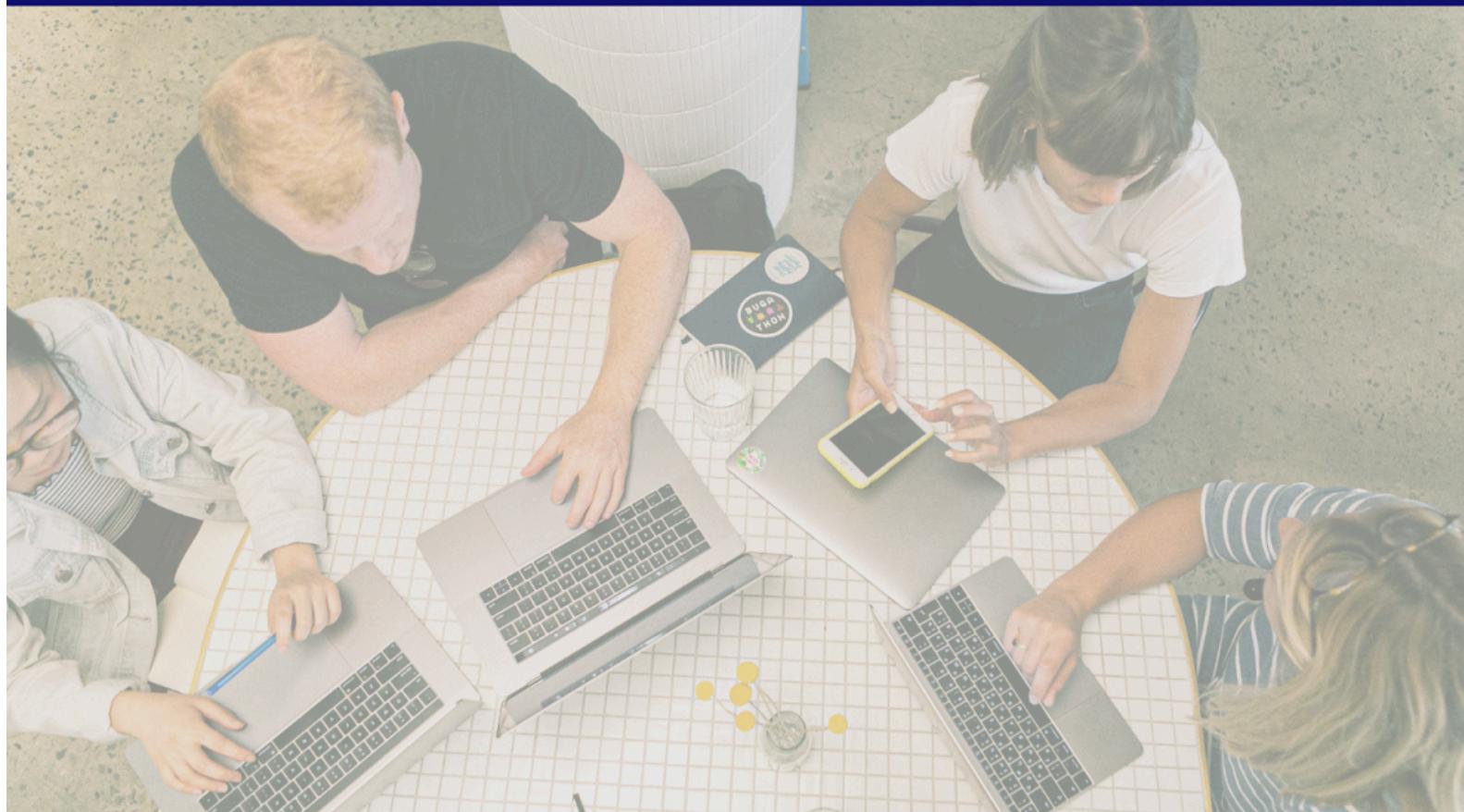


UNifeob
| ESCOLA DE NEGÓCIOS



2024

PROJETO INTEGRADO



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**

<PIZZARIA GENEROSO>

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**
<PIZZARIA GENEROSO>

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:

Erika Fernanda Sagiorato Gurgel, RA 24000411

Gabriel Gregores Petinati, RA 24000581

Giovanni Lucca Ferreira Favaro, RA 24000017

João Vitor Brito Untura de Freitas, RA 24000554

Matheus Muzi Fardim, RA 24001120

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2024

SUMÁRIO

1. INTRODUÇÃO	4
2. DESCRIÇÃO DA EMPRESA	7
3. PROJETO INTEGRADO	8
3.1 PROGRAMAÇÃO ORIENTADA A OBJETO	8
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	10
3.2 LÓGICA DE PROGRAMAÇÃO	10
3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE	11
3.2.2 DESENVOLVIMENTO DE APLICAÇÕES	11
3.2.3 IMPLEMENTAÇÃO E VALIDAÇÃO	11
3.3 MODELAGEM DE DADOS	11
3.3.1 MODELO CONCEITUAL	12
3.3.2 MODELO LÓGICO E FÍSICO	12
3.3.3 SQL	12
3.4 BUSINESS INTELLIGENCE	12
3.4.1 ORGANIZAÇÃO E IDENTIFICAÇÃO DAS INFORMAÇÕES	12
3.4.2 MANIPULAÇÃO E ANÁLISE DE DADOS	12
3.4.3 CRIAÇÃO DE MODELOS DE ANÁLISE DE DADOS	13
3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS	13
3.5.1 GERENCIANDO FINANÇAS	13
3.5.2 ESTUDANTES NA PRÁTICA	14
4. CONCLUSÃO	16
REFERÊNCIAS	17

1. INTRODUÇÃO

Neste semestre, a interdisciplinaridade entre as quatro unidades de estudo do projeto de modelagem e desenvolvimento de sistemas se mostra crucial para criar uma aplicação robusta e eficiente, sobretudo, em sistemas que abrangem várias funcionalidades integradas, como o de gerenciamento de pedidos, produtos, clientes e gastos. Este projeto aspira apresentar o desenvolvimento de um sistema de gerenciamento completo e detalhado, construído sobre a interação de diferentes módulos de desenvolvimento, cada um cumprindo funções específicas e interconectadas, com a finalidade de facilitar a administração da empresa.

