UNIFEOB
ESCOLA DE NEGÓCIOS

2024

PROJETO INTEGRADO

Ciencia

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

DESENVOLVIMENTO DE SOLUÇÕES CONSOLE INTEGRADAS PARA EDUCAÇÃO, SUSTENTABILIDADE, INCLUSÃO SOCIAL E EMPREENDEDORISMO

FUNILARIA STREET CARS TRUCKS

SÃO JOÃO DA BOA VISTA, SP

NOVEMBRO 2024

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS
CIÊNCIA DA COMPUTAÇÃO

PROJETO INTEGRADO

DESENVOLVIMENTO DE SOLUÇÕES CONSOLE INTEGRADAS PARA EDUCAÇÃO, SUSTENTABILIDADE, INCLUSÃO SOCIAL E EMPREENDEDORISMO

FUNILARIA STREET CARS TRUCKS

MÓDULO MODELAGEM E DESENVOLVIMENTO DE SISTEMAS

Business Intelligence – Profª. Mariângela Martimbianco Santos

Programação Orientada a Objeto – Prof. Nivaldo de Andrade

Lógica de Programação – Prof. Marcelo Ciacco Almeida

Modelagem de Dados – Prof. Max Streicher Vallim

Projeto de Modelagem e Desenvolvimento de Sistemas – Profª. Mariângela M. Santos

Estudantes:

Gustavo Miguel da Silva Viti, RA 24001188
João Vitor Toledo da Silva, RA 24000550
Pedro de Freitas da Silva, RA 24000809
Yasmin Beatriz Silva Ruela, RA 24000753

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2024

SUMÁRIO

1. INTRODUÇÃO 5
2. DESCRIÇÃO DA EMPRESA 6
3. PROJETO INTEGRADO 7
   3.1 PROGRAMAÇÃO ORIENTADA A OBJETO 7
      3.1.1 CLASSES E OBJETOS 8
      3.1.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO. 9
      3.1.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS 11
   3.2 LÓGICA DE PROGRAMAÇÃO 12
      3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE 13
      3.2.2 DESENVOLVIMENTO DE APLICAÇÕES 14
      3.2.3 IMPLEMENTAÇÃO E VALIDAÇÃO 15
   3.3 MODELAGEM DE DADOS 16
      3.3.1 MODELO CONCEITUAL 16
      3.3.2 MODELO LÓGICO E FÍSICO 18
      3.3.3 SQL 19
   3.4 BUSINESS INTELLIGENCE 21
      3.4.1 ORGANIZAÇÃO E IDENTIFICAÇÃO DAS INFORMAÇÕES 21
      3.4.2 MANIPULAÇÃO E ANÁLISE DE DADOS 22
      3.4.3 CRIAÇÃO DE MODELOS DE ANÁLISE DE DADOS 23
   3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS 24
      3.5.1 GERENCIANDO FINANÇAS 24
      3.5.2 ESTUDANTES NA PRÁTICA 26
4. CONCLUSÃO 27

REFERÊNCIAS 28

ANEXOS 29

ISSN 1983-6767

1. INTRODUÇÃO

Atualmente, empresas em diversos setores buscam soluções informatizadas para melhorar a gestão eficiente de seus recursos. Empresas de diferentes setores implementam sistemas informatizados, seja por meio de desenvolvimento interno ou de contratação terceirizada. Esses sistemas têm a capacidade de melhorar a organização empresarial, proporcionando uma gestão mais visual e eficiente dos recursos. Dentre as organizações que estão adotando essa estratégia, as funilarias vêm investindo significativamente na informatização de seus processos. De acordo com Souza (2024), a adoção de tecnologias, como sistemas de gestão de estoque, têm otimizado processos e aumentado a eficiência operacional nas empresas de funilaria automotiva no Brasil. Isso reflete a necessidade crescente de automação em processos que, até recentemente, eram majoritariamente manuais. Segundo uma pesquisa realizada pela IBM, 78% dos líderes empresariais brasileiros estão investindo em tecnologias como automação e inteligência artificial, visando otimizar suas operações nos próximos anos (IBM, 2023).

Sendo assim, o projeto tem como objetivo o desenvolvimento de um sistema via console para uma empresa de funilaria. Este sistema será desenvolvido em Python, integrado com banco de dados SQL. O intuito é otimizar a gestão interna, oferecendo uma solução eficaz por meio da integração de dados. Como complemento, será implementado um dashboard em Power BI, que permitirá a visualização das informações de forma mais clara e objetiva, auxiliando na tomada de decisões estratégicas.

O projeto visa modernizar e informatizar a empresa escolhida, que ainda depende de processos manuais. A criação de um sistema personalizado contribuirá diretamente para a melhoria geral dos processos, proporcionando maior eficiência e agilidade no nível operacional, com impacto positivo nos lucros. Isso é corroborado pela pesquisa da Folha Vitória, que aponta que 78% das empresas brasileiras já utilizam automação, sendo que 52% delas possuem experiência prática com essas ferramentas. Esse movimento está diretamente relacionado à busca por eficiência operacional, redução de custos e aumento da produtividade (Folha Vitória, 2024).

Além dos aspectos técnicos trabalhados no projeto, o PI proposto busca desenvolver duas soft skills essenciais para o mercado de trabalho: o trabalho em equipe e a organização. Esses elementos serão fundamentais ao longo do desenvolvimento do sistema, que precisará ser dividido em diversas partes para que o cronograma seja cumprido conforme o planejado, exigindo também uma organização rigorosa para evitar falhas ou atrasos no projeto.

Portanto, este trabalho está estruturado da seguinte forma: primeiro, será feita uma descrição da empresa. Em seguida, será apresentada a correlação entre as unidades de ensino e sua aplicação no projeto. Posteriormente, o desenvolvimento do projeto será descrito detalhadamente, incluindo a integração teórica e prática, culminando em uma conclusão que destaca as lições aprendidas e os resultados alcançados.

2. DESCRIÇÃO DA EMPRESA

A Street Cars Trucks é uma empresa de funilaria localizada em Vargem Grande do Sul, SP, especializada no reparo e manutenção de carrocerias de caminhões para diversas empresas da região. Seu portfólio de serviços inclui desde a recuperação de carrocerias até a personalização e ajustes específicos, atendendo tanto pequenas transportadoras quanto grandes empresas que dependem da segurança e qualidade das carrocerias para suas operações diárias. Além disso, a Street Cars Trucks é reconhecida pela ampla oferta de peças e acessórios para caminhões, tais como adesivos, geladeiras, escapamentos, entre outros itens que são indispensáveis para o conforto e segurança dos motoristas. Com uma equipe altamente qualificada, a empresa busca oferecer um serviço de excelência, priorizando a satisfação e a fidelização de seus clientes.

Apesar de sua trajetória consolidada no mercado, a empresa ainda enfrenta um desafio significativo: a falta de informatização em seus processos administrativos e operacionais. Atualmente, a gestão das ordens de serviço, controle de estoque e atendimento ao cliente ainda é realizada de maneira manual, por meio do tradicional "papel e caneta" — uma prática que, no contexto tecnológico atual, limita a eficiência e a capacidade de expansão dos negócios. Esse modelo, além de suscetível a erros, compromete a agilidade no atendimento e dificulta o controle preciso de materiais e processos.

