UNIFEOB
ESCOLA DE NEGÓCIOS

2022

PROJETO DE CONSULTORIA EMPRESARIAL

ISSN 1983-6767

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CIÊNCIA DA COMPUTAÇÃO

PROJETO DE CONSULTORIA EMPRESARIAL

GESTÃO FINANCEIRA

FISCON

SÃO JOÃO DA BOA VISTA, SP

NOVEMBRO 2022

ISSN 1983-6767

UNIFEOB

CENTRO UNIVERSITÁRIO DA FUNDAÇÃO DE ENSINO OCTÁVIO BASTOS

ESCOLA DE NEGÓCIOS

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CIÊNCIA DA COMPUTAÇÃO

PROJETO DE CONSULTORIA EMPRESARIAL

GESTÃO FINANCEIRA

FISCON

MÓDULO MODELAGEM E DESENVOLVIMENTO DE SISTEMAS

Gestão Financeira – Profa. Renata Alencar Marcondes

Programação Orientada a Objeto – Prof. Mauro Gloria

Lógica de Programação – Prof. Sidney Gitcoff Telles

Modelagem de Dados – Prof. Max Streicher Vallim

Projeto de Modelagem e Desenvolvimento de Sistemas – Profa Mariângela Martimbianco Santos

Estudantes:

Kamily de Oliveira Muniz, RA 22001481
Felipe Lima Quintino , RA 22000713
Diogo Daloca, RA 22001030
Luiz Felipe dos Santos Pereira, RA 22000049
Matheus Broesler, RA 22000961
Marcelo Adriano Bianchini Filho, RA 22000222

Monitor:

Caio Grilo da Cunha, RA 22000246
Fábio Luiz Barbosa Filho, RA 22000941
Altair dos Santos Santana, RA21000691

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2022

ISSN 1983-6767

SUMÁRIO

ISSN 1983-6767

1. INTRODUÇÃO

Em virtude do atual cenário competitivo no setor empresarial, torna-se imprescindível por parte das empresas, em especial as de pequeno e médio porte, se organizarem e traçarem suas metas através de um planejamento estratégico e preciso.

Sendo assim, o projeto tem como foco o desenvolvimento de um sistema de Fluxo de Caixa, na qual o empresário tenha uma visibilidade ampla de seus recursos financeiros, onde o mesmo possa: prever, planejar e controlar entradas e saídas de dinheiro em um determinado período, este que é um dos diversos benefícios que ele possa trazer para qualquer empresa que for utilizá lo, na qual serão apresentados ao longo deste projeto.

Além do modelo de Fluxo de Caixa , o usuário terá a possibilidade de monitorar nitidamente e em tempo real o que está acontecendo dentro do sistema, como por exemplo os investimentos que foram feitos na empresa e os gastos que estão sendo realizados em determinados setores, podendo então ter um controle mais eficaz de suas despesas e rendimentos, e por fim tomar decisões mais concretas e seguras para não comprometer a saúde financeira da empresa.

No decorrer deste artigo será abordado detalhadamente o processo e a metodologia usada para o desenvolvimento do modelo de Fluxo de Caixa e as vantagens de implantá-lo em uma empresa.

ISSN 1983-6767

2. DESCRIÇÃO DA EMPRESA

A FISCON - Empresa Fisco Contabil Sociedade Simples LTDA (CNPJ 48.619.449/0001-69), localizada na Rua Joaquim Valim, 98 - Jardim Satélite, São João da Boa Vista - SP (CEP 13.870-399), é uma empresa que tem por atividade realizar a apuração dos resultados fiscais e financeiros das empresas parceiras, gerenciando seus patrimônios.

Além disso, a FISCON também orienta os gestores e responsáveis quanto à administração das finanças de seus negócios, dos quais são classificados majoritariamente de MEI (Microempreendedor Individual).

ISSN 1983-6767

3. PROJETO DE CONSULTORIA EMPRESARIAL

Cada unidade de estudo possui um conteúdo responsável pela criação do projeto. Inicialmente a Gestão Financeira apresenta alguns conceitos básicos de fluxo de caixa e juros simples e compostos que são importantíssimos para o desenvolvimento correto e organizado para uma empresa.

Em Programação Orientada a Objetos, o professor Mauro, instruiu maneiras de utilizar o método POO, que é um estilo de programação que permite os desenvolvedores agruparem tarefas semelhantes em Classes.

O conteúdo de Lógica de Programação, é responsável por resolver os problemas, desenvolver um software ou executar qualquer ação por meio de um código estipular quais passos o computador deverá seguir para chegar ao objetivo final.

Por fim, Modelagem de Dados, onde o código de programação, a estrutura de POO e o fluxo de caixa desenvolvem o Banco de Dados.