O avanço da tecnologia tem proporcionado soluções inovadoras e eficientes para a gestão de diversos setores, incluindo o segmento da indústria de alimentos e redes de restaurantes. Neste contexto, o desenvolvimento de sistemas informatizados de gestão tem se mostrado uma ferramenta poderosa para otimizar processos operacionais, reduzir erros humanos e melhorar a experiência do cliente. O projeto em questão visa a criação de um sistema de gestão integrado, focado no controle de pedidos, produtos, clientes, gastos e pedidos ativos. O sistema é composto por diversos módulos interconectados, proporcionando uma solução completa para o gerenciamento de um estabelecimento comercial.

A primeira parte deste projeto concentra-se no módulo de Pedidos, onde o sistema permite o registro de novos pedidos através de uma interface simplificada. O processo de inserção de pedidos é automatizado utilizando uma abordagem inovadora com tokens, que facilitam a identificação dos produtos mencionados nas solicitações dos clientes. Ao receber o texto escrito pelo usuário, o sistema processa as palavras-chave, identifica os produtos, converte números por extenso e calcula o total do pedido com base nas quantidades e valores correspondentes. Esta abordagem reduz a probabilidade de erro e torna o processo mais ágil e preciso, o que é fundamental em ambientes de alto fluxo de trabalho.

Além disso, o projeto integra uma base de dados robusta, que é acessada por meio de classes especializadas para cada área do sistema. Através dessas classes, como BancoP (para produtos), BancoPed (para pedidos), BancoG (para gastos) e BancoC (para clientes), o sistema permite realizar operações como cadastro, atualização, exclusão e consulta de dados de forma eficiente. A utilização de uma classe abstrata, chamada BancoAbstract, proporciona uma arquitetura limpa e modular, facilitando a manutenção e a expansão do sistema. A persistência de dados no banco de dados é realizada por meio de consultas SQL otimizadas, garantindo

integridade e rapidez nas operações, fundamentais para o funcionamento de um sistema em tempo real.

A gestão de produtos, clientes e gastos é outro ponto crucial do sistema. O módulo de produtos permite não apenas o cadastro, atualização e exclusão de itens, mas também a associação de tokens ao nome de cada produto, o que melhora a precisão nas buscas realizadas pelo sistema. Já o módulo de clientes facilita o cadastro e manutenção das informações dos clientes, como nome, CPF e endereço, permitindo uma melhor personalização do atendimento. O controle de gastos é igualmente otimizado, proporcionando uma visão clara dos custos operacionais, o que é essencial para a análise financeira e o planejamento estratégico do estabelecimento. Ao integrar todos esses módulos de forma coesa, o sistema oferece uma solução inteligente e eficiente para a administração do negócio.

Por fim, o módulo de Pedidos Ativos oferece uma visão detalhada em tempo real dos pedidos em andamento. Este módulo não apenas exibe informações como a hora, mesa, status e cliente, mas também permite que os pedidos sejam finalizados de maneira simples e rápida, garantindo o fluxo contínuo de operações. Com a utilização de uma interface intuitiva, é possível monitorar o andamento dos pedidos, fazer alterações e tomar decisões rápidas, essencial em ambientes de alto desempenho. A integração entre os módulos de pedidos, produtos, clientes e gastos, juntamente com o monitoramento de pedidos ativos, torna este sistema uma solução completa e dinâmica para a gestão eficiente de estabelecimentos comerciais.

2. DESCRIÇÃO DA EMPRESA

A PIZZARIA GENEROSO, situada na charmosa cidade de Campestre, em Minas Gerais, é uma empresa que se destaca pela sua dedicação à gastronomia. Registrada como GENEROSO & GENEROSO PIZZARIA LTDA, com CNPJ 28.039.346/0001-12, a pizzaria opera como uma Microempresa (ME). Localizada na Avenida José André Avelino, nº 299, no coração do Centro, é famosa por sua vasta seleção de pratos de alta qualidade, com ênfase especial em suas pizzas.

Com um compromisso constante de inovação, a PIZZARIA GENEROSO não apenas conquista o público local, mas também busca se destacar no setor de restaurantes e similares. O cardápio é cuidadosamente elaborado, permitindo que os clientes personalizem suas pizzas com até 8 ingredientes na opção chamada GENEROSO, refletindo a paixão da pizzaria por sabores autênticos e experiências gastronômicas memoráveis.

3. PROJETO INTEGRADO

3.1 PROGRAMAÇÃO ORIENTADA A OBJETO

Embora a Simula 67 tenha lançado as bases da programação orientada a objetos, foi com a linguagem Smalltalk 80, concebida sob a visão de Alan Kay, que este paradigma de programação atingiu seu pleno desenvolvimento e popularização.

A programação orientada a objetos foi desenvolvida para superar as limitações da programação estruturada e oferecer uma abordagem mais intuitiva para o desenvolvimento de software. A ideia central é modelar elementos do mundo real como "objetos" dentro do programa, facilitando a compreensão e a organização do código. Além disso, a POO tem como uma de suas ideias principais a representação de cada elemento em termos de classes, ou objetos.

A POO é um paradigma muito popular e está presente em diversas linguagens de programação. Algumas das mais utilizadas são: Java, C++, C#, Python, Ruby, PHP, Dart, e muitas outras.

Para o projeto foi utilizado o Python como base para integrar os princípios da programação orientada a objeto. Essa linguagem foi a escolhida dentre as inúmeras nas quais suportam POO por sua versatilidade, popularidade, por oferecer diversas vantagens na hora de construir um software complexo, por sua facilidade de aprendizado e por ser a linguagem aprendida em sala de aula. Além disso, o Python se destaca pela ampla comunidade de desenvolvedores e pela rica biblioteca de ferramentas e frameworks, que facilitam a construção de sistemas de diferentes complexidades. Dessa forma, o uso de Python como linguagem base permite aplicar com eficiência os conceitos de programação orientado a objetos, tornando o processo de desenvolvimento mais intuitivo, organizado e alinhado às boas práticas do mercado.

3.1.1 CLASSES E OBJETOS

Podemos pensar em classes e objetos como uma evolução dos conceitos de tipos e variáveis em linguagens de programação mais tradicionais. Enquanto uma variável armazena um valor de um tipo específico (como um número inteiro ou uma string), um objeto é uma instância de uma classe, que define não apenas o tipo de dados que o objeto conterà, mas também as ações que podem ser realizadas com ele. Essa abordagem permite criar modelos de dados mais ricos e complexos.

As classes são a base da programação orientada a objetos. Elas fornecem um modelo para criar objetos com características e comportamentos específicos tornando o código mais organizado, reutilizável e fácil de entender.

Objetos são instâncias de classes que representam entidades do mundo real no contexto de um programa. Eles combinam dados e comportamento, tornando o código mais modular e reutilizável.