Diante dessa situação, a Street Cars Trucks demonstrou interesse em adotar um sistema informatizado que possa centralizar as informações, automatizar processos e otimizar o atendimento ao cliente. A expectativa é que essa solução tecnológica permita uma visão clara e integrada dos dados operacionais e ofereça recursos que auxiliem na tomada de decisões estratégicas, além de possibilitar uma gestão mais precisa dos recursos, contribuindo para a modernização e crescimento da empresa no mercado competitivo de serviços de funilaria e acessórios para caminhões.

ISSN 1983-6767

3. PROJETO INTEGRADO

Conforme apresentado na introdução, este capítulo visa demonstrar o conhecimento teórico e sua aplicação prática no projeto, sendo estes, divididos entre cada unidade de ensino disponível.

3.1 PROGRAMAÇÃO ORIENTADA A OBJETO

A Programação Orientada a Objetos (POO) é um paradigma que organiza o desenvolvimento de sistemas em objetos, os quais representam entidades do mundo real ou conceitos abstratos que podem ser modelados em software. Esse modelo permite uma abordagem mais intuitiva e natural ao desenvolvimento de software, pois os programadores podem trabalhar com representações que se assemelham a como entendemos e interagimos com o mundo à nossa volta. Segundo Fernanda Farinelli (2007), "A Orientação a Objetos é uma tecnologia que enxerga os sistemas como sendo uma coleção de objetos integrantes. Ela permite melhorar a reusabilidade e extensibilidade dos softwares". Essa perspectiva facilita a compreensão e a manutenção do código, uma vez que os sistemas se tornam mais alinhados com a lógica e a estrutura do mundo real.

Uma das principais vantagens da POO é sua capacidade de promover a reusabilidade de código. Ao permitir que componentes de software sejam criados como unidades independentes, a POO possibilita que partes do código sejam reutilizadas em diferentes partes de um projeto ou mesmo em diferentes projetos. Essa característica não só economiza tempo durante o desenvolvimento, mas também minimiza a ocorrência de erros, já que os componentes reutilizados podem ter sido testados e otimizados em implementações anteriores. Essa modularidade é crucial em um ambiente de desenvolvimento ágil, onde a eficiência é essencial.

Outro aspecto importante da POO é a extensibilidade. À medida que novos requisitos surgem ou que as necessidades dos usuários evoluem, o paradigma permite que sistemas existentes sejam ampliados sem a necessidade de reescrever o código do zero. Os desenvolvedores podem adicionar novas funcionalidades ou modificar comportamentos existentes, mantendo a integridade do sistema. Isso é especialmente valioso em cenários corporativos, onde as demandas de negócios podem mudar rapidamente.

A POO também contribui para a segurança da informação. Ao encapsular dados dentro de objetos e controlar o acesso a eles, é possível proteger informações sensíveis e garantir que apenas operações autorizadas possam modificar o estado de um objeto. Isso é particularmente relevante em aplicações que lidam com dados críticos, como sistemas financeiros e de saúde, onde a proteção de dados pessoais e a conformidade com regulamentações são primordiais.

Além disso, a POO oferece uma abordagem que favorece a colaboração entre desenvolvedores. Com a clareza que os objetos proporcionam, diferentes equipes podem trabalhar em módulos independentes de um sistema. Essa divisão de responsabilidades não só acelera o desenvolvimento, mas também melhora a comunicação entre os membros da equipe, pois todos compartilham uma compreensão comum de como os objetos interagem dentro do sistema.

Por fim, a Programação Orientada a Objetos é um paradigma altamente valorizado que se destaca pela sua capacidade de criar soluções de software que são não apenas funcionais, mas também organizadas, seguras e adaptáveis. Em um mundo onde a tecnologia evolui rapidamente, a POO continua a ser uma abordagem fundamental para o desenvolvimento de software, permitindo que empresas e desenvolvedores enfrentem os desafios contemporâneos de forma eficaz e inovadora.

3.1.1 CLASSES E OBJETOS

Na Programação Orientada a Objetos (POO), classes e objetos são elementos fundamentais que formam a espinha dorsal deste paradigma, sendo essenciais tanto no desenvolvimento de projetos complexos quanto em projetos menores. Esses dois elementos são a base da orientação a objetos, e tudo gira em torno deles. Compreender suas características e como se inter-relacionam é crucial para aproveitar ao máximo os benefícios da POO.

Classe: Uma classe pode ser vista como um modelo ou uma "receita" para criar objetos. Ela define a estrutura que um objeto terá, especificando quais dados (atributos) ele armazenará e quais operações (métodos) poderão ser realizadas sobre esses dados. Dentro da classe, são determinados os valores padrão, as variáveis de instância e as funções que operam sobre esses dados. Por exemplo, uma classe pode ter atributos como `nome`, `idade` e `endereço`, que representam as propriedades de uma entidade, além de métodos que definem ações que essa entidade pode realizar.

A definição de uma classe é apenas uma abstração; ela não ocupa espaço em memória até que um objeto seja criado a partir dela. É na classe que se estabelece a lógica de funcionamento e as características do que será produzido. Isso significa que, ao alterar a classe, você pode afetar todas as instâncias (objetos) criadas a partir dela, tornando o desenvolvimento mais eficiente e modular.

Objeto: O objeto, por sua vez, é uma instância concreta da classe. Quando você cria um objeto, está gerando uma cópia real da estrutura definida na classe, com dados específicos atribuídos a seus atributos. Por exemplo, ao instanciar um objeto, você define seus atributos como nome = "Maria", idade = 25 e endereço = "Rua B, 456". Dessa forma, esse objeto é uma representação específica da classe que possui todas as características e comportamentos definidos.

A diferença entre classe e objeto é essencial para entender a POO. Enquanto a classe é uma abstração que fornece a estrutura, o objeto é a aplicação prática dessa estrutura, com dados concretos. Um objeto pode ser considerado como uma "instância" viva da classe, que pode ser manipulada e utilizada em um programa. Essa distinção permite uma grande flexibilidade no desenvolvimento de software, onde múltiplas instâncias de uma mesma classe podem coexistir, cada uma com seus próprios estados e comportamentos.

Essa relação entre classes e objetos não apenas facilita a organização do código, mas também promove a reutilização e a modularidade, aspectos extremamente valiosos no desenvolvimento de software. Ao trabalhar com classes e objetos, os desenvolvedores podem criar sistemas mais robustos e escaláveis, capazes de se adaptar às necessidades em constante mudança do ambiente de negócios. A POO, ao permitir essa estrutura, proporciona um ambiente onde o código se torna mais legível, fácil de manter e ampliar.

3.1.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO.

A Programação Orientada a Objetos (POO) é estruturada para facilitar o desenvolvimento de sistemas grandes e complexos, mantendo o código organizado e compreensível. Para que a POO cumpra com essas promessas de modularidade, reutilização e escalabilidade, ela depende de alguns conceitos essenciais: atributos, métodos, encapsulamento, herança e polimorfismo. Esses elementos são fundamentais para organizar e criar objetos no desenvolvimento de software, oferecendo ferramentas que tornam o código mais modular e intuitivo para quem conhece os princípios da POO. Além disso, esses

conceitos facilitam a expansão de projetos menores e suportam a criação de aplicações de grande porte. “A Programação Orientada a Objetos (POO) proporciona melhor organização do código e facilita a manutenção e a expansão de sistemas, especialmente em projetos de grande escala” (DNC, 2024)

Atributo: Os atributos representam as características de um objeto, definindo seus dados e propriedades. Em um exemplo simples, se o objeto for um Veículo, seus atributos podem incluir informações como cor, marca, modelo, ano de fabricação e número de portas. Esses atributos ajudam a descrever cada objeto de maneira única e concreta, diferenciando-o de outros objetos da mesma classe.