3.1 GESTÃO FINANCEIRA

Nesta unidade de estudo foi mostrado os conceitos básicos de um Fluxo de Caixa e a necessidade de aplicá-lo em uma empresa seja ela de pequeno, médio ou grande porte. Além disso, mostrou a importância de se fazer o cálculo de Juros Simples e Compostos com o objetivo de corrigir os valores nas transações financeiras, isto é, a correção que se faz ao emprestar ou aplicar uma determinada quantia durante um determinado período.

3.1.1 FLUXO DE CAIXA

O Fluxo de Caixa é de suma importância para um empresário pois auxilia o controle financeiro do negócio a ser administrado, ou seja, com esta ferramenta é possível acompanhar toda a movimentação de valores da empresa, com base na entrada e saída de dinheiro.

Segundo (RODRIGUES, 2022) “Através do Fluxo de Caixa é que se sabe se uma empresa é lucrativa e sustentável. Afinal, por meio desse instrumento a organização sabe quando, como e onde deve fazer investimentos para seu crescimento.”

Dentre as diversas vantagens de se implantar um Fluxo em uma empresa pode se destacar :(BRONZERI, 2021)

- “Com o Fluxo de Caixa o usuário tem uma noção precisa de quanto dinheiro tem em caixa e quanto terá disponível em determinado período.”;
- “Melhora o clima organizacional ”;
- “Identifica onde o dinheiro da empresa está sendo gasto”;
- “Confirmar se os recursos financeiros próprios serão suficientes para tocar o negócio ou se há necessidade de buscar dinheiro extra”;

3.2 PROGRAMAÇÃO ORIENTADA A OBJETO

Nesta unidade, responsável pelo professor Mauro Glória, foram ministrados os conceitos básicos da Programação Orientada a Objetos (POO). Por conseguinte, POO classifica-se como um paradigma de programação, onde um projeto é organizado perante uma metodologia padronizada, tornando assim o projeto entendível para desenvolvedores que sequer participam na construção do mesmo. Este modelo foi criado com o intuito de aproximar o mundo real do mundo virtual e solucionar problemas recorrentes no mundo do desenvolvimento, dos quais seguem abaixo.

3.2.1 CLASSES E OBJETOS

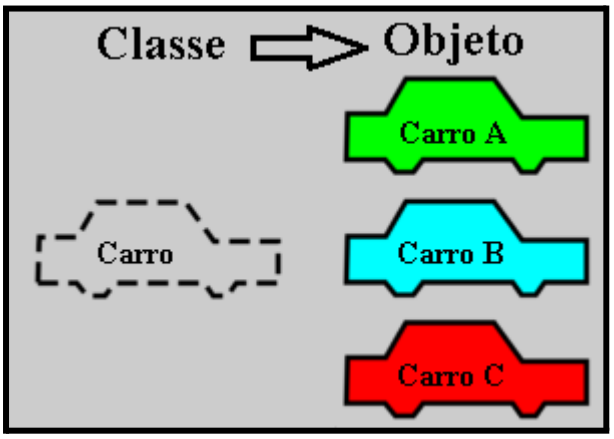
Aqui iniciamos no primeiro pilar em que a POO baseiam-se, que é a abstração. Ao lidar com uma representação de um objeto real (o que dá nome ao paradigma), deve-se planejar o que esse objeto irá realizar dentro de nosso sistema.

Suponhamos que um carro é um objeto seu mas na loja onde você o comprou existiam vários outros, muito similares, com quatro rodas, volante, faróis, dentre outras partes. Observe que, apesar do seu carro ser único (por exemplo, possui um registro único no Departamento de Trânsito), podem existir outros com exatamente os mesmos atributos, ou parecidos, ou mesmo totalmente diferentes, mas que ainda são considerados carros. Podemos dizer então que seu objeto pertence à uma classe “carro”, e que seu carro nada mais é que uma instância dessa classe chamada "carro".

Conclui-se então que uma classe é um conjunto de características e comportamentos que definem o conjunto de objetos pertencentes a essa classe. Logo, a classe em si é um conceito abstrato, como um molde, que se torna concreto e palpável através da criação de um objeto. Chamamos essa criação de instância de classe