O projeto em si é um sistema para a gestão de uma pizzaria e para a construção de uma aplicação estruturada e eficiente o uso de classes e objetos têm um papel fundamental. Baseando-se neste contexto foram utilizadas classes como:

Cientes: Para realizar operações de cadastro, listagem, atualização e exclusão de clientes.

Gastos: Responsável por gerenciar os gastos no sistema. Ela usa a classe “BancoG” para interagir com o banco de dados e oferece funcionalidades para cadastrar, listar, atualizar e excluir gastos.

Produtos: Que gerenciam os produtos. Inclui funções para cadastrar, listar, atualizar e excluir produtos.

Pedidos: Que gerencia os pedidos do sistema. Este módulo é bem detalhado, com um método principal que permite ao usuário montar um pedido a partir de produtos disponíveis, calcular o total e registrar o pedido no banco de dados.

Ativos: Mostra os produtos e quantidades de cada pedido ativo, permite finalizar pedidos ativos, mudando seu status de ativa para finalizada.

Token: Que processa textos para identificar produtos em pedidos. Ele usa métodos auxiliares para padronizar os textos e identificar produtos.

Login: Este código implementa um sistema de login básico, para verificar as credenciais do usuário.

Sign: implementa um sistema simples para cadastrar usuários. Ele coleta o nome de usuário e a senha do usuário, criptografa a senha usando o algoritmo bcrypt para segurança e, em seguida, insere essas informações ao banco de dados.

BancoAbstract: Serve como uma base sólida para criar classes que interagem com um banco de dados MySQL. Essa abordagem promove a reutilização de código, a organização e a manutenção do código.

Essa estrutura baseada em classes facilita a modularização e a manutenção do sistema, permitindo que cada aspecto da operação da pizzaria (como a gestão de clientes, controle de despesas, gerenciamento de produtos e pedidos) seja tratado de forma independente, mas

integrada. A classe BancoAbstract age como o ponto de comunicação central com o banco de dados MySQL, permitindo que o sistema seja escalável e extensível para futuras funcionalidades. Com essa abordagem, o código torna-se mais organizado, reutilizável e adaptável, proporcionando uma base sólida para a expansão da aplicação de forma consistente e confiável.

3.1.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO

A definição de atributo na área de lógica de programação segundo o Dicionário Aurélio (2010) diz: **atributo** “é cada uma das propriedades que definem um objeto ou entidade”.

A visibilidade de um atributo pode ser definida como pública, privada ou protegida. Quando configurado como privado ou protegido, o atributo fica oculto, impedindo seu acesso ou uso de forma pública. Esse princípio, conhecido como encapsulamento, assegura que apenas elementos específicos possam interagir com o atributo, promovendo segurança e controle sobre os dados da classe ou do objeto.

"Método é ‘a característica que possibilita alterar a funcionalidade de um atributo’” (Aurélio, 2010).

Um método pode ser considerado uma função de um objeto, pois é por meio dele que se torna possível alterar os atributos de um objeto. Em outras palavras, o método define as ações que um objeto pode realizar.

Os atributos e métodos do projeto foram implementados conforme as necessidades identificadas para o sistema de gestão da pizzaria. Cada método e atributo foi cuidadosamente associado à classe correspondente, garantindo que suas funcionalidades estejam alinhadas com os requisitos específicos do negócio. Dessa forma:

Clientes: Representa os clientes da pizzaria, com os seguintes atributos: nome, endereço e CPF. E os métodos, main, cadastrarCliente, listarCliente, atualizarCliente e excluirCliente.

Gastos: Modelo de custos operacionais, incluindo as despesas da pizzaria. Possui os atributos: nome, valor e data. A classe inclui os seguintes métodos, main, cadastrarGastos, listarGastos, atualizarGastos e excluirGastos.

Produtos: Representa os itens do cardápio, como pizzas e bebidas, com os atributos: nome, valor e descrição. E os seguintes métodos, main, cadastrarProdutos, listarProdutos, atualizarProdutos e excluirProdutos.

Pedidos: Registrar cada pedido realizados com os métodos, finalizar_venda, buscar_produto e forma_pagamento.

Ativos: Especificamente para pedidos que estão em andamento, com atributos que refletem o status do pedido, como ID, hora, mesa, status e cliente e possui um método para atualizar o status do pedido de “Ativo” para “Finalizado”.

Token: Processa textos para identificar produtos em pedidos, possui métodos como: processar_pedido(texto) Processa uma string para identificar produtos e quantidades; singularize(palavra) Converte palavras para o singular; remove_acentos(palavra) Remove acentos de uma string para facilitar a correspondência.

Sign: Insere um novo user e senha no banco criando assim um novo usuário do sistema, não possui atributos .

Login: Tem a função de comparar o user e senha inserida, verificando se é cadastrada no banco de dados, caso seja o acesso ao sistema é liberado. Possui os atributos username e senha.

BancoAbstract: Ela define as operações básicas de CRUD (Create, Read, Update, Delete) que são comuns a muitas classes que manipulam dados em um banco de dados.

Encapsulamento é um dos pilares da programação orientada a objetos (POO) e refere-se à prática de esconder os detalhes internos de uma classe e expor apenas as informações e funcionalidades necessárias para o uso da classe. Esse conceito tem como principal objetivo proteger o estado interno de um objeto, permitindo que ele seja manipulado de maneira controlada, através de métodos definidos, ao invés de permitir acesso direto a seus dados.

Ao aplicar o encapsulamento, os desenvolvedores podem definir limites claros sobre o que pode ou não ser acessado fora da classe. Isso proporciona maior segurança, pois evita que o estado do objeto seja alterado de forma inesperada ou incorreta. A ideia é que, ao invés de acessar diretamente os atributos de uma classe, o exterior da classe interaja com ela apenas por meio de métodos públicos (geralmente chamados de "getters" e "setters"), que permitem a leitura e modificação dos dados de forma controlada.

A herança é uma das quatro formas de relacionamento que podem ser fundamentadas entre classes. Esse tipo de relação permite que uma classe filha (ou subclasse) herde os atributos e métodos públicos ou protegidos de uma classe pai (ou superclasse), de forma direta. Contudo, atributos e métodos privados não são transferidos da classe pai para a classe filha durante a herança.