Método: Os métodos são as “ações” ou “comportamentos” que um objeto pode realizar. Eles representam funções que manipulam os dados dos atributos ou permitem que o objeto interaja com outros elementos do sistema. Por exemplo, em um objeto Veículo, os métodos poderiam incluir ligar, desligar, acelerar e frear. Assim, os métodos tornam os objetos ativos e interativos dentro de um sistema, permitindo que eles executem tarefas específicas de maneira controlada.

Encapsulamento: O encapsulamento é um conceito central da POO que visa proteger os dados dos objetos, permitindo que apenas determinados métodos possam acessar ou modificar seus atributos. Isso é feito definindo níveis de acesso, como público, privado ou protegido, que controlam quais dados são visíveis ou modificáveis por outros objetos ou classes. O encapsulamento garante a integridade dos dados e aumenta a segurança do sistema, evitando que informações sensíveis sejam alteradas indevidamente.

Herança: A herança é uma ferramenta poderosa que permite criar novas classes a partir de classes existentes, herdando seus atributos e métodos. Com a herança, é possível organizar e otimizar o código, evitando duplicidade de funções e promovendo a reutilização. Seguindo o exemplo dos automóveis, imagine que criamos uma classe Automóvel com os atributos e métodos comuns, como andar e ligar. Então, criamos subclasses específicas como Carro e Moto, que herdam os atributos e métodos de Automóvel, mas podem adicionar ou modificar funcionalidades conforme necessário. Esse mecanismo reduz redundâncias e facilita a manutenção do código.

Polimorfismo: O polimorfismo permite que métodos com o mesmo nome tenham comportamentos diferentes de acordo com a classe em que estão implementados. Esse conceito é útil para personalizar ações que são comuns a vários objetos, mas que se comportam de maneira específica para cada tipo de objeto. Por exemplo, em uma classe Casa, objetos como Janela e Porta podem ter o método abrir. No entanto, cada um desses objetos

abre de maneira diferente, e o polimorfismo permite que o método abrir seja adaptado para o funcionamento correto de cada um. Assim, o polimorfismo trabalha em conjunto com a herança para tornar o código mais flexível e versátil.

Em resumo, os conceitos de atributos, métodos, encapsulamento, herança e polimorfismo são a base que torna a Programação Orientada Objeto eficaz para o desenvolvimento de sistemas organizados e escaláveis. Esses elementos possibilitam a criação de estruturas de software complexas, facilitando a manutenção e promovendo a expansão do sistema de forma ordenada e sustentável.

3.1.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS

Um dos fundamentos centrais da Programação Orientada a Objetos (POO) é garantir a segurança e o controle sobre os métodos das classes. Para isso, a POO utiliza diferentes tipos de métodos, como os estáticos, públicos e privados, que determinam como e onde as operações podem ser executadas. "O encapsulamento é um dos pilares da Programação Orientada a Objetos, restringindo o acesso a certos componentes de um objeto e protegendo seu estado interno" (TACONTRATADO, 2024). Esses conceitos são cruciais para gerenciar interações entre objetos, estabelecendo limites claros no acesso às funções de uma classe e aumentando a segurança e a organização do sistema.

Métodos estáticos são aqueles que pertencem à própria classe, e não a uma instância específica dela. Ou seja, podem ser chamados diretamente pela classe, sem a necessidade de criação de um objeto. Esses métodos são utilizados para realizar tarefas que não dependem dos atributos específicos de uma instância, servindo, por exemplo, para operações de conversão de unidades, como converter uma temperatura de Celsius para Fahrenheit. Em sistemas de grande escala, métodos estáticos também podem ser úteis para cálculos genéricos ou validações de dados comuns a toda a classe, o que permite uma economia de recursos e melhora a performance do sistema.

Métodos públicos são acessíveis de qualquer parte do programa que possua visibilidade para a classe em que estão definidos. Esses métodos representam as ações principais que um objeto deve oferecer para interagir com outros componentes do sistema. Por exemplo, em uma classe de conta bancária, métodos públicos como depositar e sacar permitem que operações financeiras sejam realizadas de forma segura e controlada. Esse tipo de método facilita a comunicação entre os objetos e o acesso a funcionalidades externas, mas deve ser cuidadosamente planejado para não comprometer a integridade do sistema.

Métodos privados, por sua vez, são acessíveis apenas dentro da própria classe e ficam invisíveis para outras partes do programa. Eles geralmente contêm operações internas essenciais para o funcionamento da classe, mas que não devem ser expostas ou modificadas por objetos externos. Em uma classe conta bancária, um método privado que atualiza o saldo interno poderia ser utilizado para ajustar o saldo após transações, mas não estaria disponível para outros objetos fora da classe. Esse encapsulamento garante que o funcionamento interno da classe seja protegido, evitando que dados sensíveis sejam manipulados de maneira inadequada.

A utilização desses três tipos de métodos permite que os desenvolvedores controlem com precisão como as operações são realizadas e quem pode acessá-las, promovendo um ambiente seguro e bem-estruturado. A implementação adequada de métodos estáticos, públicos e privados contribui não apenas para a organização e modularidade do código, mas também para a expansão do sistema de maneira controlada, pois limita os riscos de alterações indesejadas e facilita a manutenção ao longo do ciclo de vida do software.

3.2 LÓGICA DE PROGRAMAÇÃO

O conceito de lógica pode ser definido como o estudo e a aplicação de princípios de raciocínio coerente, diretamente relacionado ao pensamento estruturado, utilizado na resolução de problemas, na análise de situações e na tomada de decisões de maneira racional e ordenada. De maneira simplificada, a lógica organiza o pensamento para que ele siga uma sequência clara e funcional. Um exemplo clássico do uso da lógica é observado em jogos de tabuleiro como Damas e Xadrez, nos quais cada jogada precisa ser estrategicamente planejada. Esses jogos não apenas desafiam o raciocínio lógico, mas também revelam a nossa capacidade de prever movimentos futuros, desenvolver estratégias e otimizar ações, demonstrando a aplicação prática da lógica.

Dentro da programação, surge o conceito de lógica de programação, que pode ser entendida como o conjunto de regras e técnicas que os programadores utilizam para projetar e desenvolver programas de computador. Essa lógica é essencial para decompor problemas complexos em etapas simples e viáveis de serem solucionadas por um algoritmo eficiente (ROCKETSEAT, 2024).

A lógica de programação desempenha um papel fundamental no desenvolvimento de software, pois garante a eficiência e a organização do código, facilitando a manutenção e a adaptação a novas linguagens. Entre as principais razões para a importância dessa habilidade, destacam-se:

- Resolução de problemas complexos;
- Otimização de eficiência;
- Facilidade de manutenção de sistemas;
- Adaptação a diferentes linguagens de programação;
- Criação de algoritmos eficazes.

Considerando todos esses aspectos, fica evidente a importância da lógica de programação no campo do desenvolvimento de software.