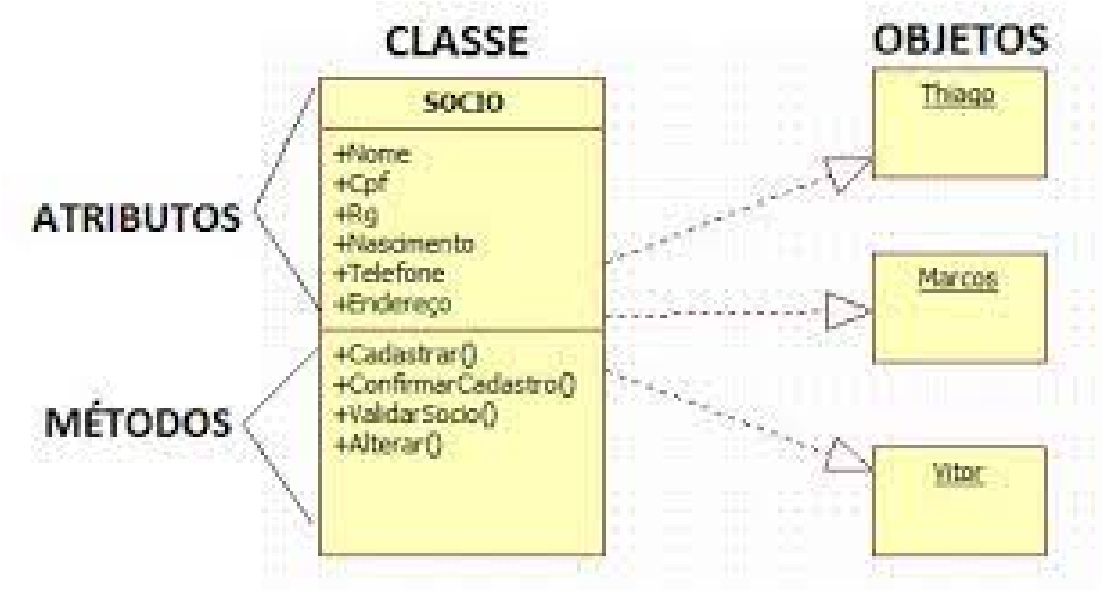
Alguns exemplos de Classes e Objetos:

Figura 1 - Classes e objetos por carros



Fonte: Produção própria dos autores

Figura 2 - Classes e objetos



Fonte: Produção própria dos autores

3.2.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO.

Ainda no primeiro pilar da POO, temos os conceitos de atributos. No mundo real qualquer objeto possui elementos que o definem. Essas características são nomeadas propriedades. Utilizando o exemplo anterior, as propriedades de um objeto “Carro” poderiam ser “Modelo”, “Cor” e “Quilometragem”.

Finalizando o conceito de abstração, se faz necessária a definição de ações que o objeto irá executar, que são chamados de métodos. Métodos são extremamente diversos, como “acenderFarol()” ou “acelerar()” em um objeto carro.

Prosseguimos então para o segundo pilar da POO, que é o encapsulamento. Se trata de um dos elementos que adicionam segurança à aplicação pelo fato de “esconder” os atributos de arquivos que não sejam da classe referente ao atributo. Logo, dizemos que o atributo “escondido” é um atributo privado. Ao usar atributos privados, forçamos a criação de métodos especiais chamados “getters” e “setters”, que irão retornar e estabelecer o valor do atributo, respectivamente. Assim, evitamos o acesso direto ao atributo do objeto e tornamos o código mais legível. Exemplificado, ao acender o farol do carro, não sabemos o que está acontecendo internamente, e de fato não precisamos saber. No fim, apenas nos interessa que os faróis liguem. Podemos então dizer que os métodos que acendem os faróis do carro estão encapsulados.

Seguimos para o terceiro pilar da POO, a programação orientada por dados, o reuso de código é uma de suas vantagens de destaque e ela se dá por herança. Essa característica otimiza a produção da aplicação em tempo e linhas de código.

Para fazer uma analogia próxima à realidade não virtual, em uma família, por exemplo, a criança herda diretamente do pai e indiretamente do avô e do bisavô. Em programação, a lógica é similar. Assim, os objetos filhos herdam as características e ações de seus “ancestrais”.

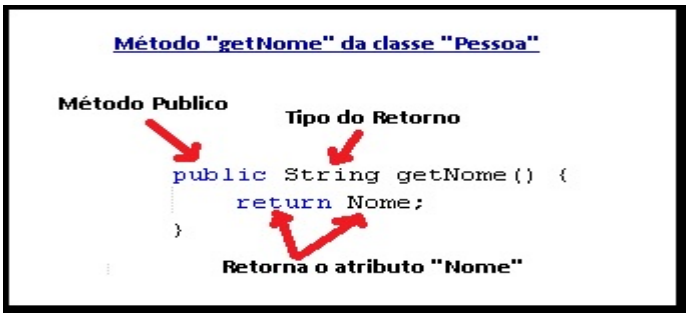
Em programação orientada a objetos, polimorfismo é o princípio pelo qual duas ou mais classes derivadas da mesma superclasse podem invocar métodos que têm a mesma assinatura, mas comportamentos distintos. Na natureza, existem animais que são capazes de alterar sua forma conforme a necessidade. Na orientação a objetos a ideia é a mesma.
O poliformismo permite herdar um método de classe pai e atribuir uma nova implementação para o método pré-definido.

3.2.3 MÉTODOS ESTÁTICOS, PÚBLICOS, PRIVADOS E PROTEGIDOS

Um dos fundamentos da orientação a objetos é evitar que classes tenham acesso a um código que não tenha a ver com sua lógica. Imagine a classe Corpo, que possui massa, volume e consequentemente densidade. Se alterarmos a massa de um corpo sem alterarmos a densidade, criamos uma inconsistência dentro da classe. Logo esse é um caso em que não podemos permitir livre acesso a esses campos.

