração de uma struct é feita no início do código ou em um arquivo de cabeçalho.

A partir deste conceito, foram criadas várias *structs*, como 'reg_livro', 'reg_cliente' e 'reg_venda' para posteriormente usar nas funções do código para manipular esses dados. No exemplo da figura abaixo, o trecho onde esses registros foram criados:

Figura 3.1 – Modelo de *Structs*





```
6
7 struct reg_livro {
8     int codigo;
9     char titulo[30];
10    float preco;
11    int estoque; //Estoque
12 };
13
14 struct reg_cliente{
15     int codigo;
16     char nome[50];
17     char endereco[50];
18     char fone[15];
19     char email[30];
20 };
21
22 struct reg_venda{
23     int nro_pedido; //Numero do pedido da venda
24     int cod_cliente; //Codigo do Cliente que comprou o Livro
25     int cod_livro; //Codigo do Livro Vendido
26     int qtde; //Quantidade vendida
27 };
28
```

Fonte. Elaborado pelo autor. 2023

Depois de criada as *structs* para armazenar as variáveis, foram criadas as funções para registrar as variáveis de cada ação necessária para o cadastro de livros.

Estas funções são necessárias para performar as ações que irão modificar ou manipular o arquivo, e no final serem chamadas para executar o código por meio de sua função *Main*.

Dentre elas, foram criadas as funções para construir a base do arquivo, como **cadaststrarLivro()** e **cadaststrarCliente()**. Ambas as funções solicitam ao usuário que digite os dados correspondentes ao livro ou cliente, respectivamente, e salvam esses dados em arquivos binários.





Figura 3.2 – Função ‘Cadastrar Livro’

```
void cadastrarLivro(){
    FILE *fplivro;
    struct reg_livro livro;
    char opc;

    //Entrada dos dados do Livro
    printf("\nCodigo: ");
    fflush(stdin); scanf("%d",&livro.codigo);
    getchar();
    printf("Titulo: ");
    fgets(livro.titulo,30,stdin);
    printf("Preco: ");
    fflush(stdin); scanf("%f",&livro.preco);
    printf("Estoque: ");
    fflush(stdin); scanf("%d",&livro.estoque);

    printf("\nSalvar(S/N)? ");
    fflush(stdin); scanf("%s",&opc);
    if ((opc=='N')||(opc=='n')){
        printf("\nOperacao Cancelada!");
        return;
    }
    //Se confirmou, continua aqui
    //Abrir o Arquivo
    fplivro = fopen(LIVROS,"ab");
    //Gravar no Arquivo a struct com os dados do Livro
    fwrite(&livro,sizeof(livro),1,fplivro);

    printf("\nLivro Cadastrado com Sucesso.");

    //Fechar o Arquivo
    fclose(fplivro);
} //Fim cadastrarLivro()
```

Fonte. Elaborado pelo autor. 2023

Figura 3.3 – Função ‘Cadastrar Cliente’

```
void cadastrarCliente(){
    FILE *fpclientes;
    struct reg_cliente cliente;
    char opc;

    printf("\nDigite o Codigo: ");
    fflush(stdin); scanf("%d",&cliente.codigo);
    getchar();
    printf("\nNome: ");
    fgets(cliente.nome, 50, stdin);
    printf("\nEndereco: ");
    fgets(cliente.endereco, 50, stdin);
    printf("\nFone: ");
    fgets(cliente.fone, 15, stdin);
    printf("\nEmail: ");
    fgets(cliente.email, 30, stdin);

    printf("\nSalvar Cliente(S/N)? ");
    fflush(stdin); scanf("%s",&opc);
    if ((opc=='N')||(opc=='n')){
        printf("\nOperacao Cancelada!");
        return;
    }
    //Se confirmou, continua aqui
    //Abrir o Arquivo
    fpclientes = fopen(CLIENTES,"ab");
    //Gravar no Arquivo a struct com os dados do Cliente
    fwrite(&cliente,sizeof(cliente),1,fpclientes);

    printf("\nCliente Cadastrado com Sucesso.");

    //Fechar o Arquivo
    fclose(fpclientes);
} //Fim cadastrarCliente()
```

Fonte. Elaborado pelo autor. 2023





Em seguida, foram criadas outras funções de consulta no arquivos, das quais contemplam **listarLivros()**, **listarClientes()**, **consultarCodigoLivro()**, **relatorioDetalhado()**, **listarEstoqueMinimo()**, **localizarCliente()** e **localizarLivro()**, como mostra a Figura 3.4:

Figura 3.4 – Função ‘Localizar Cliente’

```
struct reg_cliente localizarCliente(int codcli){
    FILE *fpcliente;
    struct reg_cliente cliente;
    int achou=0;

    fpcliente = fopen(CLIENTES,"rb");

    while((achou==0)&&(fread(&cliente,sizeof(cliente),1,fpcliente)==1)){
        if (cliente.codigo==codcli){ //Achei o cliente
            achou=1;
        }
    }

    fclose(fpcliente);

    if (achou==0){ //NaO achou o cliente com este codigo
        cliente.codigo=-1;
    }

    return cliente;
} //Fim localizarCliente()

struct reg_livro localizarLivro(int codliv){
    FILE *fplivro;
    struct reg_livro livro;
    int achou=0;

    //Abrir o Arquivo
    fplivro = fopen(LIVROS,"rb");

    //Ler registro por registro e comparar o codigo
    //Se for igual ao codigo procurado, mostrar na tela
    while ((achou==0)&&(fread(&livro,sizeof(livro),1,fplivro) ==1)) {
        if (livro.codigo == codliv){
            achou = 1; //sinaliza que encontrou o livro
        }
    }

    //Fechar o arquivo
    fclose(fplivro);

    if (achou==0){ //Livro NAO localizado
        livro.codigo = -1;
    }

    return livro;
} //Fim localizarLivro()
```

Fonte. Elaborado pelo autor, 2023

Nas funções de listagem de livros (**listarLivros()**) e de clientes (**listarClientes()**), elas abrem os arquivos correspondentes e exibem na tela os registros de todos os livros e clientes, respectivamente.

A função **consultarCodigoLivro()** permite ao usuário buscar um livro específico pelo seu código. O usuário insere o código do livro desejado, e o





programa verifica se esse código existe no arquivo de livros. Caso exista, os dados do livro são exibidos na tela.

A função **relatorioDetalhado()** lê os dados de vendas registrados, obtém informações detalhadas dos livros e clientes relacionados a cada venda e calcula o total de vendas para cada pedido exibido. O relatório resultante mostra os detalhes das vendas, o número do pedido, o nome do cliente e o valor total da venda.

A função **listarEstoqueMinimo()** lista os livros que possuem um estoque igual ou inferior à quantidade mínima especificada pelo usuário. Ela gera um relatório com essas informações, exibe o relatório na tela e também o grava em um arquivo chamado "RelLivrosEst.txt".

Finalmente, as funções **localizarCliente()** e **localizarLivro()** são funções auxiliares que buscam um cliente ou um livro no arquivo correspondente pelo código fornecido. Essas funções são utilizadas em outras partes do código para verificar a existência de um cliente ou de um livro.

Em seguida, foram criadas funções de alteração e exclusão, **alterarLivro()**, **excluirLivro()**, **fecharPedido()** e **baixar estoque()**.

A função **alteraLivro()** possibilita a alteração dos dados de um livro cadastrado. O usuário informa o código do livro a ser alterado, e o programa verifica se esse código existe no arquivo de livros. Se existir, são exibidos os dados do livro e o usuário tem a opção de alterar o título, o preço e/ou o estoque do livro. Os novos dados são salvos no arquivo.

A função **excluirLivro()** permite a exclusão de um livro cadastrado. O usuário informa o código do livro a ser excluído, e o programa verifica se esse código existe no arquivo de livros. Se existir, é solicitada a confirmação do usuário antes de realizar a exclusão. Em seguida, é criado um novo arquivo, copiando todos os registros de livros, exceto o que deve ser excluído. O arquivo antigo é excluído e o novo arquivo é renomeado.

A função **fecharPedido()** mostra os detalhes de um pedido específico, incluindo as informações do cliente, os livros comprados, seus preços e quantidades, além do total a ser pago.

E a função **baixarEstoque()** é responsável por atualizar o estoque de um livro após uma venda. O usuário informa o código do livro e a quantidade vendida, e o programa busca o livro no arquivo de livros e atualiza o seu estoque.





Com todas essas funções, desde funções de listagem e verificação, até as funções de alteração e exclusão (também baixa) de estoque, verifica-se o componente do código de manipulação de arquivos para controle da aplicação. Todas essas funções estão contempladas na função de chamada final da aplicação, **Main()**, que tem o objetivo de coordenar a execução do programa e interagir com o usuário por meio da entrada e saída do console. Ela exibe um menu com opções para o usuário escolher e executa diferentes ações com base na opção selecionada. Ela desempenha um papel fundamental e é chamada automaticamente quando o programa é executado, como pode ser observado na Figura 3.5:



Figura 3.5 – Função *Main*

```
void main( void ){
    int op;

    do {
        printf("\n\n##### - Livraria Tec Info - #####");
        printf("\n# 1) Cadastrar Livro          #");
        printf("\n# 111)Cadastrar Cliente          #");
        printf("\n# 2) Listar todos os Livros          #");
        printf("\n# 222) Listar Clientes          #");
        printf("\n# 3) Consultar Livro pelo Codigo  #");
        printf("\n# 4) Consultar Livro pelo Titulo  #");
        printf("\n# 5) Consultar Livro por Palavra-Chave no Titulo  #");
        printf("\n# 6) Alterar dados de um Livro  #");
        printf("\n# 7) Exclusao Fisica de um Livro  #");
        printf("\n# 8) Efetuar Venda          #");
        printf("\n# 9) Relatorio de Vendas          #");
        printf("\n# 999) Fechar Pedido do Cliente  #");
        printf("\n# 10) Lista Estoque Minimo          #");
        printf("\n# 11) Relatorio Detalhado          #");
        printf("\n# 0) Sair          #");
        printf("\n\n Opcao-> ");
        fflush(stdin); scanf("%i",&op);

        switch(op){
            case 1: //Cadastro de Livro
                printf("\nCadastro de Livro\n");
                cadastrarLivro();
                break;
            case 111: //Cadastro de Cliente
                printf("\nCadastro de Cliente\n");
                cadastrarCliente();
                break;
            case 2: //Relatorio de todos os Livros
                listarLivros();
                break;
            case 222: //Relatorio de todos os Clientes
                listarClientes();
                break;
            case 3: //Consultar Livro pelo Codigo
                consultarCodigoLivro();
                break;
            case 4: //Alterar dados de um Livro
                alterarLivro();
                break;
            case 7: //Exclusao Fisica de um Livro
                excluirLivro();
                break;
            case 8: //Efetuar Venda
                efetuarVenda();
                break;
            case 9: //Relatorio de Vendas
                listarVendas();
                break;
            case 999: //Fechar Pedido do Cliente
                fecharPedido();
                break;
            case 11: //Relatorio Detalhado
                relatorioDetalhado();
                break;
            case 10: //Listar estoque minimo
                listarEstoqueMinimo();
                break;
        }
    }while (op!=0);

} //Fim main
```

Fonte. Elaborado pelo autor. 2023





3.3 CONCLUSÃO

Por meio desta atividade em C, pode-se criar uma aplicação com base em arquivos para armazenamento de entradas, com criação, manipulação e execução de arquivo para entrega da atividade semestral na disciplina de Estrutura de dados.