3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE

A lógica de programação constitui a base essencial para o desenvolvimento de software, uma vez que permite a criação de instruções e estruturas que solucionam problemas e executam tarefas de forma lógica e eficiente. Para isso, o entendimento de alguns conceitos fundamentais é indispensável. Dentre esses conceitos, os algoritmos, variáveis, tipos de dados, funções, estruturas condicionais e operadores representam elementos essenciais na construção de qualquer programa.

Algoritmos são sequências de instruções que descrevem de forma precisa as etapas necessárias para solucionar um problema específico. Um bom algoritmo é claro, eficiente e adaptável a diferentes contextos, servindo como um roteiro lógico para a criação de programas.

Variáveis e tipos de dados formam outro alicerce importante, pois permitem o armazenamento e a manipulação de informações dentro dos programas. Cada variável pode conter valores de tipos específicos, como números inteiros, decimais e textos. A compreensão dos tipos de dados e sua correta aplicação é fundamental para evitar erros e otimizar o uso de recursos na execução de programas.

Funções, por sua vez, são blocos de código organizados que executam tarefas específicas. Elas permitem dividir problemas complexos em partes menores e mais gerenciáveis, promovendo a reutilização de código e facilitando a manutenção.

As estruturas condicionais, como as instruções "If" e "Case", permitem que o programa tome decisões com base em condições específicas, adaptando o comportamento conforme as necessidades. Isso é especialmente importante para criar programas que respondam de forma inteligente a diferentes cenários.

Por fim, os operadores, sejam eles matemáticos, lógicos ou relacionais, possibilitam realizar comparações e cálculos entre dados, permitindo interações dinâmicas entre variáveis e condicionais.

Esses elementos, quando combinados, fornecem uma base sólida para a criação de sistemas e a solução de problemas computacionais de forma organizada, eficaz e estruturada.

3.2.2 DESENVOLVIMENTO DE APLICAÇÕES

Ao dar início a este projeto, decidiu-se desenvolver um software que atuasse como uma versão simplificada de um sistema ERP, com o objetivo de oferecer um controle eficaz sobre os vários elementos operacionais com os quais a empresa lida. Os sistemas ERP são altamente valorizados pela sua habilidade de integrar diversos processos de negócios, eliminando redundâncias e aprimorando a eficiência global. Conforme ressaltado, "o ERP é essencial para garantir a conformidade e proporcionar uma perspectiva unificada das operações, potencializando a eficiência e diminuindo despesas" (Alterdata, 2024).

Após definir as regras de negócios, é crucial implementar uma lógica estruturada que possa sustentar a operação diária da empresa com máxima eficiência. Dessa forma, o sistema de gestão de ordens de serviço se beneficiará de um algoritmo que prioriza atendimentos com base na urgência e localização. As funcionalidades detalhadas a seguir garantem que cada processo seja realizado de forma eficiente e integrada.

As normas de negócio do aplicativo focam no gerenciamento eficaz dos recursos empresariais, oferecendo recursos que aprimoram a administração de processos internos. Um exemplo crucial é o sistema de registro de pedidos, que guarda todos os dados pertinentes a cada pedido. Essas informações abrangem o cliente que solicitou, os colaboradores envolvidos, as peças empregadas, o custo total e a data de entrega. Esta metodologia

possibilita uma avaliação minuciosa e uma administração mais eficiente dos recursos, simplificando a tomada de decisões e a supervisão dos projetos.

O programa também proporciona recursos completos para o gerenciamento de clientes, colaboradores, inventário, veículos e outros elementos cruciais para assegurar uma administração eficaz e o funcionamento adequado de uma funilaria. Essas características foram desenvolvidas para aprimorar os processos operacionais, simplificando o acesso a informações relevantes e aprimorando a gestão geral da organização.

A lógica do sistema se fundamenta na gestão eficaz do banco de dados, empregando operações como “Inserts”, ”Deletes”, “Updates” e outras ferramentas para a correta manipulação. Para assegurar a execução adequada das ações, o sistema utiliza validações baseadas na estrutura condicional “If”, que avalia a intenção do usuário, solicita as informações requeridas e executa uma segunda validação para validar a execução com êxito. Se houver um erro, o programa emite uma mensagem de erro, assegurando que o usuário seja informado e possa retificar a entrada ou a ação realizada.

O projeto também adota a ideia de “Clean Code” , com o objetivo de tornar o código mais organizado, limpo e de fácil compreensão. Para tal, empregam-se nomes de funções claras e autoexplicativas, juntamente com uma estrutura de indexação meticulosamente estruturada. Esta estratégia simplifica a manutenção do código e a inclusão de novas funcionalidades ao software, otimizando e sustentando o processo de desenvolvimento.

3.2.3 IMPLEMENTAÇÃO E VALIDAÇÃO

A etapa de implementação e validação é um processo crucial para consolidar o desenvolvimento do sistema e garantir que ele funcione como um todo integrado e eficaz. Durante a implementação, o código é construído a partir dos requisitos e especificações definidos anteriormente, com os módulos e funcionalidades sendo desenvolvidos e organizados para atingir os objetivos propostos. É nessa fase que a lógica de programação de cada componente é aplicada para transformar ideias em uma solução prática.

Na validação, o foco está em garantir que todos os requisitos do projeto sejam atendidos. Para isso, a aplicação passa por uma série de testes, onde os alunos verificam se cada módulo responde corretamente a diferentes cenários e executa suas funções sem falhas. A validação também assegura que as operações interdependentes funcionem

harmoniosamente, garantindo que a aplicação final cumpra os critérios de qualidade e esteja pronta para o uso.

A implementação e a validação, portanto, são passos essenciais para garantir que o sistema desenvolvido seja robusto, eficiente e preparado para atender às necessidades para as quais foi projetado.

3.3 MODELAGEM DE DADOS

A modelagem de dados é uma das etapas mais importantes no desenvolvimento de um sistema, pois define como os dados serão armazenados, organizados e relacionados dentro de um banco de dados. Ela possibilita estruturar as informações de maneira lógica e eficiente, proporcionando uma base sólida para o funcionamento do sistema. Para realizar essa modelagem corretamente, os desenvolvedores precisam analisar o projeto em profundidade, considerando os requisitos e as necessidades da aplicação. Dessa forma, conseguem projetar um banco de dados otimizado, reduzindo riscos de desorganização e prevenindo possíveis falhas que poderiam comprometer a integridade das informações.

Um banco de dados bem modelado deve ser estruturado para facilitar a manutenção e garantir a segurança dos dados, promovendo uma base de armazenamento consistente e acessível. Além disso, essa modelagem possibilita que as operações de consulta, inserção, atualização e exclusão de dados ocorram de forma rápida e precisa, atendendo diretamente às demandas da aplicação. Dessa maneira, a modelagem de dados se torna fundamental para garantir que o sistema possa crescer e se adaptar a futuras mudanças, sem comprometer sua estabilidade e eficiência.