Os métodos estáticos ou métodos da classe são funções que não dependem de nenhuma variável de instância, quando invocados executam uma função sem a dependência do conteúdo de um objeto ou a execução da instância de uma classe, conseguindo chamar direto qualquer método da classe e também manipulando alguns campos da classe. O mesmo têm um relacionamento com uma classe como um todo, enquanto os métodos que não são estáticos são associados a uma instância de classe específica (objeto) e podem manipular as variáveis de instância do objeto, como pode ser visto nos exemplos de declarações de métodos. Uma declaração com o método público pode ser acessada de qualquer lugar e por qualquer entidade que possa visualizar a classe a que ela pertence.

Figura 3 - Métodos de classe



Fonte: Produção própria dos autores

O papel de alguns métodos pode ser o de auxiliar outros métodos da mesma classe e muitas vezes, não é correto chamar esses métodos auxiliares de fora da sua classe diretamente Para garantir que métodos auxiliares não sejam chamados por código escrito fora da classe na qual eles foram definidos, podemos torná-los privados, acrescentando o modificador private.

Os métodos protegidos (protected) podem ser chamados apenas por objetos da mesma classe que os definem e suas subclasses. O acesso é mantido na família. Entretanto, o uso de protected é limitado. Métodos privados (private) não podem ser chamados com um recebedor explícito – o recebedor é sempre self.

Figura 4 - Atributos privados

```
public class Pessoa {

    private String Nome;
    private int Idade;
    private float Peso;
    private float Altura;

    public Pessoa(String Nome, int Idade, float Peso, float Altura) {
        this.Nome = Nome;
        this.Idade = Idade;
        this.Peso = Peso;
        this.Altura = Altura;
    }
}
```

Fonte: Produção própria dos autores

3.3 LÓGICA DE PROGRAMAÇÃO

Nesta unidade, responsável pelo professor Sydnei, foram ministrados os conceitos básicos da lógica da programação. Sabemos que a lógica de programação é uma forma de estender o nosso pensamento, tentando traduzir nosso raciocínio para os computadores, com a intenção de fazer com que eles se tornem um pouco mais inteligentes e consigam desempenhar tarefas.

Vamos tomar como exemplo o café que tomamos de manhã. Quando perguntam como tomamos nosso café, a maioria de nós responde que, ao acordarmos, preparamos o café com auxílio de uma cafeteira elétrica, colocamos ele em uma caneca e o tomamos.

Mas, ao destrinchar este processo, somos capazes de estipular uma sequência de passos que nos levaram ao ato final de beber este café. A sequência a seguir pode ser denominada um Algoritmo :

1. Ao acordar, levanto da cama;
2. Após levantar da cama, desço as escadas;
3. Após descer as escadas, entro na cozinha;
4. Após entrar na cozinha, pego o pó de café no armário;
5. Após pegar o pó de café, o coloco dentro da cafeteira;
6. Após colocar o pó na cafeteira, jogo água no compartimento específico;
7. Após inserir todos os ingredientes na máquina, aperto o botão de ligar;
8. Quando o café está pronto, pego a garrafa;
9. Após pegar a garrafa, despeje o café dentro de uma caneca;
10. Após colocar o café na caneca, bebo o café.

Se detalharmos ainda mais este processo, é possível incluir mais passos dentro desta sequência. Logo, lógica programação pode ser um programa é composto por três tipos básicos de estruturas:

- Sequências: Comandos a serem executados;
- Condições: Comandos que só devem ser executadas se uma condição for satisfeita;
- Repetições: Comandos que devem ser executados repetidamente até que uma condição seja satisfeita.

3.3.1 LÓGICA DE PROGRAMAÇÃO P.O.O.

Na lógica de programação existem características que formam o conceito de POO. Para uma linguagem de programação ser considerada orientada a objetos, devem haver quatro comportamentos, sendo eles o encapsulamento, a herança, o polimorfismo e a abstração.

A herança trata-se de uma relação de receber algo pré-existente. No caso da POO, é um evento que ocorre entre classes, onde a classe filha recebe os métodos e variáveis da classe mãe.

Já o polimorfismo é uma característica inerente aos métodos dos objetos. Significa dizer que um mesmo método pode ser utilizado em diferentes objetos, de diferentes classes, tendo a possibilidade de assumir características e comportamentos diferentes.

Por conseguinte, a abstração é modelar um objeto de forma abstrata, que seja obrigatoriamente herdado por outras classes. Suas características reais somente serão reveladas ao passo que a aplicação instância-la, definindo suas propriedades e se utilizando dos métodos.

Por fim, o encapsulamento é a capacidade que determinado método ou atributo de um objeto tem de se manter invisível. Ou seja, ele continua funcional, mas sem mostrar como. Seus atributos e métodos só poderão ser acessados dentro da própria classe

3.3.2 PROTOTIPAÇÃO

A prototipagem de software tem como objetivo central o desenvolvimento de versões incompletas e sujeitas a alteração da aplicação, a fim de alinhar as requisições do cliente com aquilo que está sendo projetado, bem como a possibilidade de realizar testes de interface, experiência do usuário e as funcionalidades da aplicação.