Esta atividade trouxe um grande passo no conhecimento de programação, estrutura de código e como orquestrar a entrada e saída de dados por meio de comandos e funções (ou métodos) dentro da capacidade interativa do código.

No geral, o código fornece funcionalidades básicas de gerenciamento de uma livraria, permitindo o cadastro, consulta e atualização de informações de livros e clientes.





4 WEBSITE USANDO *HTML* E *BOOTSTRAP*

No quarto semestre, na disciplina de Programação web, a primeira atividade do primeiro bimestre letivo foi o desenvolvimento de páginas web usando os conhecimentos de linguagem de programação, principalmente o *HyperText Markup Language (HTML)* e com elementos de estilização como o *Cascading Style Sheets (CSS)*, *Materialize* e *Bootstrap*.

Usando os conhecimentos em sala de aula com exercícios e demonstrações em vídeo, conceituou-se o que seria a linguagem de programação para web, como criar os componentes e como estilizá-los, usando as ferramentas mais atuais para desenvolvimento web.

Finalmente, foi possível, com a combinação de todos os elementos supracitados, provisionar quatro páginas em *.html*, com as estilizações correspondentes em *Bootstrap* para validação das ferramentas e conteúdo passado em sala de aula, em um dos temas sugeridos pelo professor.

4.1 CONCEITUAÇÃO

A forma de usar o computador mudou já há algumas décadas de forma abrupta e rápida, devido todas as mudanças tecnológicas pela qual passou o mundo e pelo avanço da Globalização, aproximando as pessoas e países devido a adição de formas de comunicação.

A rede de internet, ou *web*, é mais uma ferramenta tecnológica que surge como fator predominante na interação entre pessoas, empresas e até sistemas.

A princípio, faz-se necessário a conceituação do que é Desenvolvimento Web e como tem sido tão presente no dia a dia de quem usa a internet para o trabalho, lazer ou até mesmo estudos.

Podemos conceituar da seguinte forma:

O termo 'Desenvolvimento Web' pode ser definido como a construção, criação e manutenção de sites na Internet ou em uma intranet. Normalmente, o desenvolvimento na Web envolve um *front-end (client-side)*, código que interage com o cliente através de um navegador, e um *back-end (server-side)*, código executado no servidor Web, que encapsula a lógica de negócios da aplicação e a interação com o banco de dados (SANTIAGO, C. P. *et al*, 2020, p.5).

Deste modo, vê-se que o desenvolvimento de páginas para a internet tem como seu principal item o que é mostrado no front-end, tendo uma infraestrutura





por trás que dá sustentação às interações do que o usuário (ou cliente) vê e o que o site corresponde (servidor).

Sobre o *front-end*:

Tudo o que é visto e ‘navegado’ pelo usuário (identidade visual do site, menus, textos, imagens etc.) é criado e mantido pelo desenvolvedor *front-end*. Para isso, são escritos programas para vincular e estruturar os elementos visuais, adicionando interatividade com os usuários. Esses programas são executados através de um navegador (*browser*). No *front-end*, o desenvolvedor transforma a ideia de design da aplicação em realidade, concentrando-se no *layout*, design e interatividade da aplicação *Web* (SANTIAGO, C. P. *et al*, 2020, p.5).

Dentre as linguagens para se criar ambientes *front-end*, pode-se listar algumas como *HTML*, *PHP*, *Python*, *Ruby*, *Javascript* entre outras.

Dentre as mais utilizadas, há que se destacar neste trabalho o *HTML* por ser uma linguagem amplamente difundida no meio de desenvolvimento *web* e por ser o requerimento da atividade vista no quarto semestre, além de ser a linguagem abordada pelo professor durante a disciplina.

Sobre *HTML*, é a linguagem de marcação padrão usada para criar e estruturar páginas web. Ele fornece uma maneira de definir a estrutura e o conteúdo de uma página por meio de elementos e *tags* específicos. Essas *tags* são interpretadas pelos navegadores para exibir o conteúdo de maneira apropriada. O *HTML* é uma tecnologia fundamental para o desenvolvimento web e é amplamente utilizado em conjunto com outras tecnologias, como *CSS* e *JavaScript*, para criar páginas web interativas e visualmente atraentes.

Figura 4.1 – Exemplo de código em *HTML*

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

Fonte. <https://www.devmedia.com.br/html-basico-codigos-html/16596>

Já o *CSS* pode ser definido da seguinte forma:

CSS (Cascading Style Sheets): é uma “folha de estilos” que descreve como os elementos *HTML* aparecerão em uma página da *Web*. Usa-





se o CSS para controlar a apresentação, o estilo e a formatação dos elementos *HTML* em um site, configurando cores, bordas, imagens de fundo etc. Os arquivos CSS declaram um conjunto de regras, que definem propriedades e seus respectivos valores para os elementos *HTML* (SANTIAGO, C. P. et al, 2020, p.6).

Figura 4.2 - Exemplo de código em CSS

```
3 <title>Código CSS - Entendendo a folha de estilos</title>
4 <style type="text/css">
5 h1{
6     font-style: italic;
7 }
8 #fundo{
9     background-color: #a3a9fa;
10 }
11 .paragrafo{
12     color: red;
13 }
14 .paragrafo2{
15     color: #0018ff;
16 }
17 </style>
18 </head>
19 <body>
20 <div id="fundo">
```

Fonte. <https://www.devmedia.com.br/codigo-css-entendendo-a-folha-de-estilos/37459>.

Associados, o *HTML* e o *CSS* tornam-se ferramentas poderosas no Desenvolvimento *Web*.

Além do uso de linguagens de programação, o uso de *frameworks* também é uma prática bastante conhecida no desenvolvimento *Web*, pois podem ser usados tanto no *front-end* quanto no *back-end*, fazendo parte da gama de ferramentas nesse contexto. É uma combinação de software, aplicativos, linguagens de programação e ferramentas que são construídas umas sobre as outras para criar um site (Santiago, C. P. et al, 2020, p.5).

E, neste caso, para essa atividade específica, usou-se o *Bootstrap* para facilitar na criação de *layouts*, e também por organizar os componentes da página, reutilizar esses componentes, ter estilos e temas pré-definidos para utilização, além de ter uma documentação clara e concisa:

Desde que o *Bootstrap* foi lançado em agosto de 2011, ele decolou em popularidade. Ele evoluiu de um projeto totalmente baseado em *CSS* para incluir uma série de plug-ins e ícones *Javascript* para acompanhar formulários e botões (SPURLOCK, J., 2013, p. 1).

Assim, agregando outros componentes como *Javascript* e *CSS*, um *framework* como o *Bootstrap* torna-se um aliado no desenvolvimento de





aplicações *Web*, de modo fácil e prático, além de ter uma estilização bastante agradável para ambientes *front-end*, sendo o *framework* escolhido para complementar a atividade da disciplina em questão.

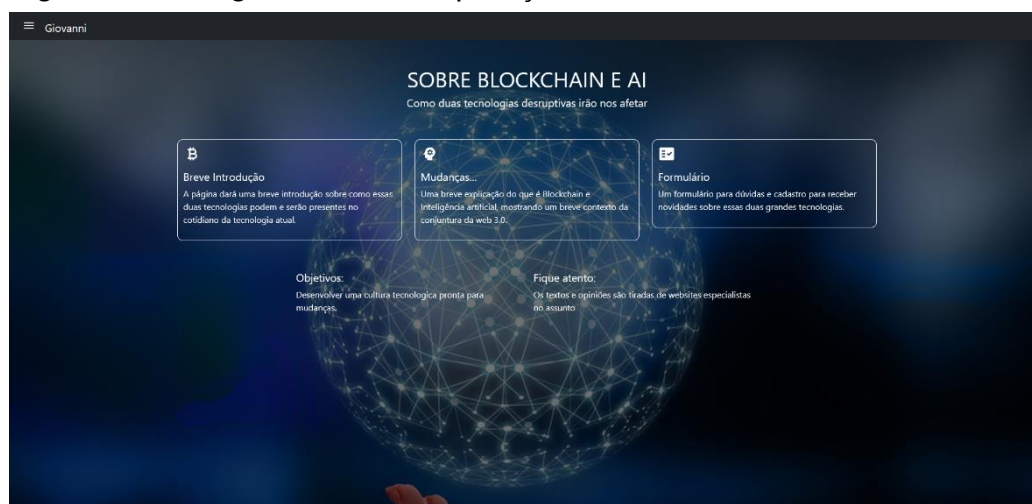
4.2 DESENVOLVIMENTO

Na produção das páginas web utilizando *HTML* e utilizando um dos *frameworks* sugeridos pelo professor, neste caso, o *Bootstrap*, dá-se início à construção da página *Home*, geralmente denominada *index.html*. Esta denominação para a página principal da aplicação é uma prática costumeira no desenvolvimento de aplicação para internet.