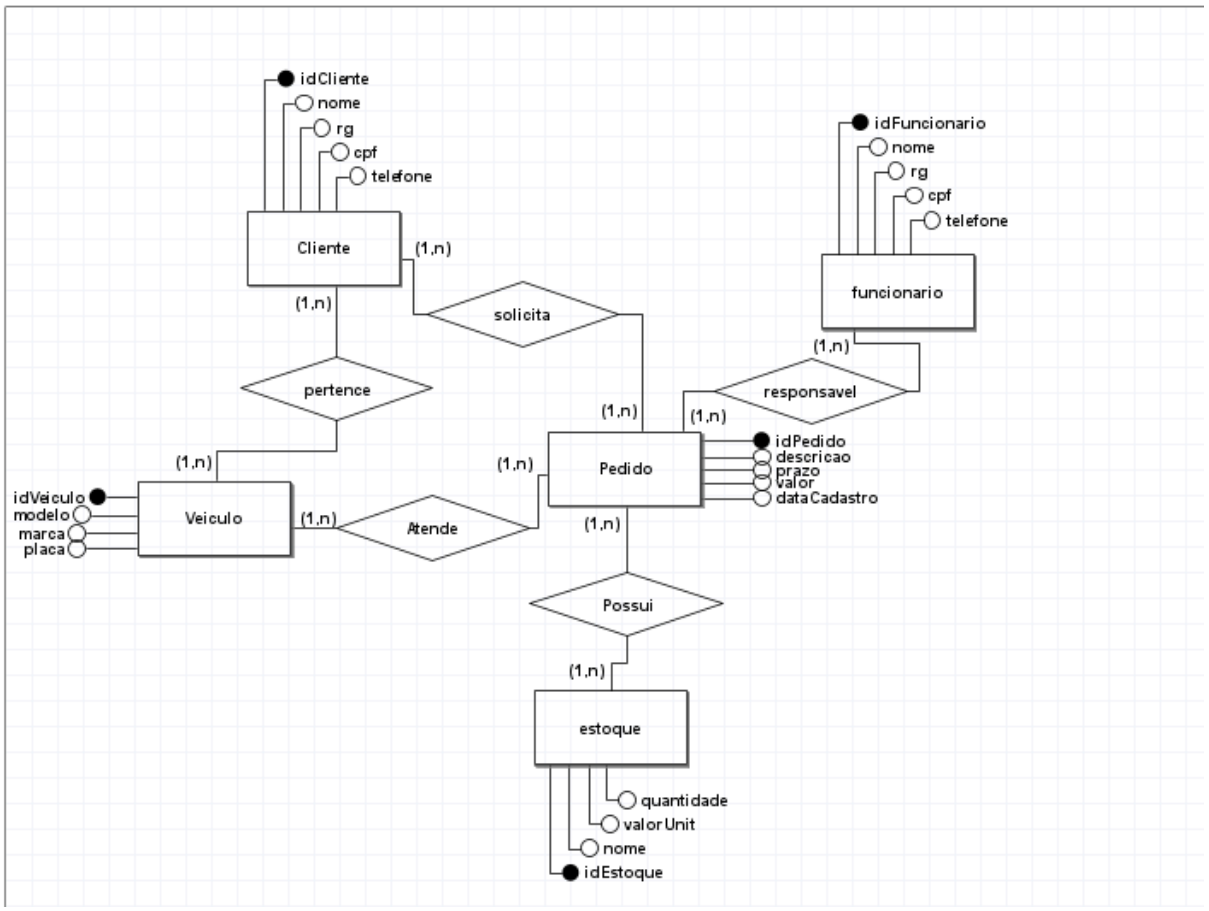
3.3.1 MODELO CONCEITUAL

A primeira fase da modelagem de dados é a criação do modelo conceitual, que tem como principal objetivo “desenhar” a estrutura do banco de dados de forma visual e abstrata. Nessa etapa, utiliza-se a representação de entidades, atributos e relacionamentos para capturar como os dados devem se comportar e se conectar. Essa representação inicial permite que

todos os integrantes do projeto compreendam a estrutura básica do banco de dados e visualizem como ele será implementado.

Desenvolver o modelo conceitual é essencial para garantir que a estrutura de dados esteja alinhada com as necessidades e requisitos do sistema. Isso facilita o entendimento do banco de dados, promovendo uma comunicação clara entre a equipe de desenvolvimento e as demais partes envolvidas. Essa etapa também guia o desenvolvedor do banco de dados nas próximas fases do projeto, como a modelagem lógica e física, estabelecendo uma base sólida para a implementação e manutenção futura dos dados.

Tomando isso como base, o modelo conceitual deste projeto visa estruturar o desenvolvimento de um sistema informatizado voltado para otimizar a gestão interna de uma empresa de funilaria automotiva. A introdução do projeto destaca a crescente adoção de tecnologias nas empresas, com foco na automação e na utilização de sistemas integrados para otimizar processos. O projeto tem como objetivo desenvolver uma solução personalizada, por meio de um sistema baseado em Python e integrado com banco de dados SQL, que permita a modernização e eficiência operacional da empresa escolhida. Além disso, será implementado um dashboard em Power BI para visualização clara e objetiva dos dados, proporcionando suporte à tomada de decisões estratégicas.



Dessa forma, o modelo conceitual define a estrutura do banco de dados a ser desenvolvido no projeto, estabelecendo as tabelas e os relacionamentos necessários para o funcionamento do sistema. A tabela "pedido" serve como o elemento central, recebendo e integrando todas as informações das demais tabelas, que são fundamentais para compor a parte "invisível" do projeto, ou seja, a camada de dados que alimenta o sistema de forma eficiente e organizada. Cada uma das tabelas secundárias possui uma relação de cardinalidade 1 para N com a tabela "pedido", garantindo que cada pedido possa se relacionar com múltiplos registros nas tabelas de clientes, peças, serviços e outras.

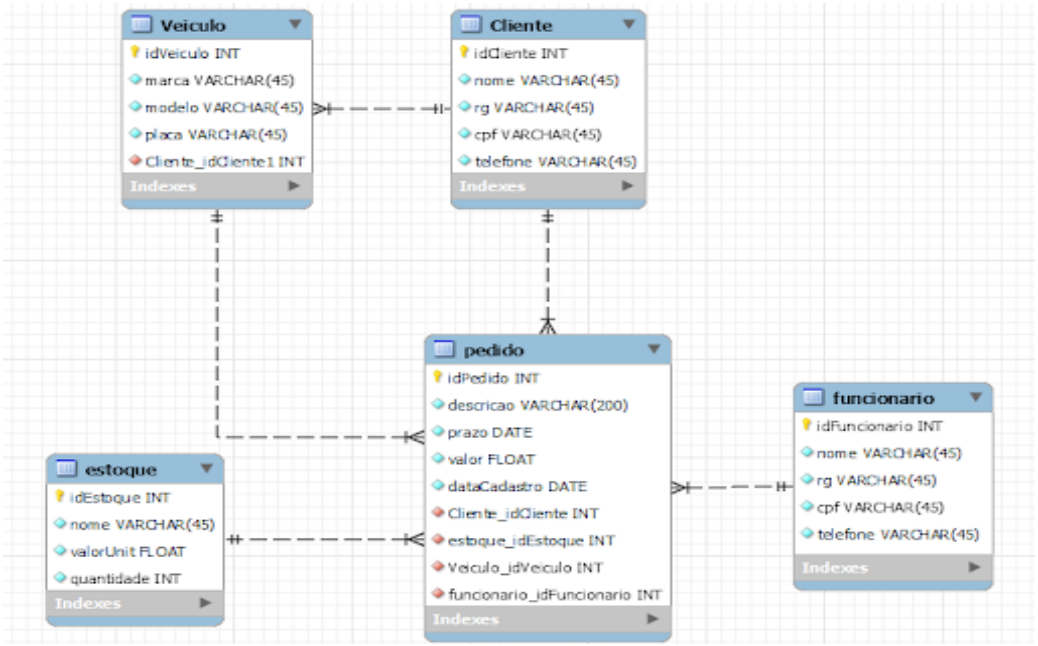
Essa organização permite que o modelo conceitual suporte a expansão e flexibilidade do sistema, mantendo uma estrutura coerente e alinhada com as necessidades do projeto. O modelo não só facilita a integração dos dados no dashboard como também garante que todas as operações, como consultas e relatórios, possam ser executadas de maneira ágil e confiável.