3.3.3 TEMPLATE

Um template é um modelo de layout pronto, montado com os elementos visuais e um texto genérico (lorem ipsum) que mostra onde você deve substituir pelas suas próprias informações. Existem muitos modelos de templates que ajudam as pessoas a criarem apresentações visuais atrativas e organizadas.

Referente ao desenvolvimento web, um template conta com toda a estrutura do site, o layout e o código-fonte do site. É um documento composto por um conjunto de arquivos, podendo conter folhas de estilos CSS, códigos HTML, scripts em JavaScript, arquivos de imagens e vídeos, dentre outras soluções para o desenvolvimento web.

ISSN 1983-6767

3.4 MODELAGEM DE DADOS

A modelagem de dados é um conceito difundido no desenvolvimento de software que permite projetar como serão construídas as estruturas de dados que darão suporte aos processos de negócios, como os dados estão organizados e quais os relacionamentos que se pretende estabelecer entre eles.

De acordo com o Blog Digital House (DIGITAL HOUSE, 2022) : “A modelagem de dados tem tudo a ver com as decisões que você toma à frente de um projeto, pois é o processo mais relevante na geração de informações coerentes, estratégicas e de valor para os negócios, tornando possível a análise e definição de todos os diferentes dados que sua empresa coleta e produz, bem como as relações entre eles. Ela consiste em construir estruturas que possibilitam o armazenamento e a recuperação de informações em pesquisas para contextos específicos.”

3.4.1 MODELO CONCEITUAL

O modelo conceitual é o primeiro passo para a projeção de um sistema de dados. Está intrinsecamente conectado com os levantamentos de requisitos que o sistema irá compor, como cadastrar clientes, registrar vendas, registrar produtos, e assim por diante. Nele, utilizamos recursos visuais que estão relacionados com a estrutura que o banco de dados pode tomar. Para a modelagem conceitual, temos as seguintes características que devem ser incorporadas:

Primeiramente, definimos quais são as entidades que serão utilizadas no sistema. Entidades são abstrações do mundo real, das quais inserimos de acordo com as necessidades daquilo que está sendo desenvolvido. As entidades podem ser baseadas em objetos do mundo real, como “Funcionário” ou “Cliente”, como também podem ser objetos não palpáveis, como “Venda” ou “Departamento”.

Segundamente, definimos quais são os relacionamentos entre as entidades elencadas. Ao passo que, por exemplo, queremos desenvolver uma aplicação que registra as vendas de um funcionário, já conseguimos extrair o relacionamento entre as entidades “Venda” e “Funcionário” pelo ato de descrever a funcionalidade do sistema, que, neste caso, poderia ser “registra”. É comum que os relacionamentos sejam nomeados utilizando verbos no presente do indicativo, como “faz”, “realiza”, “contém”, “registra”, e assim por diante.

Finalmente, definimos quais serão os atributos das entidades. Os atributos são as características que as entidades podem compor. Existem diversos tipos de atributos, sendo os principais:

- Simples: São os atributos indivisíveis, também chamados de “atributos atômicos”. Alguns exemplos são a data de nascimento de uma pessoa, o cargo de um funcionário, dentre outros.
- Composto: São atributos que permitem ser desdobrados em mais valores. Um exemplo seria um atributo endereço, que contém “Estado”, “Cidade”, “Bairro”, “Rua”, “Número” e “CEP”.
- Identificador: São atributos que têm por função, como o nome já diz, identificar aquele registro, sendo proibido a repetição desse valor na sua tabela original. Exemplos comuns são o CPF de uma pessoa ou o CNPJ de uma empresa.
- Multivalorado: São atributos que permitem a entrada de 2 ou mais dados. Um exemplo seria o atributo Telefone, podendo conter 2 números de telefone para serem inseridos

Como o nome já denota, a modelagem conceitual tem como objetivo ser apenas um conceito do projeto. Logo, muito daquilo que precisa ser definido, como tipo de dados, não se faz presente neste estágio. Para uma modelagem mais detalhada, precisamos ir além, criando então um modelo lógico do sistema.

3.4.2 MODELO LÓGICO

Tendo como base o Modelo Conceitual, o Modelo Lógico é o próximo passo para a estruturação do banco de dados, desta vez criando um modelo que se aproxima ainda mais da lógica que os requisitos de negócio demandam, que é baseado na lógica de tabelas.