Dentro da gama de assuntos escolhidos para o desenvolvimento do conteúdo da página, foi escolhido o tema de *Blockchain* e Inteligência Artificial, apenas como modo de destacar o conteúdo por ter muitos elementos visuais interessantes e ser um assunto em alta no mundo da tecnologia.

Foram usados alguns componentes do *Bootstrap* para a primeira página como um menu retrátil lateral, abrindo para mostrar a indexação das outras páginas e uma barra no *header* da página, muito utilizados em páginas *Web*.

Figura 4.3 – Página *Home* da aplicação web



Fonte. Elaborado pelo autor. 2023

Para a segunda página, foi requisitado pelo professor que fosse usado ao menos uma imagem no centro da página e um *link* abrindo para uma página externa. Neste elemento, foram usadas linhas de códigos nativas de *HTML* para incorporar a imagem à página por meio de uma *Uniform Resource Locator* (URL)





para direcionar a página externa; e um *link* no rodapé da página como uma estilização também do *Bootstrap*, de modo a ficar mais visível na página.

Figura 4.4 – Exemplo da página 2 com imagem incorporada

A primeira aparição do termo blockchain foi em 2008, no artigo acadêmico Bitcoin: um sistema financeiro eletrônico peer-to-peer. O artigo, no entanto, foi publicado por Satoshi Nakamoto, pseudônimo do suposto criador da bitcoin. Desde o início, o bitcoin foi descrito por Satoshi Nakamoto, como "um sistema para transações eletrônicas sem depender da confiança de terceiros". Em plena crise econômica, o bitcoin foi criado para prevenir o gasto duplo dos valores. Além disso, a criptomoeda foi gerada visando aumentar a confiança das transações financeiras, levando-as para a internet. Porém, no ambiente digital, os dados podem ser copiados, alterados e trocados. Para isso, foi necessário implementar a tecnologia da blockchain, conjunto de regras que fazem a moeda funcionar.



Fonte. Elaborado pelo Autor, 2023

Figura 4.5 – Exemplo da página 2 com *link* externo no rodapé

Vantagens da rede de blocos criptograficos:

1) Segurança:

Os blocos do blockchain são criptografados em segurança. Para hackear uma operação, seria necessário alterar todas as transações naquele bloco, não apenas em um computador mas em milhões de computadores simultaneamente.

2) Rapidez:

Ao contrário do ambiente físico, o mundo digital dos algoritmos e da blockchain funciona 24 horas por dia e sete dias por semana. Assim, o tempo das transações bancárias podem ser reduzidas a minutos.

3) Menos custo de Transferência:

Não há intermediação entre as transações. Além de reduzir consideravelmente os riscos e as vulnerabilidades da transação, a falta de intermediários diminui o custo das operações.

4) Transparência:

Qualquer mudança na blockchain se torna pública para todas as partes e usuários, o que garante uma transparência inquestionável. Essas alterações também não podem ser modificadas, o que garante ainda mais confiabilidade ao sistema.

5) Privacidade:

Ao mesmo tempo em que informações essenciais são mantidas públicas, detalhes mais sensíveis são mantidos em sigilo absoluto.

[Mais sobre blockchain](#)

Fonte. Elaborado pelo Autor, 2023

Já para a terceira página, foram usados os mesmo elementos e técnicas da segunda página, tendo não tendo diferenciais relevantes em comparação às outras páginas, com exceção de duas imagens incorporadas ao invés de uma. As mesmas linhas de código usadas para incorporar imagens foram atribuídas ao código para dar o efeito esperado, inclusive em uma delas que compõe todo





o *grid* (representação gráfica que serve como um suporte para os elementos que comporão um layout). Um ponto relevante do uso do *Bootstrap* e outros *frameworks* é a possibilidade de criação de elementos que deixam a página “responsiva”, ou melhor, flexível de acordo com o dispositivo usado ou o tamanho da tela.

A responsividade proporcionada pelo *Bootstrap* permite que os usuários possam acessar os sites em todos os tipos de dispositivos de forma otimizada e sem que informações sejam perdidas durante a navegação. Por conta disso, os desenvolvedores não precisam se preocupar em fazer várias versões de um site para todos os tipos e tamanhos de telas disponíveis. O sistema de *grid* (ou grade, em português) do *Bootstrap*, faz uso de vários tipos de containers, linhas e colunas para organizar e alinhar visualmente o conteúdo (*Bootstrap* [2020b]). Sistema *grid*. *apud* SANTIAGO, C. P. *et al*, 2020, p.6).

Figura 4.6 – Exemplo da página 3 com imagem incorporada na totalidade do *grid*



Fonte. Elaborado pelo Autor, 2023

Por fim, a última página exibe um formulário de preenchimento do suposto usuário do site. O formulário é um componente que geralmente pode ser usado pronto, direto do *framework Bootstrap*. Vários *frameworks* proporcionam este tipo de usabilidade, em que o desenvolvedor pode fazer o *download* do elemento na sua integridade e utilizar dentro do código para ser visualizado no site.

Os formulários são elementos usados desde muito tempo no desenvolvimento *Web*. O *Bootstrap* estiliza elementos de entradas como





<input>, <textarea> e <select> melhorando a visualização dos formulários (Santiago, C. P. *et al*, 2020, p.16).

Neste caso, o formulário não possui nenhum tipo de persistência de dados, sendo feito somente a propósito de exemplificar o uso de um formulário de um *framework*.

Figura 4.7 – Página 4 com exemplo de formulário

Quer saber mais?

O espaço abaixo está destinado para suas possíveis dúvidas e para deixar seu cadastro caso queira se incluir no mailing para novidades e curiosidades no mundo de Blockchain e IA

Deixe suas dúvidas!

I

Cadastro

Nome Sobrenome

E-mail

voce@email.com.br

Celular

+55 11 99999 8888

Endereço

Rua Vinte e dois de Março

Complemento

Apartamento / Bloco

Cidade

Fonte. Elaborado pelo Autor, 2023

4.3 CONCLUSÃO

Deste modo, esta atividade do quarto semestre proporcionou uma bagagem considerável de ferramentas e elementos *front-end* para possíveis próximas oportunidades desenvolvimento *Web*.

Obteve-se conhecimento da principal linguagem de programação, o *HTML*, juntamente com ferramentas de estilização importantes como *CSS* e um dos *frameworks* mais utilizados, o *Bootstrap*.

HTML e *Bootstrap* se tornaram comuns no desenvolvimento web devido a diversos fatores, como sua facilidade de uso, documentação abrangente, suporte da comunidade, recursos avançados e popularidade em grandes empresas de tecnologia. O *HTML* tem sido usado desde os primeiros dias da web e é uma linguagem fundamental para a criação de páginas web. Já o *Bootstrap* ganhou popularidade por sua capacidade de agilizar o processo de desenvolvimento, permitindo que os desenvolvedores criem páginas responsivas e atraentes com pouco esforço.





5 AGENDA TELEFÔNICA PARA MOBILE USANDO JQUERY

Na primeira entrega do 5º semestre, como atividade da primeira prova do bimestre, foi desenvolvido uma página *index.html* para mobile com uma agenda telefônica com algumas funcionalidades básicas de entrada de dados, como “adicionar”, “atualizar” e “procurar” contatos na agenda. Todas as *features* do trabalho foram desenvolvidas em HTML juntamente com a biblioteca em *JavaScript JQUERY*.

Quando se trata de desenvolvimento *web* ou desenvolvimento de aplicações *mobile* (leia-se para dispositivos móveis ou aparelhos celulares), o uso de bibliotecas associados a uma linguagem de desenvolvimento *web* são muito comuns pois acrescentam funcionalidades extras à aplicação e aos componentes para interação entre a plataforma/aplicação e o usuário.

Um bom exemplo é exatamente o uso da biblioteca *JQUERY* com *HTML*, que possui funcionalidades que são facilmente usadas em programação *web* para acrescentar *plugins*, efeitos de animação, controles deslizantes, entre outros, que dão uma dinâmica e interatividade extra quando se trata de aplicações e páginas *web*.

Neste trabalho, focou-se em trazer os principais elementos de *JQUERY*, como seletores e métodos de manipulação para construção de uma agenda telefônica simples com o objetivo de apresentar essas funcionalidades da biblioteca em *JavaScript*.

5.1 CONCEITUAÇÃO

Primeiramente, o conteúdo da disciplina de Programação para Dispositivos Móveis, no 5º semestre, traz a importância da programação para interação com o usuário no dispositivo mais usado hoje na atualidade.

Diversos aplicativos atualmente exigem novas formas de interface e interação com o usuário devido ao grande volume de aplicativos móveis e seus mais variados objetivos. Assim, faz-se uso de novos e modernos instrumentos para transformar o uso de aplicações *mobile* e *web* divertidas, engajadoras e que façam com que os usuários continuem a buscar cada vez mais utilidade nestas aplicações.





Neste quesito, o *JQUERY* aparece como uma biblioteca fácil de usar e que, no caso deste trabalho para o 5º semestre dessa disciplina, tornou-se útil para aplicar componentes básicos para a agenda que foi proposta.

Inicialmente, destaca-se o *JQUERY* como uma biblioteca *Javascript* apresentada pela primeira vez em 2006 por John Resig para desenvolver aplicações que seriam suportadas pela maioria dos navegadores e plataformas de *smartphones*. O *framework* aplica o princípio “fazer mais escrevendo menos” com a finalidade de criar aplicações “únicas” para todos os dispositivos móveis e sistemas operacionais (SILVA, 2012, p. 21).