No sistema desenvolvido vemos o processo de herança na classe “BancoAbstract” (classe mãe) que atua como base para as classes BancoP, BancoPed, BancoG, BancoC e BancoU (classes filhas). Ela define os métodos básicos de interação com o banco de dados, como inserir,

selecionar, atualizar e excluir registros. Ao herdar da classe BancoAbstract, as classes filhas podem se concentrar na lógica específica de cada entidade (cliente, gasto, pedido, produto, usuário), reutilizando o código da classe base para as operações comuns.

Polimorfismo é um conceito central de programação orientada a objetos (POO) que permite que objetos de diferentes classes, derivados de uma mesma superclasse, sejam tratados como se fossem do mesmo tipo, desde que compartilhem uma estrutura ou comportamento comum. Isso significa que o mesmo método ou operação pode funcionar de maneiras específicas, dependendo do objeto que o executa, utilizando uma referência ao tipo da superclasse. Dessa forma, o polimorfismo traz flexibilidade ao código, permitindo que ele seja mais extensível e modular, já que possibilita a criação de novas funcionalidades sem alterar o código existente.

No sistema o polimorfismo acontece quando se tem o uso de diversos métodos como: Cadastrar, Listar, Atualizar e Excluir nas diferentes classes como, Clientes, Pedidos, Gastos e Produtos. Por mais que o comportamento, o nome e a estrutura dos métodos sejam as mesmas, cada método exerce a função em sua classe específica.

3.1.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS

Método estático é aquele que não depende de instâncias ou objetos para ser chamado e nem pode ser herdado. Este método geralmente é indicado para funcionalidades que não precisam acessar atributos ou métodos de instâncias.

Quando um método pode ser utilizado para interagir com objetos e ser acessado de fora da classe são denominados Métodos Públicos.

No sistema desenvolvido para gerir uma pizzaria, alguns métodos públicos como cadastrar (clientes, produtos, pedidos, gastos), atualizar (clientes, produtos, pedidos, gastos), listar (clientes, produtos, pedidos, gastos) e excluir (clientes, produtos, pedidos, gastos) cada um em sua determinada classe, faz com que seja possível a interação de outras partes do código com os objetos dessas classes.

Os métodos privados só podem ser acessados dentro da sua classe e não são visíveis dentro de seus objetos.

No sistema desenvolvido foram utilizados os métodos da classe Token como privados, por serem métodos de dicionários, não é necessário ser mostrado dentro do objeto.

Foi utilizado também na classe BancoAbstract o método abstrato, que nada mais é do que uma SuperClasse destinada a outras classes. Uma classe abstrata não pode ser instanciada.

Para importar ABC e abstractmethod utilizamos `from abc import ABC, abstractmethod` do método módulo. O decorador responsável por indicar métodos abstratos em uma classe é denominado abstractmethod, enquanto ABC é uma classe que serve para criar classes abstratas.

3.2 LÓGICA DE PROGRAMAÇÃO

A lógica de programação é a principal ferramenta para o desenvolvimento inicial de um programa e profissional de tecnologia é a partir dela que você planeja e organiza o comportamento do seu código, usando uma continuação de passos um seguido do outro para alcançar determinado objetivo. A linguagem estudada por nós nesse semestre foi o Python, uma das linguagens mais utilizadas dentro do mercado de programação.

Dentre os inúmeros diferenciais do Python, destacam-se a construção de algoritmos e a criação de sequências lógicas de variáveis, que determinam a execução das tarefas. Além disso, o entendimento dos tipos de dados, como números inteiros, textos, entre outros, é fundamental, pois esses tipos são usados para armazenar e manipular dados durante a execução de um programa.

Ademais, a apresentação de estruturas que possibilitam a tomadas de decisões ao longo do código, com base em condições específicas promovem a flexibilidade dentro dos programas, denominado “condicionais”.

Outro ponto a se destacar é os operadores lógicos que sua função é combinar ou inverter condições, voltando (verdadeiros ou falsos) e os operadores de comparação que sua função é comparar dois valores ou mais, voltando como resposta se são true ou false

3.2.1 CONCEITOS FUNDAMENTAIS DO DESENVOLVIMENTO DE SOFTWARE

No decorrer do projeto e das metas que foram estabelecidas, o grupo fez um planejamento bem detalhado da estrutura do programa, pensando em como o sistema ia ser fácil de gerir, de entregar e também de escalar, tanto pros clientes quanto para funcionários que irão utilizar a aplicação, sendo uma das principais preocupações a experiência do usuário.

Para que isso fosse possível, durante as semanas trabalhadas os códigos foram separados por módulos, contendo cada parte uma realização que iria gerar, conforme o estabelecido no pedido referente ao cliente.

Além disso, a modelagem do código também facilita a manutenção e evolução futura da aplicação. Isso significa que caso ocorresse em adicionar novas funcionalidades, como novas formas de pagamento ou promoções especiais, esses ajustes poderiam ser feitos de forma independente, sem afetar outras partes do sistema.

Por fim, para o desenvolvimento ágil, esta abordagem garante que o sistema possa ser dimensionado à medida que as necessidades do negócio aumentam, sem comprometer o desempenho ou a estabilidade do programa.

3.2.2 DESENVOLVIMENTO DE APLICAÇÕES

O código do nosso projeto apresenta e exemplifica a implementação das funções e módulos que compõem a aplicação final, com um foco claro na organização e modularização do código, o que facilita a manutenção e evolução do software. A estrutura modular é refletida pela organização do código em diferentes módulos como Cliente, Gastos, Pedidos_Ativos, Pedidos_Auto, Produtos, Login e Sign. cada módulo contém uma etapa e funcionalidades específicas.

O uso de funções principais como main() em cada etapa permite a organização lógica do código, separando as responsabilidades de cada componente e tornando o código mais simples possível. Por exemplo, o módulo Login é responsável pela autenticação do usuário, enquanto o módulo Sign gerencia o cadastro de novos usuários. Esses 2 módulos são chamados dentro de um loop principal, que gera a navegação entre as diferentes opções do sistema, causando uma experiência interativa para pessoa que está controlando. Essa estrutura modular permite que cada funcionalidade seja testada e utilizada de forma independente, sem interferir no restante do código do programa.

Ademais, a lógica de navegação é controlada por um menu interativo, onde o usuário pode escolher entre opções como “pedidos”, “pedidos ativos”, “produtos”, “clientes e gastos”, ficando o programa em loop contínuo até que o usuário decida sair.

Em termos de manutenção e evolução, a modularização e a separação clara de responsabilidades tornam o código fácil de entender e modificar. Se, por exemplo, for necessário adicionar novas funcionalidades, como a gestão de promoções ou descontos, isso poderia ser feito criando um novo módulo e integrando-o ao sistema sem grandes impactos no código existente. Dessa forma, a aplicação se torna escalável e adaptável às necessidades futuras, mantendo sua integridade e estabilidade.