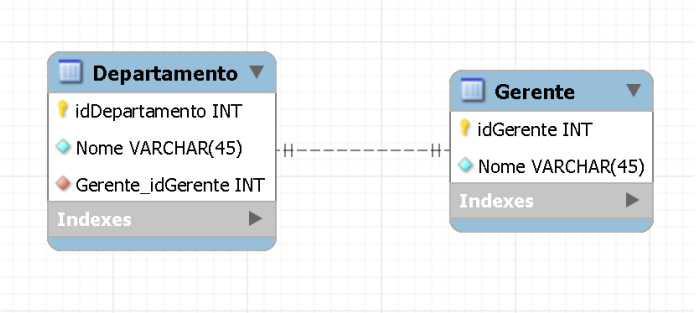
Nesta fase, a modelagem lógica determina se todos os requisitos do negócio foram reunidos. Nela, a tipificação dos dados é realizada, definindo inteiros, texto, data, booleano (verdadeiro ou falso), dentre outros. Além disso, normalmente a definição de chaves primárias e chaves estrangeiras acontece nessa fase, das quais são responsáveis por identificar as linhas de dados de suas respectivas tabelas bem como estabelecer relações entre tabelas. Ademais, outras ações podem ser realizadas, como a definição de caracteres máximo que um tipo texto pode suportar, determinar diferentes tipos de dados para preservar espaço na memória (tinyint, smallint, int, bigint), definir se o campo pode deixar de receber um valor, ou seja, se tornando nulo, bem como definir se o campo precisa ser

preenchido (not null), predefinir valores para um campo (default value), dentre outros controles lógicos.

Por fim, nesta fase a relação entre as entidades (tabelas) se torna mais sofisticada, sendo necessário entender a diferença entre os 3 tipos de relacionamentos.

Relacionamento 1 para 1:Neste tipo de relacionamento, só pode haver um tipo de ocorrência de chave estrangeira na tabela dependente. Seguindo o exemplo abaixo, um departamento pode ter apenas um gerente, e um gerente pode gerenciar apenas um departamento.

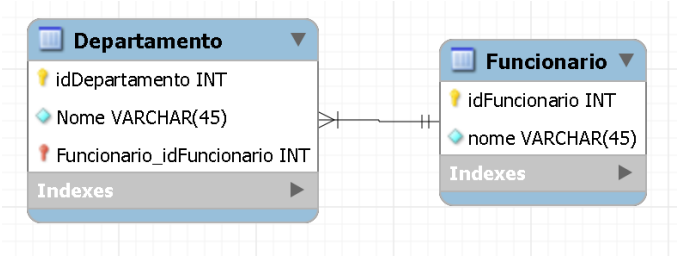
Figura 5 - Relacionamento 1 para 1



Fonte: Produção própria dos autores

Relacionamento 1 para N: Neste tipo de relacionamento, a tabela dependente pode receber vários valores diferentes da tabela relacionada. Seguindo o exemplo abaixo, um departamento pode ter vários funcionários diferentes, mas cada funcionário pode trabalhar apenas em um departamento.

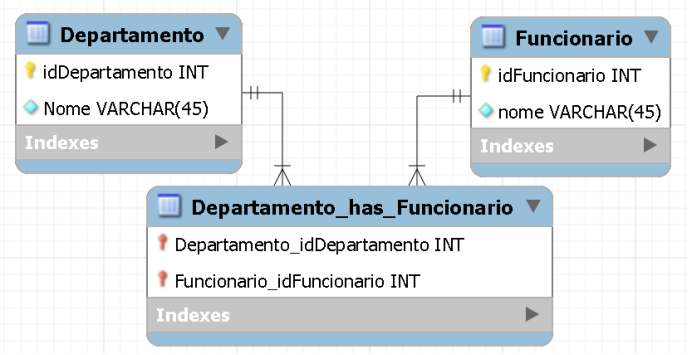
Figura 6 - Relacionamento 1 para N



Fonte: Produção própria dos autores

Relacionamento N para M: Neste tipo de relacionamento, ambas as tabelas podem receber vários tipos de valores perante sua relação. Seguindo o exemplo abaixo, um departamento pode ter vários funcionários, e um funcionário pode trabalhar em vários departamentos.

Figura 7 - Relacionamento N para N



Fonte: Produção própria dos autores

3.4.3 SQL

O SQL (Structured Query Language) é uma linguagem de computador para trabalhar com conjuntos de fatos e as relações entre eles. Programas de banco de dados relacionais. Como muitas linguagens de computador, SQL é um padrão internacional reconhecido por entidades de padrões como ISO e ANSI, porém existem versões diferentes mantidas por diferentes empresas, como MySQL, Microsoft SQL Server, e PostgreSQL, que criaram funções específicas para atender às necessidades dos usuários.