Para fazer uso do *JQUERY* nesta aplicação móvel criada para a atividade dessa disciplina, foi necessário acessar a biblioteca por meio do *Content Distribution Network* (CDN), que vincula remotamente o conteúdo da biblioteca sem que esta esteja hospedada no próprio servidor do desenvolvedor. Na definição de SILVA, temos:

CDN é um sistema de computadores, contendo cópias de um arquivo de dados, colocados em diferentes nós de uma rede. Um CDN bem planejado e implementado tem inúmeras vantagens sobre o armazenamento em um servidor próprio; dentre elas, destacamos as seguintes: melhoria sensível do acesso aos dados em cache e consequentemente redução do consumo de banda; incremento do paralelismo, ou seja, da capacidade de oferecer acesso simultâneo; redução da latência servindo os dados de um servidor mais próximo do usuário. (SILVA.2012, p. 26)

Deste modo, a utilização da biblioteca fica fácil de usar e sem nenhum impedimento de configuração de arquivos em pastas locais, no servidor do usuário. Basta a utilização de uma URL para acessar o conteúdo da biblioteca remotamente como o exemplo abaixo:

```
<script src="https://code.jquery.com/jquery-3.6.0.js" integrity="sha256-H+K7U5CnXI1h5ywQfKtSj8PCmoN9aaq30gDh27Xc0jk=" crossorigin="anonymous"></script>
```

Outro ponto de importância ao iniciar a utilização do *JQUERY* é ter uma estrutura mínima de código para que a biblioteca possa ser aplicada e seus métodos e componentes possam manipular o *Document Object Model* (DOM).

O DOM nada mais é do que a representação da estrutura do documento que fornece uma interface que os programas possam usar para acessar e modificar dinamicamente os elementos e atributos da página. E é exatamente a





proposta do *JQUERY* neste caso: a manipulação do DOM por meio de acesso a elementos, modificação de conteúdo, manipulação de estilos, gestão de eventos, entre outros.

Deste modo, é feito o desenvolvimento da aplicação para apresentação de uma interação fluida e ter opções de IU/UX cabíveis ao objetivo do projeto. Assim, justifica-se o uso de bibliotecas para ambientes mobile conforme foi explanado acima.

5.2 DESENVOLVIMENTO

A partir do estudo e conhecimento mais aprofundado em *JQUERY* e uso dentro de *HTML*, em adição com as aulas ministradas pelo professor da disciplina, foi proposto que atividade contivesse 6 funcionalidades, todas feitas por meio de *JQUERY*, que são descritas abaixo:

- Funcionalidade 1 - Adicionar um registro: Após o usuário inserir os dados do formulário (nome e telefone) e clicar em [ADICIONAR], os dados devem ser listados abaixo, conforme o layout apresentado na descrição da atividade.
- Funcionalidade 2 - Atualizar um registro parte 1: ao clicar no ícone [EDITAR], os dados atuais devem ser previamente carregados nos campos que se encontram na aba atualizar. Para carregar a aba atualizar, utilize o seguinte código `$('.tabs').tabs('select','up');`
- Funcionalidade 3 - Atualizar um registro parte 2: ao clicar em [ATUALIZAR], deve-se atualizar os dados (nome e telefone) do registro em questão, levando-se em consideração o id escolhido.
- Funcionalidade 4 - Excluir um registro: ao clicar no ícone [EXCLUIR], deve-se excluir o registro em questão.
- Funcionalidade 5 - Procurar um registro: após digitar o nome a ser procurado e clicar em [PROCURAR], deve-se listar apenas o registro o qual o nome é exatamente igual ao digitado no campo.
- Funcionalidade 6 - Mostrar um registro: ao clicar em [MOSTRAR TODOS], deve-se listar todos os registros cadastrados pelo usuário e limpar o campo procurar.





Para cada funcionalidade, foi usado um método com seletores para aplicar o efeito desejado e manipular a página de acordo com o que foi pedido em cada item.

Na primeira funcionalidade, para adicionar uma listagem, foi construído o seguinte bloco de código, no arquivo *index.js*:

Figura 5.1 – Função #btn-add [ADICIONAR]

```
$("#btn-add").on("click", function () {
    var newRowId = $("#lista tr").length;
    var id = $("#addID").val() || newRowId;
    var nome = $("#addNome").val();
    var telefone = $("#addTelefone").val();

    if (nome.trim() === "") {
        alert("O campo de nome não pode ficar em branco.");
        $("#addNome").css("border", "2px solid red")
            .animate({ marginLeft: "-=10px" }, 50)
            .animate({ marginLeft: "+=20px" }, 50)
            .animate({ marginLeft: "-=20px" }, 50)
            .animate({ marginLeft: "+=20px" }, 50)
            .animate({ marginLeft: "-=10px" }, 50);
        return;
    } else {
        $("#addNome").css("border", "");
    }

    if (!checarTelefone(telefone)) {
        alert("O número de telefone precisa ter mais que 11 dígitos (DDD + Número).");
        $("#addTelefone").css("border", "2px solid red")
            .animate({ marginLeft: "-=10px" }, 50)
            .animate({ marginLeft: "+=20px" }, 50)
            .animate({ marginLeft: "-=20px" }, 50)
            .animate({ marginLeft: "+=20px" }, 50)
            .animate({ marginLeft: "-=10px" }, 50);
        return;
    } else {
        $(this).css("border", "");
    }

    $("#addTelefone").css("border", "");

    var novalinha = "<tr>" +
        "<td>" + id + "</td>" +
        "<td>" + nome + "</td>" +
        "<td>" + telefone + "</td>" +
        "<td><a href='#><i class='material-icons'>edit</i></a></td>" +
        "<td><a href='#><i class='material-icons'>delete_forever</i></a></td>" +
        "</tr>";

    $("#lista").append(novalinha);
    $("#addNome").val("");
    $("#addTelefone").val("");
});
```

Fonte. Elaborado pelo Autor, 2023

Este bloco de código *JQUERY* é um manipulador de evento que responde ao clique no elemento com o ID **#btn-add**. Ele executa uma série de ações quando o botão é clicado, principalmente adicionando uma nova linha à tabela com base nos valores inseridos nos campos de entrada (**#addID**, **#addNome** e **#addTelefone**).





Em resumo, este bloco de código manipula a adição de uma nova linha à tabela, exibindo alertas e aplicando estilos visuais se a entrada não estiver de acordo com as condições especificadas.

Para a segunda funcionalidade [EDITAR], temos um ícone de um “lápis” destacado ao lado de cada linha da agenda, e ao clicar no ícone, chama o código do elemento **#btn-up** e, conseqüentemente, abre a aba “atualizar” da agenda onde a função é referenciada.

Figura 5.2 – Função [EDITAR]

```
$("#lista").on("click", "td > a > i.material-icons:contains('edit')", function (event) {
    event.preventDefault();

    var selecionarDados = {
        id: null,
        nome: null,
        telefone: null
    };

    var row = $(this).closest("tr");

    selecionarDados.id = row.find("td:eq(0)").text();
    selecionarDados.nome = row.find("td:eq(1)").text();
    selecionarDados.telefone = row.find("td:eq(2)").text();

    $('#upId').val(selecionarDados.id);
    $('#upNome').val(selecionarDados.nome);
    $('#upTelefone').val(selecionarDados.telefone);

    $('.tabs').tabs('select', 'up');
});
```

Fonte. Elaborado pelo Autor, 2023

Esse bloco de código jQuery é um manipulador de evento que responde ao clique em um objeto (neste caso, o ícone do “lápis”) dentro da agenda com o ID **#lista**. Ele é especificamente acionado quando um ícone com a classe *material-icons* contendo o texto *'edit'* é clicado. Este bloco de código captura os dados da linha da tabela associada ao ícone *'edit'* clicado e os exibe nos campos correspondentes na aba “Atualizar”.

Já a terceira funcionalidade, [ATUALIZAR], parece-se muito com a segunda, com a única diferença que o acesso é feito pela aba “Atualizar” da agenda ao invés do ícone “lápis” na frente da linha de registro.





Em relação ao código, a função usada para chamar o método com os seletores é o mesmo, sendo a base da funcionalidade [EDITAR], feito diretamente em um dos registros da agenda.

Figura 5.3 – Função [ATUALIZAR] da Agenda

```
$("#btn-up").on("click", function (event) {  
    event.preventDefault();  
  
    var updatedId = $("#upId").val();  
    var updatedNome = $("#upNome").val();  
    var updatedTelefone = $("#upTelefone").val();  
  
    if (updatedNome.trim() === "") {  
        alert("O campo de nome não pode ficar em branco.");  
        $("#upNome").css("border", "2px solid red")  
            .animate({ marginLeft: "-=10px" }, 50)  
            .animate({ marginLeft: "+=20px" }, 50)  
            .animate({ marginLeft: "-=20px" }, 50)  
            .animate({ marginLeft: "+=20px" }, 50)  
            .animate({ marginLeft: "-=10px" }, 50);  
        return;  
    } else {  
        $(this).css("border", "");  
    }  
  
    if (!checarTelefone(updatedTelefone)) {  
        alert("O número de telefone precisa ter mais que 11 dígitos (DDD + Número)");  
        $("#upTelefone").css("border", "2px solid red")  
            .animate({ marginLeft: "-=10px" }, 50)  
            .animate({ marginLeft: "+=20px" }, 50)  
            .animate({ marginLeft: "-=20px" }, 50)  
            .animate({ marginLeft: "+=20px" }, 50)  
            .animate({ marginLeft: "-=10px" }, 50);  
        return;  
    } else {  
        $(this).css("border", "");  
    }  
  
    $("#lista tr").each(function () {  
        var rowId = $(this).find("td:eq(0)").text();  
        if (rowId === updatedId) {  
            $(this).find("td:eq(1)").text(updatedNome);  
            $(this).find("td:eq(2)").text(updatedTelefone);  
            return false;  
        }  
    });  
});
```

Fonte. Elaborado pelo Autor, 2023

Esse bloco de código jQuery é um manipulador de evento que é acionado quando o botão com o ID **#btn-up** é clicado (na aba “Atualizar” da agenda).

Este código é utilizado para atualizar os dados de uma linha de registro na agenda com base nos valores fornecidos nos campos de edição. Ele inclui verificações de validação (condicionais) para garantir que o nome não esteja em branco e que o número de telefone atenda a determinados critérios.