3.2.3 IMPLEMENTAÇÃO E VALIDAÇÃO :

Durante o estágio de implementação e validação do projeto, reunimos todos os elementos criados para assegurar que o aplicativo operasse de maneira integrada e eficiente. Durante o processo de desenvolvimento, todas as funções de gestão e processamento dos pedidos foram incorporadas através dos métodos principais coordenando assim o funcionamento contínuo do sistema em um loop interminável.

Nas seções principais dessas funções foram desenvolvidos métodos específicos para atividades como registro de dados novos ou atualização e inserção de registros existentes em adição a outras operações críticas do sistema informatizado. A organização desse conjunto de procedimentos facilitou a interação entre todas as partes do programa de forma eficiente assegurando uma execução sem intercorrências ou falhas significativas.

Além disso, foi concedida uma atenção especial à gestão de erros durante os ensaios realizados. Se surgisse algum contratempo ou falha não prevista o sistema foi ajustado para detectar e comunicar o problema de modo claro assegurando que o progresso do projeto permanecesse em conformidade com os requisitos e operar adequadamente.

Essas práticas de integração e verificação asseguraram que todas as demandas do projeto fossem cumpridas com sucesso e culminaram em um sistema resiliente e eficaz pronto para ser entregue ao estabelecimento.

3.3 MODELAGEM DE DADOS

A modelagem de dados é uma etapa fundamental no desenvolvimento de sistemas, pois define a estrutura do banco de dados que suportará as operações da aplicação. Para garantir que o

sistema da Pizzaria Generoso fosse eficiente, foi desenvolvido um modelo de banco de dados desde a concepção até a implementação para auxiliar na construção do nosso sistema.

3.3.1 MODELO CONCEITUAL

A primeira fase da modelagem de dados foi a criação do Modelo Conceitual, que visou entender as entidades do sistema e suas interrelações. Neste projeto, o Modelo Conceitual foi desenvolvido utilizando um Diagrama de Entidade-Relacionamento(DER). O DER representou de forma visual as entidades essenciais para o funcionamento do sistema, como Clientes, Pedidos, Produtos, Gastos, Ativos e Banco, entre outras.

Por exemplo, a entidade Cliente tem atributos como nome, endereço e CPF, e se relaciona com a entidade Pedidos, já que cada cliente pode realizar vários pedidos. A entidade Pedido se relaciona com Produtos, representando os itens pedidos, e com Gastos, que agregam custos relacionados aos pedidos. O Banco será uma entidade de apoio, representando a conexão do sistema com o banco de dados, onde todas as transações serão armazenadas e consultadas.

O modelo conceitual foi documentado detalhadamente, especificando cada entidade e seus atributos, além do relacionamentos entre elas. A análise e o desenho do DER garantiram que a estrutura do banco de dados fosse bem definida, servindo como base para as demais etapas do projeto.

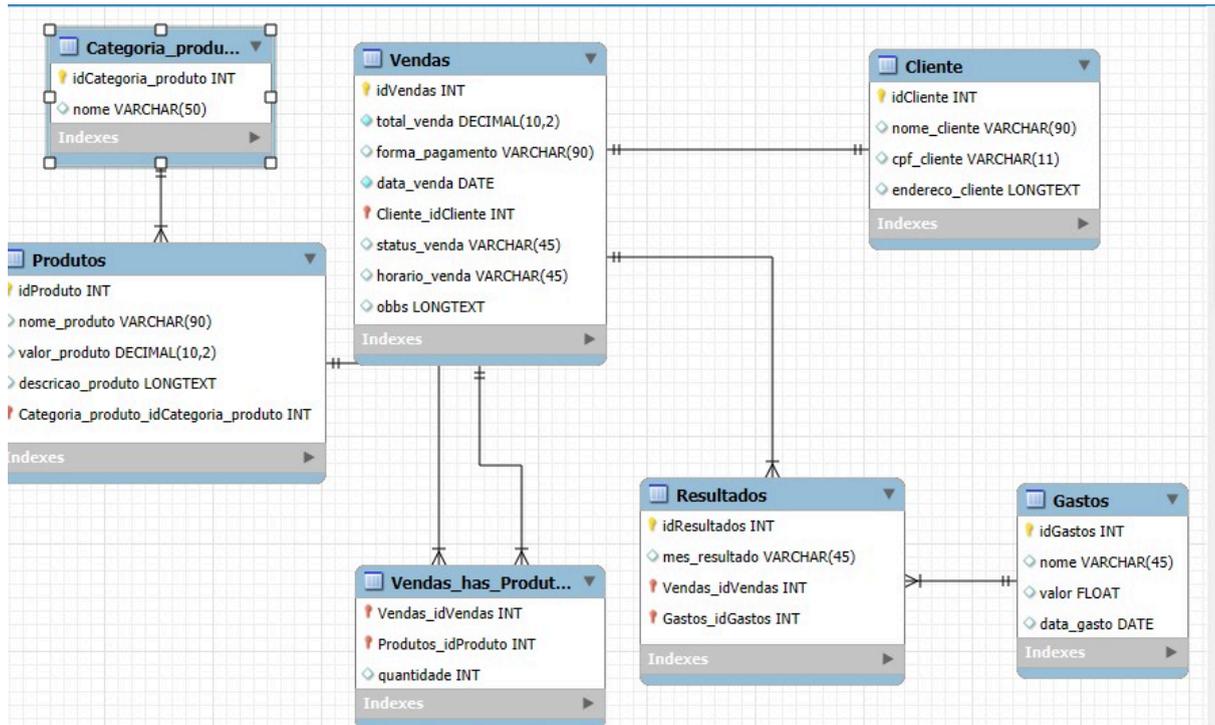
3.3.2 MODELO LÓGICO E FÍSICO

O Modelo Lógico é a transcrição do Modelo Conceitual para uma estrutura mais detalhada, mas ainda independente do SGBD (Sistema de Gerenciamento de Banco de Dados). Nessa fase, foram especificados os tipos de dados para cada atributo e os relacionamentos entre as tabelas. A normalização dos dados foi considerada, já que visamos eliminar as redundâncias e garantir a integridade dos dados.

Para o sistema da Pizzaria Generoso, foram criadas tabelas para cada entidade identificada no modelo conceitual, como clientes, pedidos, produtos, gastos e ativos. Cada tabela tinha suas colunas definidas com tipos de dados apropriados (por exemplo, VARCHAR para nomes, DATE para datas, DECIMAL para valores monetários).