SQL é utilizado para criar, ler, alterar, excluir dados, bem como realizar a manutenção contínua deles, a fim de utilizá-los em uma aplicação. Por exemplo, uma instrução SQL simples

A linguagem SQL é usada para executar comandos em bancos de dados relacionais, ou seja, bancos de dados baseados em tabelas. Ela é dividida em alguns subgrupos:

- DML – Data Manipulation Language: São os comandos que alteram informações nas tabelas, seja para inserir ou excluir dados. Utilizamos “INSERT” para inserir dados, “UPDATE” para atualizar dados já existentes e “DELETE” para apagar dados;
- DDL – Data Definition Language: São comandos que modificam o banco de dados. Utilizamos o “CREATE” para criar novos bancos de dados e tabelas, “ALTER” para fazer modificações nos objetos criados através do “CREATE” e “DROP” para remover aquilo que foi criado com o “CREATE”;
- DCL – Data Control Language: é o grupo responsável pelas permissões, restrições ou bloqueios lógicos no banco de dados;
- DTL – Linguagem de Transição de Dados: é responsável por salvar as alterações feitas pelos usuários.
- DQL - Data Query Language: São os comandos que fazem consultas nas tabelas. Utilizamos o “SELECT” para fazer a operação.

Imagine que temos uma tabela para guardar alguns dados de clientes. Então, para consultar todos os dados dos clientes, o comando que devemos utilizar é uma que vai selecionar (SELECT) todos os campos (*) da tabela (FROM) clientes:

- SELECT * FROM clientes;

Já o INSERT é responsável por inserir dados dentro da tabela. A sintaxe base é:

- INSERT INTO tabela (campo1, campo2, campo3…) VALUES (valor1, valor2, valor3…);

Neste caso vamos inserir novos dados na tabela de clientes, passando os argumentos no insert, que estão relacionados com o cadastro do Miguel.

- INSERT INTO clientes (id, nome, telefone, gênero) VALUES (5, ‘Miguel’, 4444-4444, ‘M’);

Prosseguindo, o UPDATE atualiza as linhas dentro da tabela, mas não podemos esquecer o WHERE onde inserimos uma condição, ou seja, uma regra que impõe a execução do comando. Se a condição não for informada a tabela inteira será atualizada indiscriminadamente. A sintaxe base é:

- UPDATE tabela SET campo = valor1 WHERE id = valor2;

Utilizando como exemplo, vamos atualizar o telefone de um cliente que tem como código o número 4 dentro do nosso banco de dados:

- UPDATE clientes SET telefone = 2222-2222 WHERE id = 4

Por fim, o DELETE apaga os dados registrados dentro da tabela. Assim como o UPDATE, não podemos esquecer do WHERE. Caso contrário, toda tabela será apagada. A sintaxe base é:

- DELETE FROM tabela WHERE id = valor1

Utilizando como exemplo, vamos apagar os dados de um cliente que tem como código o número 5 dentro do nosso banco de dados:

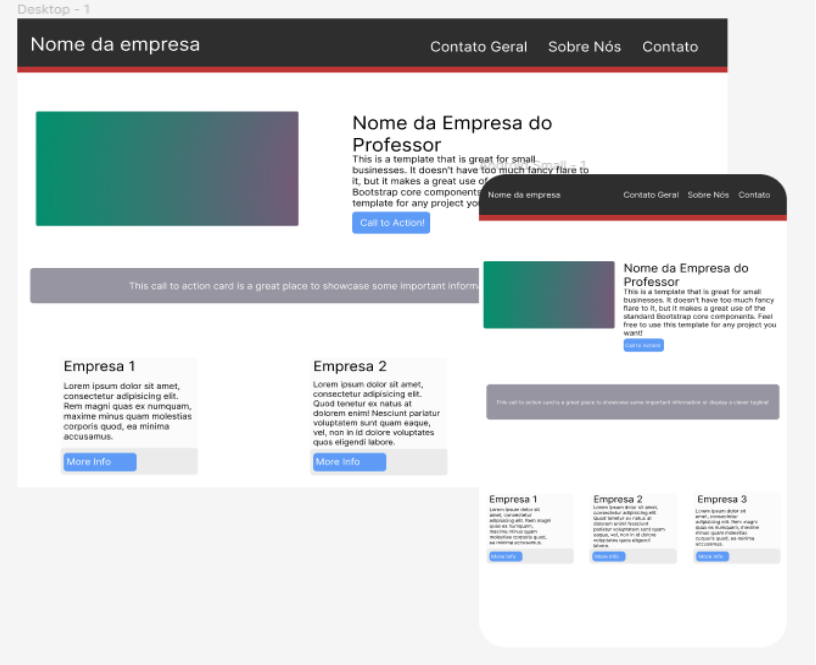
- DELETE FROM clientes WHERE id = 5

4. RESULTADOS

Nesta seção serão apresentados os resultados obtidos através das aulas ministradas, tendo em vista os requisitos estabelecidos pelos cliente e o desenvolvimento realizados pelos integrantes do grupo

Primeiramente, criamos um protótipo visual para o site, tomando como base um template para negócios.