Por exemplo, a verificação *if (!checarTelefone(updatedTelefone))*, verifica se a função *checarTelefone* retorna falso para o número de telefone, traz um alerta que um número com pelo menos 11 dígitos seja inserido para que seja válido.

Em seguida, na quarta funcionalidade [EXCLUIR], utiliza-se a mesma interface da funcionalidade [EDITAR], em que um ícone (“lixeira”) é mostrado ao lado de um registro da agenda, e um *prompt* (uma janela na página) é acionado perguntando ao usuário se gostaria de remover o item (registro) da agenda.

Figura 5.4 – Função [EXCLUIR] da Agenda

```
$("#lista").on("click", "td > a > i.material-icons:contains('delete_forever')", function (event) {  
    event.preventDefault();  
  
    var row = $(this).closest("tr");  
  
    if (confirm("Gostaria de remover o registro?")) {  
        row.remove();  
    }  
});
```

Fonte. Elaborado pelo Autor, 2023

Esse bloco de código é mais um manipulador de evento associado a um elemento específico dentro da agenda. Ele cria um mecanismo para excluir registros de uma tabela quando o usuário clica no ícone "lixeira" ao lado do registro da agenda. Antes de excluir, ele exibe uma janela de *prompt* de confirmação para garantir que o usuário realmente deseja remover o registro.

Finalmente, a quarta funcionalidade [PROCURAR] também representa uma das abas da agenda, em que o usuário clica e essa chama o elemento **#btn-find** do código *index.js* para que faça consultas de nomes que estão na lista da agenda. Contudo, além de possuir a função de "Procurar", também chama o elemento **#btn-show-all** para que mostre toda a listagem dos registros da agenda, e procure dentre eles o valor requerido.

Deste modo, a aba "Procurar" compreende tanto a funcionalidade [PROCURAR] quanto a funcionalidade [MOSTRAR], esta última sendo a sexta funcionalidade da atividade proposta.





Figura 5.5 – Funções [ATUALIZAR] e [MOSTRAR] da Agenda

```
$("#btn-find").on("click", function(event) {
    event.preventDefault();

    var nomeInput = $("#findNome").val().toLowerCase();

    $("#lista tr").each(function () {
        var rowNome = $(this).find("td:eq(1)").text().toLowerCase();
        var normalizedNome = rowNome.normalize("NFD").replace(/[\u0300-\u036f]/g, "");

        if (normalizedNome.includes(nomeInput)) {
            $(this).show();
        } else {
            $(this).hide();
        }
    });
});

$("#btn-show-all").on("click", function(event) {
    event.preventDefault();

    $("#lista tr").each(function() {
        $(this).show();
    });
});
```

Fonte. Elaborado pelo Autor, 2023

Essencialmente, o bloco **#btn-find** é responsável por ocultar as linhas que não correspondem ao critério de pesquisa, enquanto o bloco **#btn-show-all** exibe todas as linhas novamente, restaurando a visibilidade completa da tabela. A pesquisa é sensível a maiúsculas e minúsculas, além de tratar acentos e caracteres especiais.

5.3 CONCLUSÃO

Assim sendo, esta atividade entregue como P1 do 5º semestre proporcionou mais conhecimento em programação, desta vez para dispositivos móveis, utilizando a biblioteca *JQUERY* como ferramenta, uma poderosa porém simples biblioteca *JavaScript* para acrescentar elementos importantes de UI e UX em uma aplicação. Este é o resultado da interface da agenda:





Figura 5.6 – Exibição da aplicação de Agenda Telefônica

Código	Nome	Telefone
0	Felício Felizardo	(14) 99799.1234

Fonte. Elaborado pelo Autor, 2023

Durante a implementação da agenda telefônica, foi necessário empregar seletores específicos em *JQUERY* para identificar e modificar elementos no DOM, proporcionando uma experiência de usuário mais fluida e responsiva. Essa abordagem não apenas demonstrou a aplicação efetiva de conhecimentos teóricos, mas também enfatizou a importância de adotar tecnologias modernas para a criação de interfaces intuitivas e amigáveis. A agenda telefônica desenvolvida serve como um exemplo prático da aplicação desses conceitos em um contexto real, consolidando os aprendizados teóricos em programação para dispositivos móveis.

Além disso, o desenvolvimento dessa agenda telefônica trouxe contribuições significativas para o aprimoramento do conhecimento em programação e design de interfaces, especialmente no contexto de dispositivos móveis. A experiência proporcionou insights valiosos sobre as melhores práticas de desenvolvimento, otimização de código e a importância de considerar as características específicas desses dispositivos ao criar soluções interativas. A conclusão deste capítulo destaca não apenas a realização bem-sucedida do projeto, mas também os benefícios educacionais e práticos alcançados no processo de construção desta aplicação voltada para dispositivos móveis.





6 CRIAÇÃO DE BANCO DE DADOS AWS DOCUMENTDB

Como último trabalho deste portfólio, para a disciplina de Laboratório de Banco de Dados, foi proposta uma atividade em formato de relatório e apresentação em que consistia na criação e *setup* de um banco de dados não-relacional, ou *Not Only SQL (NoSQL)*.

Na disciplina foi abordado o *setup* de um banco de dados relacional na plataforma *SQL Developer* da Oracle e o estudo da linguagem *Structured Query Language (SQL)* para comandos *CRUD* (*create, read, update, delete* – leia-se criar, ler, atualizar, excluir) muito comuns em bancos relacionais. Como proposta de conhecimento de outros tipos de bancos de dados, foi proposto o estudo individual de um banco de dados não-relacional para apresentação em sala.

Para este propósito, foi escolhido o banco de dados *DocumentDB* da *Amazon Web Services (AWS)*, uma aplicação em nuvem que pode ser configurada internamente no ambiente *AWS* com compatibilidade de interface *MongoDB*, uma das principais interfaces e banco de dados não-relacionais do mercado.

Com isso, fez-se o estudo de como funcionam os bancos de dados não-relacionais, os principais atores do mercado e como fazer o *setup* de um banco de dados não-relacional, neste caso o *DocumentDB* dentro da própria *AWS*, e a geração de comandos *CRUD* para apresentar as principais diferenças dos comandos em banco de dados *SQL* em comparação com os *NoSQL*.

6.1 CONCEITUAÇÃO

No universo de banco de dados, faz-se necessário enfatizar para que servem, quais os tipos e quais os reflexos que os banco de dados proporcionam no mundo dos dados.

Bancos de dados são sistemas organizados que armazenam e gerenciam dados, permitindo a criação, recuperação, atualização e gestão eficiente das informações. Uma definição mais precisa sobre banco de dados:

Um sistema de banco de dados é basicamente apenas um sistema computadorizado de manutenção de registros. O banco de dados, por si só, pode ser considerado como o equivalente eletrônico de um armário de arquivamento; ou seja, ele é um repositório ou recipiente para uma coleção de arquivos de dados computadorizados (DATE, C.J. 2004, p.3)





Com essa definição, podemos enxergar o banco de dados como uma forma de armazenamento de dados dentro de um servidor, ou simplesmente um computador.

Os bancos de dados são utilizados por meio de sistemas de banco de dados. Conforme Date (2004, p.6), um sistema de banco de dados é basicamente um sistema computadorizado de manutenção de registros (...). É um repositório ou recipiente para uma coleção de arquivos de dados computadorizados.

Um banco de dados pode ser modelado de formas diferentes, mas aqui abordar-se-á dois tipos principais e mais comumente usados: o relacional e o não-relacional, este último fruto do trabalho desenvolvido.

De acordo com Silberschatz (2006, p.5), o modelo relacional usa uma coleção de tabelas para representar os dados e as relações entre eles. Cada tabela possui diversas colunas e cada coluna possui seu nome único. Estes dados podem ser consultados por meio de uma linguagem conhecida como SQL. Por meio dessa linguagem, é possível a leitura em bancos de dados, além de criar, alterar e excluir dados (sigla *CRUD*).

Já o modelo não-relacional não é representado por meio de linhas em tabelas, e adota abordagens diversas como documentos, chave-valor, grafos e colunas, como no caso do *DocumentDB*, um banco de dados de documento no modelo semiestruturado.

Os modelos de dados semiestruturados permitem a especificação dos dados, em que itens de dados individuais do mesmo tipo podem ser diferentes conjuntos de atributos. Isso está em contraste com os modelos de dados mencionados anteriormente (relacionais), em que cada item de dados de um tipo específico precisa ter o mesmo conjunto de atributos (SILBERSCHATZ, 2006, p. 13).

No caso de banco de dados de documentos, os dados são frequentemente armazenados em formato de documento, que pode ser semelhante a *JavaScript Object Notation (JSON)* ou *Binary JSON (BSON)*. Esses formatos permitem que os dados sejam representados de maneira hierárquica e aninhada, oferecendo flexibilidade na adição ou remoção de campos em diferentes documentos.





Os bancos de dados de documentos normalmente possuem uma *Application Programming Interface* (API) ou linguagem de consulta que permite aos desenvolvedores executarem as operações *CRUD*.

No mundo dos dados não-relacionais, um dos principais bancos que usam dados semiestruturados de documentos é o *MongoDB*.

O *MongoDB* é um banco de dados *NoSQL* de código aberto, orientado a documentos e desenvolvido pela empresa MongoDB Inc. Ele é projetado para lidar com dados semiestruturados ou não-relacionais, proporcionando flexibilidade e escalabilidade para diferentes tipos de aplicações. Desenvolvido por Dwight Merriman, Eliot Horowitz e Kevin Ryan e foi lançado pela primeira vez em 2009. A ideia surgiu da necessidade de lidar com grandes volumes de dados não estruturados em um ambiente *web*.

Os bancos de dados de documentos oferecem uma variedade de vantagens, incluindo um modelo de dados intuitivo, rápido e fácil para os desenvolvedores trabalharem; um esquema flexível que permite que o modelo de dados evolua conforme as necessidades do aplicativo muda; e a capacidade de expansão horizontal (MONGODB. Acesso em 2023).