3.3.2 MODELO LÓGICO E FÍSICO

A segunda etapa da modelagem de dados é desenvolver o modelo lógico, que tem a função de detalhar a estrutura do banco. No modelo lógico é utilizados tabelas com os dados de uma entidade e criar as relações entre essas tabelas, definindo as chaves primárias e estrangeiras.

O modelo físico é a etapa final da modelagem, é a criação do banco de dados, utilizando as outras duas etapas como base e implementar o banco dentro de um SGBD(Sistema de Gerenciamento de Banco de Dados) como por exemplo o MySQL. As tabelas criadas no modelo físico são com base nas tabelas criadas no modelo lógico. Após concluir a parte do modelo físico, o banco de dados vai estar preparado para o sistema ser conectado.

O modelo lógico do projeto se refere à abstração dos dados e das funcionalidades do sistema sem se preocupar com a implementação física (como banco de dados, linguagens de programação ou estrutura de hardware). Ele descreve como o sistema irá funcionar em termos de fluxo de informações e processos de negócios, sem detalhes sobre como esses elementos serão fisicamente implementados.



A estrutura central do sistema é composta pela tabela “pedido”, que serve como o núcleo para consolidar as informações mais importantes e organizar os dados de forma prática. A tabela “pedido” conecta-se com diversas outras tabelas através de relações de cardinalidade 1 para N, nas quais cada registro em "pedido" pode se relacionar com múltiplos registros nas tabelas associadas (como "cliente", "veículo", "estoque" e "funcionário"), mas cada entrada nas tabelas secundárias está associada a apenas um pedido específico.

Para garantir a integridade dos dados e facilitar o processo de consulta, foram estabelecidas chaves estrangeiras em cada tabela secundária, ligando-as diretamente à tabela “pedido” com identificadores exclusivos (IDs). Por exemplo, o campo “Cliente_idCliente” na tabela “pedido” refere-se ao ID único da tabela “cliente”, enquanto “estoque_idEstoque” estabelece a conexão com a tabela “estoque”, e assim por diante com “funcionário” e “veículo”. Essas referências por meio de IDs permitem que o sistema realize consultas precisas e obtenha informações agregadas de maneira rápida e eficiente.

Essa arquitetura proporciona uma integração efetiva entre dados de clientes, peças, serviços, funcionários e prazos, centralizando as informações em um único ponto e possibilitando a criação de relatórios detalhados e o cálculo de indicadores de desempenho. Além disso, essa estrutura modular facilita futuras expansões do sistema, permitindo a adição de novas tabelas sem a necessidade de mudanças significativas na organização já estabelecida.

3.3.3 SQL

O SQL desempenha um papel central no desenvolvimento do sistema informatizado proposto para a empresa de funilaria, funcionando como a ferramenta essencial para o armazenamento, a manipulação e a consulta dos dados. Sendo uma linguagem de consulta estruturada, o SQL é utilizado para interagir com o banco de dados relacional, que armazena informações cruciais sobre os processos da empresa, como estoque de peças, ordens de serviço e dados de funcionários. A estrutura relacional do banco de dados, construída com tabelas interconectadas, garante que as informações sejam organizadas de forma eficiente, permitindo que dados como quantidades de peças, status das ordens e histórico de movimentações sejam acessados de maneira clara e lógica.

A partir da implementação do SQL, o sistema pode realizar consultas complexas que geram relatórios operacionais detalhados. Por exemplo, o SQL será usado para gerar relatórios sobre o estado do estoque, identificando peças com baixo estoque ou alertando sobre a necessidade de reposição. Ele também será utilizado para controlar o status das ordens de serviço, facilitando o acompanhamento da execução de tarefas, bem como para medir a produtividade dos funcionários ao longo do tempo. O SQL permite não apenas consultar esses dados, mas também atualizá-los, como ao registrar novas entradas de estoque ou alterar o status de uma ordem de serviço conforme ela progride.