Já o Modelo Físico é a implementação do Modelo Lógico em um sistema de banco de dados real, neste caso, utilizando o SGBD MySQL. Isso envolve a criação de scripts SQL para

gerar as tabelas no banco de dados, implementando índices para otimizar a performance e estabelecer restrições de integridade para garantir a consistência dos dados. Nesta fase, as tabelas foram efetivamente criadas no banco de dados MySQL, com comandos específicos para gerar as estruturas e estabelecer as relações entre as entidades.



3.3.3 SQL

- **Insert:** Para inserir novos dados nas tabelas. Por exemplo, inserir novos clientes ou novos pedidos no sistema.

```
INSERT INTO Clientes (nome, endereco, cpf) VALUES ('João Silva', 'Rua A, 123', '123.456.789-00');
```

- **Update:** Para atualizar registros existentes, como modificar os dados de um cliente ou atualizar o status de um pedido.

```
UPDATE Pedidos SET status = 'Concluído' WHERE id_pedido = 1;
```

- **Delete:** Para excluir registros, como quando um cliente cancela um pedido ou quando uma despesa é removida do sistema.

```
DELETE FROM Gastos WHERE id_gasto = 5;
```

- **Select:** Para realizar consultas no banco de dados, recuperando informações específicas. Por exemplo, listar todos os pedidos de um cliente ou consultar os produtos disponíveis no menu da pizzaria.

```
SELECT * FROM Produtos WHERE categoria = 'Pizza';
```

Esses comandos foram utilizados ao longo do desenvolvimento para preencher o banco de dados com informações reais, realizar testes e criar consultas necessárias para o funcionamento do sistema. O uso adequado de comandos SQL garantiu que a aplicação interagisse corretamente com o banco de dados, permitindo a inserção, atualização, exclusão e consulta dos dados de forma eficiente e segura.

3.4 BUSINESS INTELLIGENCE

Através do DashBoard, é possível obter uma vista clara e visual das principais métricas da receita e despesas mensais da pizzaria, por meio de diagramas destacando vendas de cada produto, lucro total, margem de lucro e despesas mensais, é possível analisar as informações da performance da empresa GENEROSO.

3.4.1 ORGANIZAÇÃO E IDENTIFICAÇÃO DAS INFORMAÇÕES

Foi criado um dashboard no Power BI que recebe dados e os organiza de forma sistemática e visualmente intuitiva as informações essenciais para avaliar o desempenho dos produtos e a evolução dos volumes de transações ao longo do tempo.

Os elementos do dashboard exercem funções para fornecer informações sobre o sistema: o gráfico de barras de "Faturamento de cada produto" permite identificar quais produtos geram mais receita, facilitando a identificar itens de maior e menor demanda; o gráfico de linhas "Faturamento por mês" permite analisar a variação nas receitas ao longo do ano, destacando picos e quedas de vendas mensais; o gráfico de pizza "Total por produto" detalha a distribuição

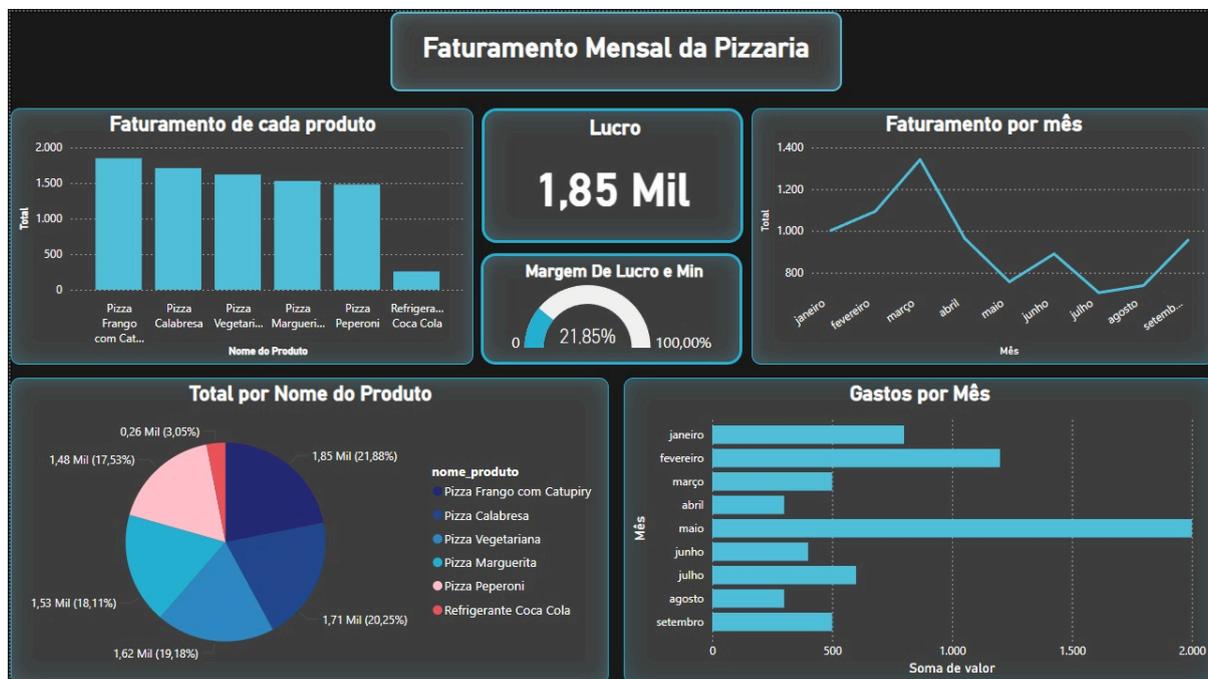
percentual de faturamento entre os produtos demonstrando possíveis tendências, e o gráfico de barras "Gastos por mês" permite monitorar os custos operacionais mensais. A "Margem de Lucro" mostra de maneira objetiva a lucratividade geral. A organização dessas informações foi planejada para que gestores possam rapidamente obter uma visão estratégica dos principais indicadores de desempenho.

3.4.2 MANIPULAÇÃO E ANÁLISE DE DADOS

Para construir este dashboard foram utilizados dados de um banco de dados MySQL, que é constantemente atualizado para permitir análises em tempo real. Esses dados são manipulados e organizados para criar indicadores específicos, como o faturamento total por produto e a evolução das despesas mensais, que permitem uma visão detalhada dos resultados.