Contudo, a interface do *MongoDB* também pode ser trabalhada na nuvem pública AWS, tal qual foi escolhido para o presente projeto. Enquanto o MongoDB pode ser executado em qualquer ambiente, a utilização na AWS oferece vantagens específicas relacionadas à facilidade de implementação, escalabilidade, integração com outros serviços e gerenciamento de segurança e acesso.

Por buscar uma abordagem diferente do padrão *MongoDB* e pela demonstração dos benefícios de banco de dados em ambiente de nuvem, o *AWS DocumentDB* foi escolhido para o projeto neste caso.

O DocumentDB é um banco de dados não-relacional dentro da AWS com compatibilidade para MongoDB. Como todo micro serviço em nuvem, esse banco de dados herda características importantes de serviços em nuvem, pois é escalável, confiável, rápido, e completamente gerenciado pela AWS.

O serviço roda em *clusters* de instâncias ou *clusters* elásticos (*elastic clusters*) dentro da nuvem – este último usado para casos de uso com milhões de usuários ao mesmo tempo, com inúmeras leituras e escritas concomitantes.





Usado em uma rede privada da nuvem (*Virtual Private Cloud* - VPC), isola o banco de dados no próprio ambiente de nuvem e isso o torna mais seguro e com uma latência mais baixa. Outro benefício é o fato de poder ser escalável, conforme o uso aumenta e mais espaço é exigido para armazenamento.

O *DocumentDB* pode ser replicado para outras Zonas de Disponibilidade dentro de uma região, garantindo a confiabilidade do serviço e reduzindo a possibilidade de indisponibilidade do recurso.

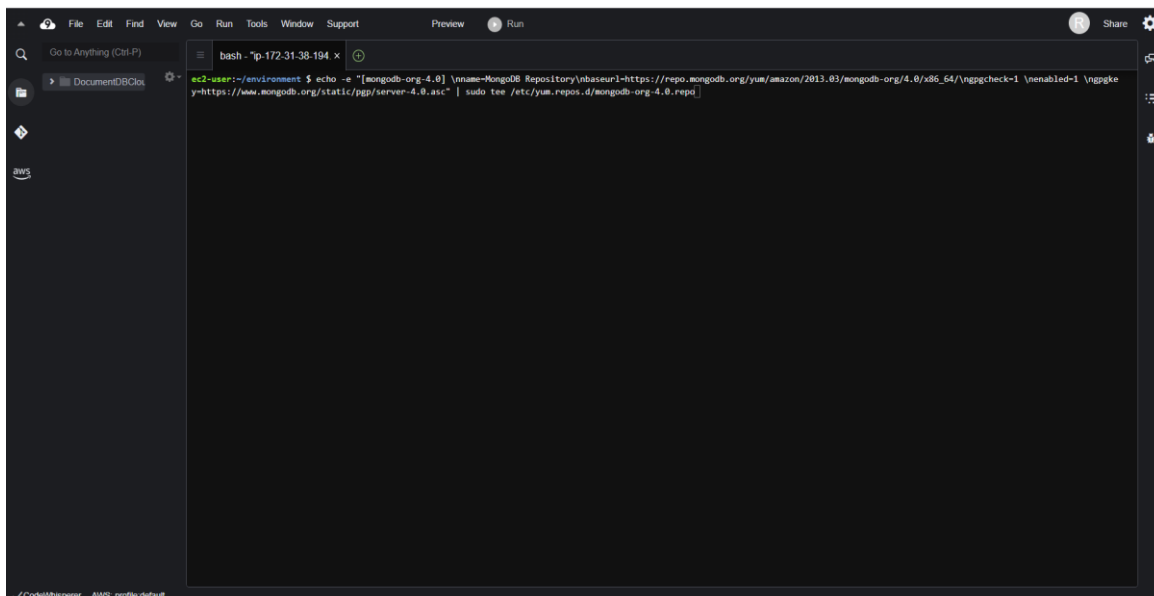
6.2 DESENVOLVIMENTO

No desenvolvimento do projeto de criação do *DocumentDB*, destaca-se o uso da nuvem da *Amazon*, *AWS*, assim como o preparo e *setup* de conta previamente para que a criação da “instância” (provisionamento do serviço na nuvem) do banco de dados fosse feita.

Por ser um serviço dentro da nuvem *AWS*, algumas etapas de configuração de serviços antes de usar o Banco de Dados propriamente. Os passos são:

- Criar uma conta grátis na *AWS*. A *AWS* possui um mandate de você paga pelo que usar. Neste caso, como é necessário instanciar uma máquina virtual, uma rede virtual e um *cluster* para o banco de Dados, custos serão gerados. Porém, existe uma faixa *free tier* para a utilização do *DocumentDB* em que é possível usá-lo por 30 dias grátis, por até 750 horas.
- Instanciar um ambiente para o *Cloud9* – um *Integrated development environment* (IDE) da *AWS* - e uma *Elastic Compute Cloud* (EC2) - máquina virtual da *AWS* - para se trabalhar o código e os comandos *CRUD* no prompt do banco de dados. A EC2 é criada automaticamente ao se criar um ambiente para o *Cloud9*.



Figura 6.1 – Interface do *IDE Cloud9* na *AWS*

Fonte. Elaborado pelo Autor, 2023

- Criar um Grupo de Segurança (*Security Group*). O *Security Group* serve para conectar o Cloud9 ao banco de dados *DocumentDB*. Isso é feito no console da *AWS*, nas configurações da instancia *EC2* criada.
- Criar um cluster *DocumentDB*. É onde o banco de dados ficará armazenado.
- Instalar o shell do *MongoDB*. Como a interface é a mesma, é necessário instalar essa interface virtual dentro do Cloud9.
- Configurar a conexão para *DocumentDB* (necessita baixar o certificado e conectar usando o *mongo shell* e o *cluster* do *DocumentDB*).

Após os passos listados acima, o ambiente do *prompt* de comando *Cloud9* está pronto para receber os comandos de *CRUD* no banco de Dados *DocumentDB*.

A partir disso, foi requerido a criação de um modelo de dados e a apresentação das entradas no banco de dados não-relacional por meio de comandos *CRUD*.

Para o projeto proposto, foi criado um modelo de uma garagem de carros para venda, com as seguintes entidades:

- Carro
- Fabricante
- Proprietário





- Serviço
- Revisão

Adicionalmente, para as ações de *CRUD* no DocumentDB, por serem dados não relacionais, buscam os valores de cada chave de acordo com o comando escolhido, similar a um banco SQL.

- Criar (INSERT no SQL)

```
db.carros.insert({modelo: "Civic", fabricante:
"Honda", ano: 2022})
```

- Ler (SELECT no SQL)

```
db.carros.find({fabricante: "Honda"})
```

- Atualizar (UPDATE no SQL)

```
db.carros.update({modelo: "Civic"}, {$set: {ano:
2023}})
```

- Deletar (DROP no SQL)

```
db.carros.remove({modelo: "Civic"})
```

Deste modo, usando os comandos de *Insert* no banco de dados, foram inseridos os registros para cada uma das coleções dos modelos, como no exemplo abaixo:

```
db.fabricante.insert({"id": 1, "nome": "Honda",
"pais": "Japão", "fundacao": 1948})
```

Após a inserção dos dados, foi feita uma demonstração de consulta para a classe, para exemplificar como é feito um comando read similar ao SELECT do SQL:

```
db.carros.find({fabricante: "Honda"})
```

Foi ressaltado também que no caso da criação de relações entre as coleções de documentos (cada documento é uma linha respectivamente) podem ser feitas por meio de “relacionamento por referência”, ou então, a própria flexibilidade do banco *NoSQL* consegue que as especificações (ou





valores) de uma chave estejam dentro do próprio documento, como um relacionamento aninhado:

```
{
  "id": 1,
  "modelo": "Civic",
  "fabricante": {"id": 1, "nome": "Honda"}
}
```

6.3 CONCLUSÃO

A partir da construção do banco de dados em ambiente AWS, inserção dos registros em seus respectivos modelos e explicação em sala de aula, conclui-se o projeto de criação de banco de dados *DocumentDB*.

Ao longo deste capítulo, houve aprendizagem no universo dos bancos de dados, focando especificamente na construção e utilização do *DocumentDB*. Este projeto proporcionou uma imersão profunda no campo de prática em gerenciamento de banco de dados, destacando importantes aprendizados e insights.

O primeiro ponto que se destaca é a compreensão aprimorada do paradigma não-relacional. Ao confrontar e comparar modelos de dados relacionais e não-relacionais, percebe-se a versatilidade do *DocumentDB* em lidar com estruturas dinâmicas e semiestruturadas. A flexibilidade inerente ao modelo de documentos revelou-se valiosa ao enfrentar desafios associados à variedade e evolução dos dados.

Adicionalmente, destacou-se a versatilidade do serviço em ter a compatibilidade com *MongoDB*, usando a interface deste. Ao utilizar o *DocumentDB* para evidenciar uma ferramenta de banco de dados *NoSQL* mostra que o *MongoDB*, apesar de ser expoente nesse modelo de dados, mostra versatilidade dentro da nuvem permitindo que sejam usadas código e ferramentas existentes no *MongoDB* para interagir com o *DocumentDB*.

A migração para ambientes em nuvem, especificamente utilizando a AWS com o *DocumentDB*, adicionou uma camada adicional de complexidade, mas também ofereceu oportunidades significativas. A elasticidade e escalabilidade proporcionadas pela nuvem foram considerados fortes fatores para associar um banco de dados não-relacional à nuvem.





No contexto do *CRUD* em bancos de dados não-relacionais, ficou entendido que as operações não são apenas uma tradução direta das práticas em bancos de dados relacionais. O *DocumentDB* introduz abordagens distintas para a manipulação de dados, enfatizando a criação e atualização flexíveis de documentos.

Em resumo, a experiência prática com a construção do *DocumentDB* proporcionou uma visão abrangente do papel vital dos bancos de dados na gestão eficiente de dados semiestruturados. A familiaridade com as diferenças entre modelos de dados e a aplicação prática do *CRUD* em ambientes não-relacionais ampliaram significativamente nosso conhecimento e habilidades nesta área.