Um exemplo do uso de SQL em nosso projeto é a consulta mais frequente realizada no sistema, que consiste em um comando "SELECT" aplicado na tabela "pedido". Esse comando é complementado por três "INNER JOINs" para integrar dados de outras tabelas relacionadas, todas conectadas por suas respectivas chaves estrangeiras. Dessa forma, a consulta traz informações consolidadas de diferentes tabelas, facilitando o acesso aos dados necessários para a análise e gerenciamento dos pedidos.

```
17 •  select c.nome Cliente, dataCadastro DatadeCadastro,
18    prazo Prazo, valor ValorFinal, v.marca MarcadoVeiculo, v.modelo ModelodoVeiculo, v.placa Placa,
19     e.nome Peca, f.nome FuncionarioResponsavel from pedido p
20    inner join cliente c on c.idCliente = p.Cliente_idCliente
21    inner join veiculo v on v.idVeiculo = p.Veiculo_idVeiculo
22    inner join funcionario f on f.idFuncionario = p.Funcionario_idFuncionario
23    inner join estoque e on e.idEstoque = p.Estoque_idEstoque
24    where c.nome like "%Carlos%";
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Cliente	DatadeCadastro	Prazo	ValorFinal	MarcadoVeiculo	ModelodoVeiculo	Placa	Peca	FuncionarioResponsavel
Carlos Silva	2023-05-06	2023-05-12	602.75	Fiat	Palio	ABC-1234	Pintura Automotiva	Ricardo Mendes
Carlos Eduardo	2022-06-29	2022-07-05	1212.15	Kia	Sportage	KIA-3456	Parafusos e Porcas	Aline Rocha

Este script SQL exemplifica a praticidade e eficiência que a linguagem proporciona ao sistema. Ele busca, em uma única consulta, diversas informações essenciais, como o nome do cliente, data de cadastro do pedido, prazo de entrega, valor do pedido, marca, modelo e placa

do veículo, além do material utilizado e o funcionário responsável pelo serviço. O filtro é realizado com base na presença do nome "Carlos" em qualquer posição do campo de nome do cliente, garantindo que apenas os registros relevantes sejam retornados.

Essa abordagem demonstra como o SQL permite consolidar dados importantes em uma única operação, otimizando o tempo de processamento e facilitando o acesso às informações de maneira integrada e ágil.

3.4 BUSINESS INTELLIGENCE

O Power BI é uma ferramenta de Business Intelligence (BI) desenvolvida pela Microsoft que transforma dados em informações visuais, interativas e fáceis de interpretar, facilitando a tomada de decisões empresariais. Destinada a analistas e profissionais que precisam entender dados de forma eficiente, a plataforma permite a criação de relatórios dinâmicos e dashboards intuitivos a partir de diversas fontes de dados, como bancos de dados SQL, arquivos Excel, serviços em nuvem e APIs de aplicações. "Com o Power BI, as empresas podem transformar seus dados em insights de maneira prática, centralizando informações e oferecendo acesso em tempo real para os tomadores de decisão" (INDICIUM, 2024).

A plataforma possui uma interface amigável de arrastar-e-soltar, que simplifica o processo de conexão, transformação e visualização dos dados, mesmo para usuários sem experiência avançada em análise de dados. Entre suas principais funcionalidades estão a capacidade de criar gráficos interativos, aplicar filtros, realizar drill-down em dados específicos e automatizar atualizações de relatórios. Power BI também oferece a possibilidade de compartilhamento e colaboração entre equipes, integrando-se ao Microsoft Teams e outros aplicativos da Microsoft para um fluxo de trabalho mais eficaz.

Além da versão de desktop, usada para desenvolvimento, o Power BI possui uma versão online (Power BI Service) que permite publicar e acessar relatórios em qualquer lugar e até mesmo visualizar dados em dispositivos móveis. Com o uso crescente de dados nas estratégias empresariais, o Power BI se tornou uma ferramenta essencial para promover a cultura data-driven, permitindo que empresas obtenham insights práticos de forma rápida e eficaz.

3.4.1 ORGANIZAÇÃO E IDENTIFICAÇÃO DAS INFORMAÇÕES

O projeto utiliza uma gama de dados que, à primeira vista, pode parecer simples, mas essa simplicidade é justamente o que permite a criação de um dashboard altamente flexível e intuitivo. O uso de dados estratégicos permite uma visualização clara e concisa das principais métricas de desempenho, essenciais para a gestão. Entre os dados principais monitorados estão:

- **Valores dos serviços**: Permite o acompanhamento financeiro dos serviços prestados.
- **Quantidade de peças e valor**: Controla o estoque oferecendo a possibilidade de se visualizar quais peças serão utilizadas para cada cliente, além do valor unitário e a quantidade de peças que se possui no estoque.
- **Visualização de clientes:** Permite a visualização do veículo que o cliente possui além da do valor já gasto do cliente na empresa.
- **Faturamento da empresa em determinado período**: Acompanha o valor de serviços realizados em determinado ano, oferecendo uma visão sobre a capacidade de produção e a eficiência operacional em diferentes períodos.

Com essas informações, o dashboard pode ser configurado para fornecer uma visão abrangente e prática de tudo que é relevante para a gestão, facilitando a identificação de áreas para otimização de processos e aumento de eficiência. Esse painel fornece insights para apoiar decisões estratégicas, como a alocação de recursos, gestão de estoque e previsão de demandas futuras. Dessa forma, a ferramenta contribui diretamente para o aumento de lucro ao garantir que todos os processos estejam alinhados com as metas de desempenho da empresa e promovam uma operação enxuta e eficaz.

3.4.2 MANIPULAÇÃO E ANÁLISE DE DADOS

A base de dados foi organizada de maneira simples e objetiva, proporcionando maior flexibilidade para o uso de fórmulas DAX e outras funções necessárias para a construção do dashboard. Essa estrutura simplificada permite que as operações sejam realizadas de forma mais eficiente, já que os dados estão preparados no próprio banco para serem inseridos diretamente no dashboard, sem a necessidade de correções ou adaptações posteriores. Esse planejamento otimiza o desempenho do sistema, reduzindo o tempo de carregamento e facilitando a manutenção e atualização dos dados.

Ao garantir que a base de dados esteja organizada e padronizada desde a origem, o projeto evita etapas redundantes de processamento e aumenta a precisão das análises. Dessa forma, a equipe de gestão pode focar na interpretação dos dados e na tomada de decisões estratégicas com maior agilidade, já que o dashboard estará sempre atualizado e acessível com informações precisas.

3.4.3 CRIAÇÃO DE MODELOS DE ANÁLISE DE DADOS

O dashboard foi desenvolvido para apoiar a tomada de decisões na gestão da empresa, proporcionando uma visão clara e objetiva dos dados mais relevantes. À esquerda, foram inseridos dois filtros: um para peças e outro para clientes. Quando um desses filtros é selecionado, todas as visualizações no dashboard se ajustam automaticamente, refletindo com precisão os dados relacionados às seleções feitas no banco de dados.

Na parte superior, há cartões que exibem informações gerais da empresa, como faturamento total e número de pedidos, que se atualizam conforme o filtro escolhido. No centro, um gráfico de linha apresenta o faturamento da empresa ao longo dos anos, fornecendo uma visão histórica do desempenho financeiro. Na parte inferior, o veículo do cliente selecionado é exibido, permitindo identificar rapidamente os serviços realizados para cada cliente. À direita, um gráfico indica o progresso em direção à meta mensal, permitindo que a gerência acompanhe o quanto falta para alcançar os objetivos de faturamento da empresa

Esse layout organizado e intuitivo facilita a análise de dados e oferece uma experiência interativa e dinâmica, tornando o dashboard uma ferramenta valiosa para a gestão estratégica e o monitoramento dos indicadores de desempenho da empresa.

3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS

A Formação para a Vida é um dos eixos do Projeto Pedagógico de Formação por Competências da UNIFEOB.

Esta parte do projeto está diretamente relacionada com a extensão universitária, ou seja, o objetivo é que seja aplicável e que tenha real utilidade para a sociedade, de um modo geral.

3.5.1 GERENCIANDO FINANÇAS

Está disponível para os estudantes no Classroom, o tema “Gerenciando Finanças”.

Nesta parte do Projeto, os estudantes deverão realizar uma síntese dos 4 (quatro) tópicos deste tema, quais sejam:

- **Tópico 1:** Introdução aos conceitos econômicos e financeiros básicos:

  Este tópico trata da importância de se entender como o dinheiro circula na sociedade, tanto entre pessoas quanto entre empresas, também abordando conceitos fundamentais de micro e macroeconomia, contabilidade e a classificação de gastos. Esses gastos podem ser classificados como custos fixos e variáveis. Um exemplo prático desses conceitos seria uma família que separa as suas despesas mensais entre essas duas categorias, como aluguel (custo fixo), internet e alimentação (custo variáveis). Ao registrar estes gastos é possível acompanhar a variação da inflação de preços em sua renda mensal.