Além disso, o cálculo da margem de lucro, graças a uma medição estabelecida no PowerBI, permite analisar a rentabilidade da operação, proporcionando um quadro completo das vendas e custos por período. A estrutura de dados facilita a comparação entre produtos e períodos, promove análises completas e informadas para apoiar decisões estratégicas e identificar oportunidades de melhorias operacionais ou ajustes nas políticas de vendas e custos.

3.4.3 CRIAÇÃO DE MODELOS DE ANÁLISE DE DADOS



Foi criado um modelo de análise de dados dinâmico e visualmente acessível no Power BI, que integra os dados gerados pelo sistema de forma consolidada e interativa. Esse modelo permite a visualização de tendências e sazonalidades, como observado nos gráficos de faturamento e de gastos mensais, ajudando a identificar padrões de comportamento nos

diferentes períodos do ano. O cálculo automático da margem de lucro permite uma análise simplificada da rentabilidade, com uma exibição clara que facilita o acompanhamento do desempenho financeiro.

A integração com a base de dados MySQL permite que os dados sejam atualizados regularmente, garantindo que o dashboard reflète sempre a situação mais recente. A utilização de gráficos diversos, como barras, linhas e pizzas, foi essencial para representar cada métrica de forma adequada, promovendo um entendimento rápido e eficiente das variações de desempenho. Este modelo visual facilita o acompanhamento dos principais indicadores e ajustes estratégicos em tempo real, contribuindo assim para uma gestão mais precisa e informada.

O dashboard abrangente promove uma visão estratégica dos dados do sistema, integrando-se ao banco de dados MySQL para atualizações automáticas e permite a tomada de decisões informadas com base em informações sempre atualizadas.

3.5 CONTEÚDO DA FORMAÇÃO PARA A VIDA: GERENCIANDO FINANÇAS

3.5.1 GERENCIANDO FINANÇAS

A compreensão dos conceitos econômicos e financeiros é essencial para o desenvolvimento de uma sociedade informada e capaz de tomar decisões conscientes. A economia se divide em microeconomia, que analisa o comportamento de indivíduos e empresas, e macroeconomia, que estuda fenômenos agregados como crescimento e inflação. Já as finanças focam no gerenciamento de ativos e passivos, essencial para garantir a saúde financeira de indivíduos e empresas.

O conhecimento econômico e financeiro é crucial não apenas para a vida pessoal e empresarial, mas também para a formulação de políticas públicas eficazes e o desenvolvimento sustentável. Cidadãos informados podem participar ativamente do debate econômico e contribuir para a criação de políticas que promovam o bem-estar social. Em um mundo globalizado, essa compreensão se torna vital para enfrentar desafios como crises financeiras e desigualdade econômica.

Em resumo, a educação em economia e finanças capacita indivíduos a gerenciar suas finanças e a entender o cenário econômico, promovendo um futuro mais equilibrado e justo. Investir nesse conhecimento é fundamental para o progresso coletivo e a construção de uma sociedade mais consciente e participativa.

A independência financeira é um objetivo almejado por muitos, pois representa a capacidade de viver sem depender de um emprego ou de outras fontes de renda ativas. Esse conceito está intrinsecamente ligado ao entendimento do valor da riqueza, que vai além de acumular bens materiais, englobando também a segurança financeira e a liberdade de escolhas.

Para alcançar a independência financeira, é crucial registrar e monitorar os gastos diários. Esse controle permite identificar padrões de consumo, ajustar hábitos e priorizar investimentos que contribuam para o aumento da riqueza. O registro do dia a dia, seja por meio de planilhas, aplicativos ou anotações, proporciona uma visão clara da saúde financeira e ajuda a estabelecer metas realistas.

Além disso, o valor da riqueza deve ser compreendido em um contexto mais amplo, considerando não apenas a quantia de dinheiro, mas também a qualidade de vida que ela pode proporcionar. A educação financeira desempenha um papel vital nesse processo, capacitando os indivíduos a tomar decisões informadas sobre investimentos, economia e planejamento para o futuro.

Em síntese, entender o ambiente da independência financeira envolve reconhecer o verdadeiro valor da riqueza e a importância do registro das atividades diárias. Essa abordagem permite construir uma base sólida para a liberdade financeira, promovendo uma vida mais equilibrada e com menos preocupações. Investir tempo e esforço nesse aprendizado é fundamental para alcançar os objetivos financeiros desejados.

O manejo de dívidas e a compreensão dos juros compostos são essenciais para a saúde financeira. Dívidas, se não geridas corretamente, podem levar a um ciclo vicioso de endividamento, dificultando a recuperação financeira. Os juros compostos, que são juros calculados sobre o valor total da dívida, incluindo os juros acumulados, podem agravar essa situação, tornando o pagamento da dívida muito mais caro ao longo do tempo.

Para quem está endividado, é fundamental explorar opções de empréstimo com cautela. Empréstimos pessoais, crédito consignado e cartões de crédito são algumas das alternativas disponíveis, mas cada uma apresenta diferentes taxas de juros e condições. Antes de optar por um empréstimo, é crucial avaliar a capacidade de pagamento e buscar opções com juros mais baixos.

Além disso, existem alternativas para evitar ou minimizar o endividamento. A educação financeira é uma ferramenta poderosa, permitindo que os indivíduos aprendam a gerenciar seu orçamento, cortem gastos desnecessários e priorizem o pagamento de dívidas com juros altos. Programas de renegociação de dívidas e consultas a serviços de proteção ao crédito também podem ajudar a aliviar a pressão financeira.

Em resumo, a gestão de dívidas e a compreensão dos juros compostos são fundamentais para evitar o endividamento excessivo. Explorar opções de empréstimo de forma consciente e considerar alternativas viáveis são passos essenciais para recuperar a saúde financeira. Com educação financeira e planejamento adequado, é possível reverter a situação e construir uma base financeira sólida.

Estabelecer metas claras é essencial para transformar sonhos em realidade. Ao definir objetivos específicos, conseguimos criar um plano de ação que guia nossos esforços. A realização de metas individuais pode ser significativamente potencializada ao envolver o grupo a que pertencemos, seja uma família, amigos ou colegas de trabalho.

Ao compartilhar nossas aspirações e desafios com outras pessoas, podemos criar um ambiente de apoio e responsabilidade mútua. O envolvimento do grupo proporciona motivação, pois cada membro se sente encorajado a trabalhar em direção a seus objetivos, sabendo que conta com o suporte dos outros. Essa dinâmica fortalece os laços e torna o processo mais colaborativo.