7 MANUAL DO PORTFÓLIO

No decorrer destes cinco semestres da graduação em Análise e Desenvolvimento de Sistemas, foram desenvolvidas atividades relevantes para o mercado de tecnologia, tanto quanto um conjunto de conceitos teóricos importantes para a continuidade do aprendizado na área da tecnologia da informação.

Como apresentação do conhecimento adquirido, foi implementado um portfólio digital com o objetivo de demonstrar as atividades e projetos propostos em cada um dos semestres relevantes. Detalhe especial para o 5º Semestre letivo, que possui dois projetos devido o adiantamento da matéria.

Para este portfólio, foram usadas as mesmas técnicas de desenvolvimento *web* praticadas no decorrer da graduação, usando HTML, CSS e a biblioteca *Materialize* como estrutura principal do código e da página.

Como uma breve explicação, o *Materialize* é um *framework front-end* responsivo baseado nas diretrizes de *Material Design* do Google. Ele fornece um conjunto de componentes de interface do usuário pré projetados, estilos em CSS e funcionalidades *JavaScript* para simplificar e aprimorar o desenvolvimento de design web. Dentre os pontos fortes desse *framework*, pode-se destacar a responsividade do design, os componentes prontos para o uso e a facilidade de usá-los, a flexibilidade no uso do CSS permitindo sua customização, funcionalidades em *JavaScript* e uma comunidade de suporte forte e presente.

Outro ponto relevante do desenvolvimento deste portfólio foi levar em consideração os conceitos aprendidos de *User Experience (UX)*. Dentro de um dos projetos apresentados no próprio portfólio digital, muito falou-se do quão importante é uma usabilidade fluida, de fácil entendimento e muitas vezes prazerosa.

Levando esse quesito em consideração, usou-se uma página única com seções separadas que podem ser acessadas por meio da barra de navegação superior, ou então pela barra de rolagem. A página é dividida em seis componentes, que são: “*Landing Page*”, “Sobre mim”, “Conhecimentos”, “Experiência”, “Portfólio” e “Contato”.





7.1 LANDING PAGE (SEÇÃO PRINCIPAL)

Nesta página principal, é feita uma apresentação do portfólio e uma mensagem de apresentação de quem sou eu e algumas informações básicas. Foi usado um componente de “carrossel” para a passagem de uma tela para a outra de forma dinâmica e interativa, o que permite uma sensação boa ao navegar pela página.

Acima da *Landing Page*, pode-se ver a barra de navegação superior com as seções da página, que, ao serem clicados, “escorregam” para a seção correspondente, também destacando uma boa interatividade com o usuário.

A ideia de uma *Landing Page* com um conteúdo que pode ser rolado para baixo em uma única página permite uma interação moderna e prazerosa juntamente com uma simplicidade de estrutura que tem sido usada comumente na internet.

Figura 7.1 – *Landing Page* do Portfólio



Fonte. Elaborado pelo Autor, 2023

7.2 SOBRE MIM

A próxima seção é a página onde defino algumas características e histórico de trabalho, além de informações úteis da minha trajetória profissional.

É uma parte básica da página em si pois complementa a introdução da seção de *Landing Page* e instiga o usuário e buscar mais informações.

Como destaque, percebe-se uma foto de perfil ao lado do texto, detalhe para qual adiciona mais conteúdo à seção.





Figura 7.2 – Seção “Sobre Mim”



Fonte. Elaborado pelo Autor, 2023

7.3 CONHECIMENTOS

Nesta seção, procura-se elencar as principais habilidades desenvolvidas no decorrer do desenvolvimento acadêmico, com maior enfoque neste caso para as ferramentas de tecnologia dos quais fiz parte e me ajudaram na capacitação profissional.

Usei um componente do próprio *Materialize* conhecido como “barra de progresso” para representar o nível técnico que possuo em cada ferramenta/linguagem descrita.

Essas barras de progresso são uma característica típica do *Materialize* para visualizar níveis de habilidade ou conclusão em determinadas áreas. O estilo e a aparência são fornecidos pelo *Materialize CSS framework*.

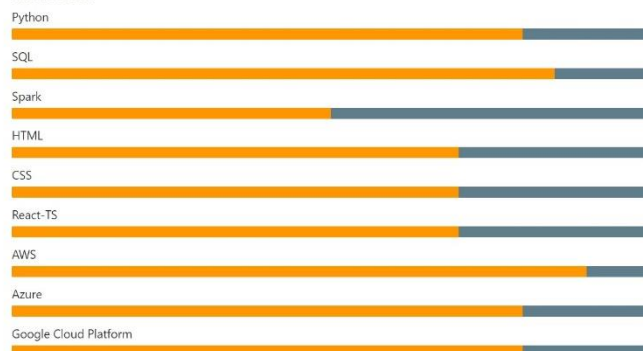




Figura 7.3 – Seção “Conhecimentos”

Conhecimentos

Por estar focado em dados e aplicações em nuvem, busquei conhecer as principais linguagens usadas nesse ambientes e participei de projetos importantes dentro da empresa onde trabalho, ganhando mais conhecimento em programação, algoritmos e engenharia de software e Dados.



Fonte. Elaborado pelo Autor, 2023

7.4 EXPERIÊNCIA

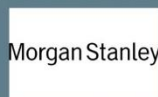
Na seção “Experiência”, encontra-se um breve resumo da minha carreira profissional, destacando os logotipos das empresas para qual trabalhei. Há também uma breve descrição acima para complementar as imagens dos logotipos.

É uma seção pequena que tem por objetivo resumir parte da trajetória profissional.

Figura 7.4 – Seção “Experiência”

Experiência

Trabalhei em importantes empresas do mercado, em diferentes ramos de negócio dentro de diferentes funções. Isso me deu muito conhecimento de negócios, soft skills importante e grande capacidade de gerenciar projetos extensos e cruciais no desenvolvimento da empresa.



Fonte. Elaborado pelo Autor, 2023





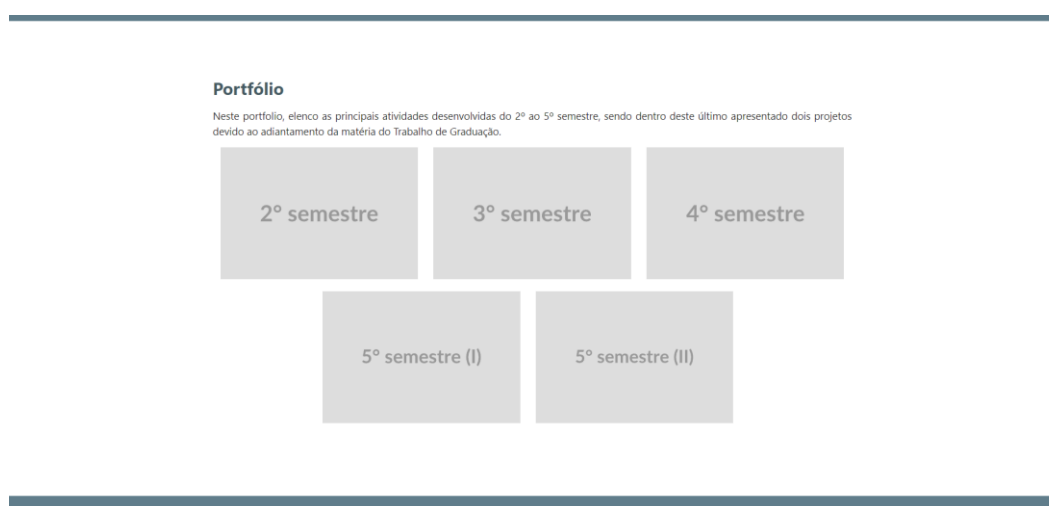
7.5 PORTFÓLIO

A principal seção do portfólio é a que tem o mesmo nome do projeto principal que. Nesta seção, compreendem os cinco projetos desenvolvidos ao longo do curso em formato de blocos usando o *grid* específico do *Materialize* com auxílio de CSS para formatação.

Antes do início dos blocos, há também uma descrição do que propõe a seção para destaque dos projetos dentro dos blocos.

Cada bloco possui uma animação de *hover* (passagem do *mouse* por cima do elemento) que o destaca ao usuário caso queira clicar nele. Há também um título incorporado do site *placeholder.co*, que basicamente cria um *placeholder* de maneira fácil e prática para incorporação em páginas *web* e documentos.

Figura 7.5 – Seção “Portfólio”



Fonte. Elaborado pelo Autor, 2023

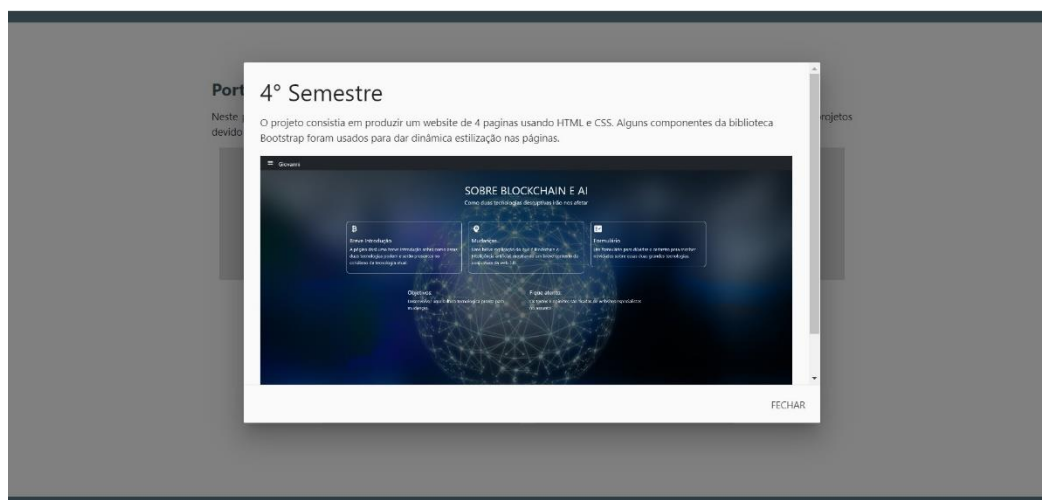
Dentro de cada bloco, há um *modal*. Modal é uma tela (janela) que bloqueia a tela (janela) principal. Ela abre em cima de uma tela e não permite que se interaja com a tela parente enquanto não concluir a ação que a tela modal quer que você faça. Para isso, possui um comando “fechar” no modal para que se volte para a página inicial e navegue normalmente pela página.