- **Tópico 2:** Entendendo o ambiente: independência financeira, o valor da minha riqueza e o registro do dia a dia

Entende-se como independência financeira acumular riqueza suficiente para cobrir seus custos sem precisar depender de um salário. Para alcançar tal objetivo é necessário controlar custos e buscar fontes diversificadas de renda. Como exemplo prático, manter uma planilha de controle financeiro, onde você registra suas entradas e saídas de capital diariamente, possibilitando identificar áreas onde se pode cortar gastos e investir essa diferença, como um fundo de investimento ou ações.

- **Tópico 3:** Dívidas e juros compostos, opções de empréstimo e alternativas ao endividado

Este tópico trata de como as dívidas podem afetar a vida financeira e como os juros compostos podem aumentar essas dívidas de forma exponencial. Sendo os juros compostos aqueles em que os juros incidem não somente ao valor inicial da dívida, mas também sobre os juros acumulados ao longo do tempo, resultando em um crescimento absurdo da dívida.

Um exemplo claro é o uso dos cartões de crédito. Se alguém decide realizar uma compra de R$1.000,00 e escolhe parcelar a mesma em 12 vezes com uma taxa de juros de 5% ao mês, o valor final pago supera os R$1.795,00 no final do período, valor extremamente superior ao valor original. Esse efeito ocorre justamente pela característica dos juros compostos onde se acumula juros sobre juros. Compreender o funcionamento dos juros ajuda os consumidores a tomar decisões mais assertivas em relação a empréstimos e a gerenciar suas dívidas de forma mais eficaz.

- **Tópico 4:** Estabelecer metas para a realização de seus sonhos e como envolver o grupo a que você pertence para atingir seus objetivos

Estabelecer metas financeiras mais claras e envolver demais pessoas ao seu redor no processo é fundamental para alcançar objetivos de longo prazo, como realizar

um sonho familiar. Um exemplo prático seria uma família que opta por economizar para uma viagem nas férias. Todos podem participar ativamente, ajustando seus gastos diários, como reduzir despesas com compras não essenciais, ou despesas com festas e restaurantes. Esse esforço em conjunto permite que a família poupe de forma mais eficiente e ao mesmo tempo se mantenha alinhada a um objetivo em comum.

Esse processo envolve a definição de prazos específicos, o desenvolvimento de um planejamento detalhado, adaptação de hábitos consumistas ao longo do tempo. Ferramentas como planilhas ou aplicativos de planejamento financeiro podem ser muito úteis para acompanhar o progresso e ajustar as estratégias conforme necessário.

Estabelecer metas específicas, mensuráveis, alcançáveis, relevantes e com prazo definido ajuda a direcionar os esforços e manter o foco no objetivo final.

3.5.2 ESTUDANTES NA PRÁTICA

O grupo decidiu criar um banner informativo com orientações para melhorar o gerenciamento financeiro, contendo quatro dicas desenvolvidas com base na apostila fornecida. O objetivo do banner foi transmitir a mensagem de forma clara e acessível, permitindo que pessoas de todas as idades compreendam e apliquem as sugestões, independentemente do público-alvo.

O banner pode ser acessado no seguinte link: https://www.canva.com/design/DAGSXtTr5RA/hyBtmJCkHg26zGQ-ObGOmw/edit?utm_content=DAGSXtTr5RA&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton. E também pode ser visualizado nos anexos do projeto escrito: (anexo 1).

O Banner foi divido em 4 dicas principais sendo elas:

- **Entenda a Economia no Seu Dia a Dia!**
- **Conquiste sua Independência Financeira!**
- **Cuidado com as Dívidas!**
- **Realize seus sonhos com metas!**

Cada dica possui sua explicação logo abaixo de seu título, juntamente de uma imagem que complementa a informação e auxilia na interpretação.

ISSN 1983-6767

4. CONCLUSÃO

Este trabalho demonstrou a importância da informatização e automação nos processos empresariais, especialmente no contexto das empresas de funilaria automotiva, que estão cada vez mais adotando tecnologias para otimizar sua gestão e melhorar sua competitividade. O desenvolvimento de um sistema informatizado para a gestão interna de uma empresa de funilaria, utilizando Python, banco de dados SQL e um dashboard em Power BI, mostrou-se uma solução eficaz para modernizar e agilizar processos operacionais que, até então, eram predominantemente manuais. Através da integração desses sistemas, foi possível otimizar a gestão de recursos, aprimorar a tomada de decisões e contribuir diretamente para a eficiência e aumento de produtividade da organização.

O projeto também destacou a relevância do uso de ferramentas tecnológicas na busca por eficiência operacional, redução de custos e melhoria nos resultados, conforme evidenciado pelas pesquisas mencionadas. Além disso, o desenvolvimento deste sistema não se limitou apenas à aplicação de conhecimentos técnicos, mas também envolveu o desenvolvimento de habilidades essenciais, como o trabalho em equipe e a organização, fundamentais para o sucesso do projeto.

No desenvolvimento de um sistema informatizado utilizando Programação Orientada a Objetos (POO) , algumas dificuldades surgiram. Uma das principais dificuldades é a definição correta das classes e objetos. Identificar corretamente as entidades do sistema e traduzir essas entidades para classes de maneira eficiente pode ser desafiador, pois uma modelagem inadequada pode gerar um código difícil de manter e expandir. Além disso, o uso inadequado de herança e polimorfismo complica o design do sistema, tornando-o difícil de entender e de adaptar. O processo de testes e depuração de objetos também é mais complexo em sistemas orientados a objetos. Identificar e corrigir erros foi desafiador pois há muitas interações entre as classes, o que exige uma abordagem de erros pode ser desafiador quando há muitas interações entre as classes, o que exige uma abordagem estruturada para a realização de testes unitários e a aplicação de boas práticas de desenvolvimento, como injeção de dependências.

Com base nos resultados obtidos e nas lições aprendidas ao longo do desenvolvimento, podemos concluir que a implementação de soluções informatizadas, como a proposta neste trabalho, não só atende às necessidades imediatas da empresa, mas também estabelece uma base sólida para futuros avanços tecnológicos e operacionais, contribuindo para a evolução e sustentabilidade das empresas de funilaria no Brasil. O sucesso do projeto, portanto, vai além da melhoria dos processos internos, refletindo um passo significativo para a adaptação das empresas às exigências de um mercado cada vez mais digitalizado e competitivo.

REFERÊNCIAS

SOUZA, Davi. *O Mercado de Funilaria no Brasil: Oportunidades e Tecnologias*. Partsfy. Disponível em: https://www.partsfy.com.br. Acesso em: 22 set. 2024.

IBM. 78% dos líderes empresariais brasileiros investem em tecnologia para otimizar operações. *IBM Brasil*, 2023. Disponível em: https://www.ibm.com/br-pt. Acesso em: 22 set. 2024.

FOLHA VITÓRIA. A automação está presente em 78% das empresas brasileiras. *Folha Vitória*, 2024. Disponível em: https://www.folhavitoria.com.br. Acesso em: 22 set. 2024.

ROCKETSEAT. Lógica de Programação para Iniciantes em Programação. Rocketseat.com.br. Disponível em:

https://www.rocketseat.com.br/blog/artigos/post/logica-de-programacao-para-iniciantes-em-programacao. Acesso em: 04 out. 2024.

REMESSA ONLINE. Lógica de Programação: o que é, principais conceitos e como aprender!. 2023. Disponível em: https://www.remessaonline.com.br/blog/logica-de-programacao/. Acesso em: 27 out. 2024.

PRODALY. A importância da definição e validação do escopo em projetos de implantação de ERP. 2024. Disponível em: https://prodaly.com.br/erp/a-importancia-da-validacao-do-escopo-em-projetos-de-implantacao-de-erp/. Acesso em: 27 out. 2024.

ALTERDATA. O que é ERP e como funciona? Disponível em: https://www.alterdata.com.br/blog/o-que-e-erp/. Acesso em: 4 out. 2024.

TACONTRATADO. Programação Orientada a Objetos: entenda o conceito. Disponível em: https://www.tacontratado.com.br. Acesso em: 4 out. 2024.

DNC. Orientação à Objeto: entenda os principais pilares. Disponível em: https://www.escoladnc.com.br/orientacao-a-objeto. Acesso em: 04 out. 2024.

FARINELLI, Fernanda. Conceitos Básicos de Programação Orientada a Objetos. [s.l.]: [s.n.], 2007.

INDICIUM. Power BI: o que é, por que usar e quais as vantagens. Disponível em: https://academy.indicium.tech. Acesso em: 04 nov. 2024.