Além disso, a troca de ideias e experiências enriquece a jornada. Discutir estratégias e desafios em grupo permite que cada um ofereça perspectivas diferentes e soluções criativas, aumentando a probabilidade de sucesso coletivo. A colaboração torna a caminhada mais agradável e gera um senso de comunidade e propósito.

Em resumo, estabelecer metas para a realização de sonhos é um passo crucial. Ao envolver o grupo ao nosso redor, potencializamos as chances de alcançar esses objetivos. A união e o apoio mútuo não apenas tornam a busca mais leve, mas também criam uma experiência compartilhada e enriquecedora.

3.5.2 ESTUDANTES NA PRÁTICA

A equipe optou por fazer um banner para realizar a unidade de formação para a vida.

O banner foi desenvolvido através dos seguintes temas: 1-Introdução aos conceitos econômicos e financeiros básicos. 2-Entendendo o ambiente: independência financeira, o valor da minha riqueza e o registro do dia a dia. 3-Dívidas e juros compostos, opções de empréstimos

e alternativas ao endividado. 4-Estabelecer metas para a realização de seus sonhos e como envolver o grupo a que você pertence para atingir seus objetivos.

Com a criação do tópico 3.5.1 do projeto a equipe passou a ter mais conhecimento sobre os temas acima e assim foi criado o banner que engloba todos esses temas e traz ao público “6 Passo para o sucesso e estabilidade financeira”.

6

Passos para o Sucesso e Estabilidade Financeira

1 **REGISTRO DIÁRIO: MEMÓRIAS QUE TRANSFORMAM**

Documentar o dia a dia ajuda a identificar padrões, metas e áreas de crescimento, além de valorizar pequenas ações que tornam a vida mais rica e significativa.



2 **RESERVA DE EMERGÊNCIA: UM ESCUDO CONTRA DÍVIDAS**

Construir uma reserva financeira protege contra imprevistos e evita o ciclo de endividamento, promovendo segurança e estabilidade a longo prazo.



3 **SOLUÇÕES INTELIGENTES PARA SAIR DAS DÍVIDAS**

Portabilidade de crédito e consolidação de dívidas são estratégias para renegociar condições e organizar finanças, tornando os pagamentos mais acessíveis e fáceis de gerenciar.



4 **METAS CLARAS: O PRIMEIRO PASSO PARA REALIZAR SONHOS**

Definir objetivos específicos e dividi-los em etapas alcançáveis facilita o caminho, tornando o sonho uma realidade concreta.



5 **GRATIDÃO: VALORIZE CADA PASSO DA JORNADA**

Reconheça as pequenas conquistas do dia a dia. A prática da gratidão fortalece a motivação e revela a riqueza presente em cada momento.



6 **TRABALHO EM EQUIPE: A FORÇA DO SUCESSO COMPARTILHADO**

Alinhe esforços com sua equipe, família ou amigos. A colaboração fortalece objetivos e cria laços enquanto transforma sonhos em conquistas.



4. CONCLUSÃO

O projeto de criação de um sistema de gestão integrado, focado no controle de pedidos, produtos, clientes, gastos e pedidos ativos, representa uma solução robusta para otimizar a administração de estabelecimentos comerciais. A integração entre os diferentes módulos abordados nas unidades de estudo, permite um fluxo de informações eficiente, reduzindo erros operacionais e proporcionando maior agilidade na tomada de decisões. Ao centralizar o gerenciamento em uma única plataforma, o sistema oferece uma visão holística do negócio, o que é crucial para o funcionamento do projeto.

Entretanto, o desenvolvimento de um sistema desse porte não está isento de desafios. A integração dos módulos requer um design arquitetônico sólido e a utilização de tecnologias que garantam a escalabilidade e a segurança dos dados. Além disso, a constante comunicação entre as diferentes áreas do sistema pode demandar tempo e recursos para garantir a harmonização dos fluxos de trabalho. A personalização do sistema de acordo com as necessidades específicas de cada estabelecimento também se apresenta como uma tarefa complexa, uma vez que diferentes tipos de negócio exigem funcionalidades distintas, o que pode exigir ajustes contínuos durante a fase de implementação.

Por fim, a implementação de um sistema de gestão integrado exige, além da tecnologia, um processo de adaptação por parte dos usuários finais, que devem ser treinados para maximizar o uso das funcionalidades oferecidas pela plataforma. O sucesso do projeto depende não apenas da eficácia técnica, mas também da aceitação e adaptação dos colaboradores ao novo sistema. Portanto, um acompanhamento contínuo e o suporte adequado são essenciais para garantir a sustentabilidade e a eficiência do sistema ao longo do tempo, assegurando que ele possa se adaptar às mudanças e evoluções do mercado.

REFERÊNCIAS

Essa parte está reservada para as referências, as quais devem estar metodologicamente discriminadas em ordem alfabética e corresponder às citações realizadas ao longo dos textos. A utilização da metodologia científica é obrigatória e deve ser utilizado o Manual UNIFEOB para Trabalhos Acadêmicos ou as Normas da ABNT.

SILVA, Fabrício M.; LEITE, Márcia CD; OLIVEIRA, Diego B. Paradigmas de programação . Porto Alegre: SAGAH, 2019. E-book. págs.60 e 88. ISBN 9788533500426. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9788533500426/>. Acesso em: 16 out. 2024.

João Henrique: Programação orientada a objetos e programação estruturada. 2023. Disponível em: <https://www.alura.com.br/artigos/poo-programacao-orientada-a-objetos?>. Acesso em: 21 out 2024

MANZANO, José Augusto Navarro G.; OLIVEIRA, Jayr Figueiredo de. Algoritmos - Lógica para Desenvolvimento de Programação de Computadores. 29ª edição. Rio de Janeiro: Érica, 2019. E-book. págs.296 e 315. ISBN 9788536531472. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9788536531472/>. Acesso em: 06 nov. 2024.

FERREIRA, A. B. H. Dicionário Aurélio da língua portuguesa. 5. ed. Curitiba: Positivo, 2010.

AGUILAR, Luis J. Fundamentos de programação. 3ª edição. Porto Alegre: AMGH, 2008. E-book. pág.563. ISBN 9788580550146. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9788580550146/>. Acesso em: 06 nov. 2024.

Jr. Couto, Erico: Conceitos de Programação Orientada a Objetos em Python. Jan 21,2024. Disponível em: <https://erico-coutojr.medium.com/conceitos-de-programação-orientada-a-objetos-em-python>. Acesso em 08 nov 2024.