Além de uma imagem resumindo o que foi cada projeto, tem ainda uma descrição do que foi o projeto e o principal objetivo de cada um.





Figura 7.6 – Modal da Seção “Portfólio”



Fonte. Elaborado pelo Autor, 2023

7.6 CONTATO

A última seção da página consiste nos meus contatos caso haja o interesse.

A barra horizontal do final da página contendo essa seção deixa uma mensagem ao usuário para que busque a conexão seja por motivos pessoais e profissionais.

Cada bloco da seção “Contato” é um *link* para o respectivo tipo de contato, sendo eles o *email*, *Whatsapp* e *Linkedin*. Também foi disponibilizado o currículo em formato .pdf que abre em uma aba separada, disponibilizando-o para *download*.

Figura 7.7 –Seção “Contato”



Fonte. Elaborado pelo Autor, 2023





8 CONSIDERAÇÕES FINAIS

Ao fim deste trabalho, pode-se passar por vários projetos que contribuíram imensamente para o desenvolvimento de habilidades cruciais para o desenvolvimento em tecnologia da informação e para o mercado de trabalho.

Resgatando os projetos passados com suas respectivas disciplinas, cada um deles foi imprescindível para a conclusão desse portfólio, abordando diversos temas.

Para o primeiro projeto, foi feito um protótipo de interface para um sistema de gestão acadêmica usando o *Figma*, uma plataforma *open-source* para prototipação. Explorou-se conceitos importantes de *IU/UX*, percebendo sempre a interação com o usuário como prioridade e outros parâmetros da disciplina de *IHC* como design e interatividade fluida para uma experiência agradável e funcional.

Para o segundo projeto, focou-se na criação de uma livreria virtual usando a linguagem de programação C++. Neste aspecto, foi importante o conhecimento de lógica de algoritmos e de estrutura de dados. Também foi explorado o conceito de arquivos em banco de dados, *structs* e formas de entrada e saída de registros dentro de um banco de dados simples.

Já no terceiro projeto, foi construído um *website* usando *HTML* e as bibliotecas de estilização *CSS* e, principalmente, o *Bootstrap*. Foi de extrema valia o conhecimento adquirido em programação *front-end*, a conceituação de “desenvolvimento *web*”, sua relevância nos dias de hoje e os conceitos de estilização, assim como interface com o usuário.

No quarto projeto, houve a elaboração de uma aplicação feita com *HTML* e *JQUERY*, uma biblioteca *JavaScript* muito intuitiva e com muito alcance para seletores e funções de manipulação de *DOM* no código principal. Além de ter sido uma experiência única no desenvolvimento *mobile* na minha carreira, garantiu também mais do que tudo as práticas de desenvolvimento em geral, procedendo com consultas *online* e aplicando conhecimento adquirido diretamente na internet para elaborar um trabalho distinto dos demais e com um design e interface próprios.





No quinto e último projeto do quinto semestre, foi construído em ambiente de nuvem AWS um banco de dados não-relacional *DocumentDB*. A atividade proporcionou um conhecimento a mais, que foi a utilização de serviços em nuvem em conta *free tier* e a versatilidade de criar registros dentro da nuvem, diferentemente do banco de dados não-relacional mais usado, o *MongoDB*. O projeto foi finalizado expondo para a sala de aula, ganhando assim um caráter extra na didática e aprimoramento de *soft skills*.

O projeto final foi a apresentação de todos os projetos em um só, culminando em um Portfólio Digital dentro da *web*, destacando minhas experiências acadêmicas e profissionais, além de aplicar um pouco do conhecimento adquirido durante o curso na própria página, usando a biblioteca *Materialize*, juntamente com *HTML* e *CSS* para estilização.

Com este último feito, expõe as atividades em um único lugar e apresenta-se como divulgador de conteúdo, tendo como objetivo a visibilidade no mundo virtual e a troca de informações com usuários que venham a se interessar pela página e pelo conteúdo que desenvolvi usando meu conhecimento.

Finalizando, pode-se concluir que ter um portfólio abre portas e gera sabedoria ao passo que nos traz desafios distintos de reconhecimento do que foi exercitado e aprendido durante os semestres do curso de Análise e Desenvolvimento de Sistema, proporcionando bagagem para evoluir profissionalmente e pessoalmente.





REFERÊNCIAS BIBLIOGRÁFICAS

HEWETT, T. T., Baecker, R. M., Card, S. K., Carey, T., Gasen, J., Mantei, M., Perlman, G., Strong, G., & Verplank, W. (1992). *ACM SIGCHI Curricula for Human-Computer Interaction*. SIGCHI Bulletin, 24(1), 21-28.

NIELSEN, J. (1993). *Usability Engineering*. Academic Press.

CHIEF OF DESIGN. **Fundamentos do Design Visual**. Disponível em: <https://chiefofdesign.com.br/ihc-interacao-humano-computador/>. Acesso em: 08 de maio 2023.

WIKIPEDIA. **Figma**. Disponível em: <<https://en.wikipedia.org/wiki/Figma>>. Acesso em: 08 de maio 2023.

GONZALEZ, R. (2020). **Figma, o programa de design que está revolucionando a indústria tecnológica**. WIRED. Disponível em: <<https://www.wired.co.uk/article/figma-design-app>>. Acesso em: 08 de maio 2023.

LIVING THE INTERNET. **History of the C Programming Language (em inglês)**. <https://livinginternet.com/i/iw_unix_c.htm>. Acesso em: 25 de abril de 2023.

CPROGRAMMINGLANGUAGE.COM. **C – The Influence Factor of Many Popular Programming Languages (em inglês)**. <<https://www.cprogramminglanguage.com.>> Acesso em 25 de abril de 2023.

CALLE, J. D. **Introdução à Programação Orientada para Objetos em Linguagem C++**. 2017. Universidade de São Paulo – USP.

PUCRS. **Structs**. <<https://www.inf.pucrs.br/~pinho/Laprol/Structs/Structs.htm>>. Acesso em: 21 de maio 2023.

SANTIAGO, C. P., Veras, N. L., de Aragão, A. P., Carvalho, D. A., & Amaral, L. A. **Desenvolvimento de sistemas Web orientado a reuso com Python, Django e Bootstrap**. 2020. *Sociedade Brasileira de Computação*.

SPURLOCK, J. **Bootstrap: responsive web development (em inglês)**. 2013. O'Reilly Media, Inc.





SILVA, Maurício Samy. ***JQuery Mobile: desenvolva aplicações web para dispositivos móveis com HTML5, CSS3, Ajax, jQuery e jQuery UI***. 2012. Novatec Editora. São Paulo – SP.

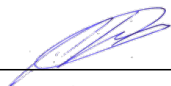
DATE, C J. **Introdução a sistemas de banco de dados**. 2004. Elsevier. Rio de Janeiro.

SILBERSCHATZ, Abraham. **Sistema de Banco de Dados**. 2006. Elsevier. Rio de Janeiro.

MONGODB. ***What is a Document Database? (em inglês)***. Disponível em: [https:// www.mongodb.com/pt-br/document-databases/](https://www.mongodb.com/pt-br/document-databases/). Acesso em: 20 de novembro de 2023.



Página de assinaturas



Anderson Pazin
264.548.978-85
Signatário



DIEGO LAZARO
341.175.938-03
Signatário



Felipe Rodrigues
345.484.558-09
Signatário



Julio Lieira
080.769.668-41
Signatário

HISTÓRICO

19 dez 2023 11:13:44		Anderson Pazin criou este documento. (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85)
19 dez 2023 11:13:44		Anderson Pazin (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85) visualizou este documento por meio do IP 177.95.133.194 localizado em Guaratingueta - Sao Paulo - Brazil
19 dez 2023 11:13:47		Anderson Pazin (E-mail: anderson.pazin@fatec.sp.gov.br, CPF: 264.548.978-85) assinou este documento por meio do IP 177.95.133.194 localizado em Guaratingueta - Sao Paulo - Brazil
19 dez 2023 17:54:55		Felipe Maciel Rodrigues (E-mail: felipe.rodrigues30@fatec.sp.gov.br, CPF: 345.484.558-09) visualizou este documento por meio do IP 201.182.122.11 localizado em Lins - Sao Paulo - Brazil
19 dez 2023 17:55:00		Felipe Maciel Rodrigues (E-mail: felipe.rodrigues30@fatec.sp.gov.br, CPF: 345.484.558-09) assinou este documento por meio do IP 201.182.122.11 localizado em Lins - Sao Paulo - Brazil
20 dez 2023 15:36:00		Julio Fernando Lieira (E-mail: julio.lieira3@fatec.sp.gov.br, CPF: 080.769.668-41) visualizou este documento por meio do IP 189.126.178.186 localizado em Lins - Sao Paulo - Brazil
20 dez 2023 15:36:12		Julio Fernando Lieira (E-mail: julio.lieira3@fatec.sp.gov.br, CPF: 080.769.668-41) assinou este documento por meio do IP 189.126.178.186 localizado em Lins - Sao Paulo - Brazil
19 dez 2023 11:47:52		DIEGO HENRIQUE EMYGDIO LAZARO (E-mail: diego.lazaro@fatec.sp.gov.br, CPF: 341.175.938-03) visualizou este documento por meio do IP 201.182.122.11 localizado em Lins - Sao Paulo - Brazil



19 dez 2023

11:47:55



DIEGO HENRIQUE EMYGDIO LAZARO (E-mail: diego.lazaro@fatec.sp.gov.br, CPF: 341.175.938-03) assinou este documento por meio do IP 201.182.122.11 localizado em Lins - Sao Paulo - Brazil