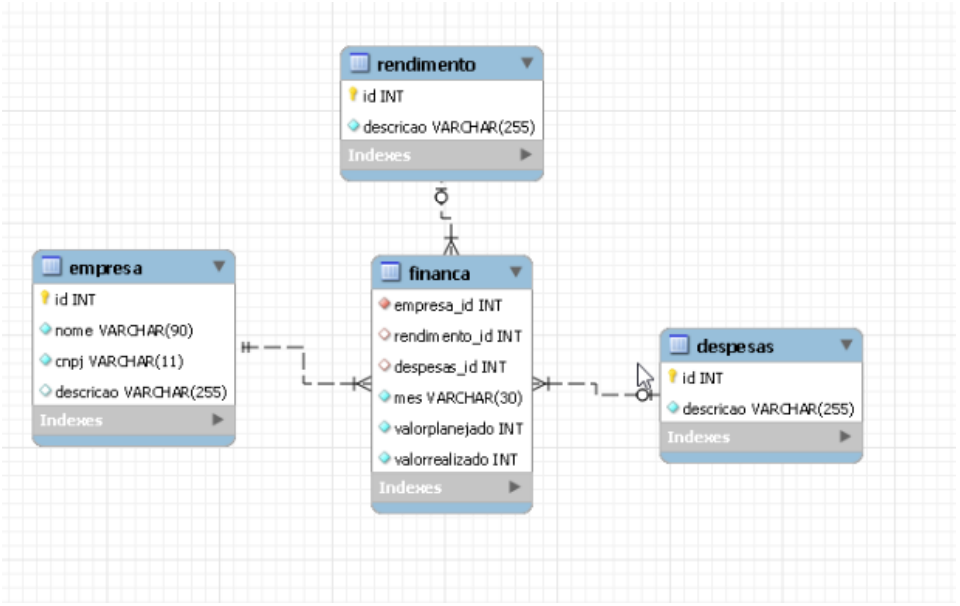
Figura 8 - Protótipo visual



Fonte: Produção própria dos autores

Em seguida, definimos as funcionalidades que deveriam compor a aplicação e realizamos a modelagem lógica do banco de dados. Através da ferramenta de desenvolvimento MySQL WorkBench, geramos o código do banco de dados utilizando como base o modelo lógico

Figura 9 - Modelo Lógico



Fonte: Produção própria dos autores

Reunindo todos os conhecimentos disponibilizados pelas unidades de estudo, chegamos numa aplicação web, que contém as seguintes telas:

- Tela home, onde está a apresentação do site;

Figura 10 - Tela Home

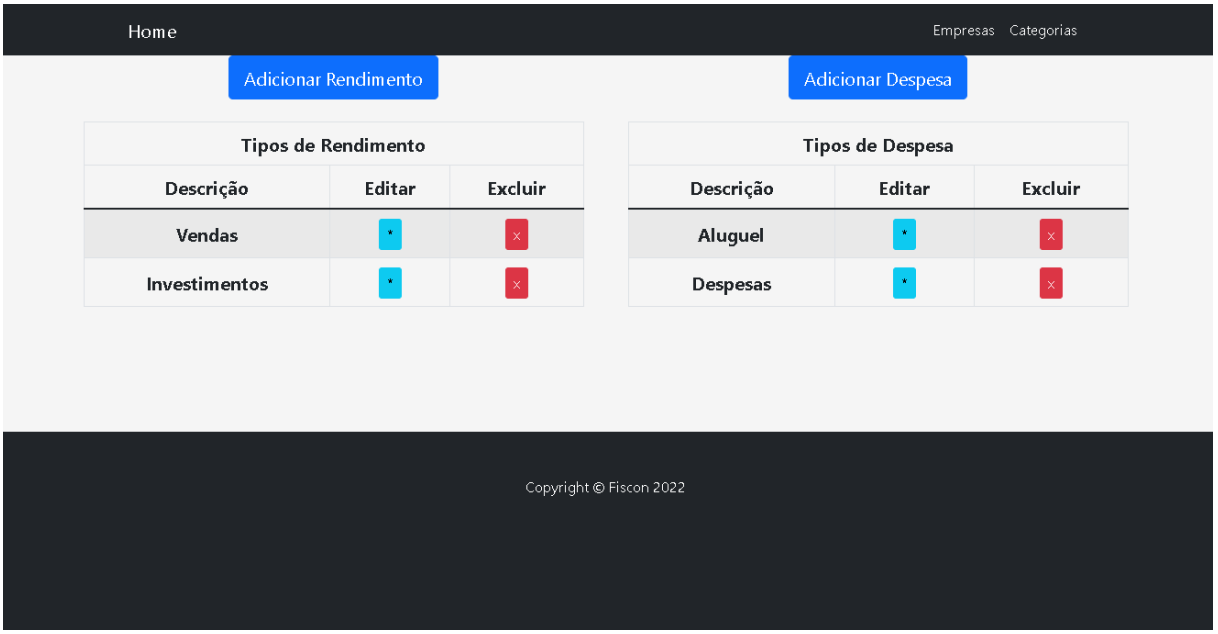
Fonte: Produção própria dos autores

Figura 11 - Tela Home

Fonte: Produção própria dos autores

- Tela de categorias, onde o administrador pode cadastrar, alterar e excluir as possibilidades de renda e despesas utilizadas pelas empresas do sistema:

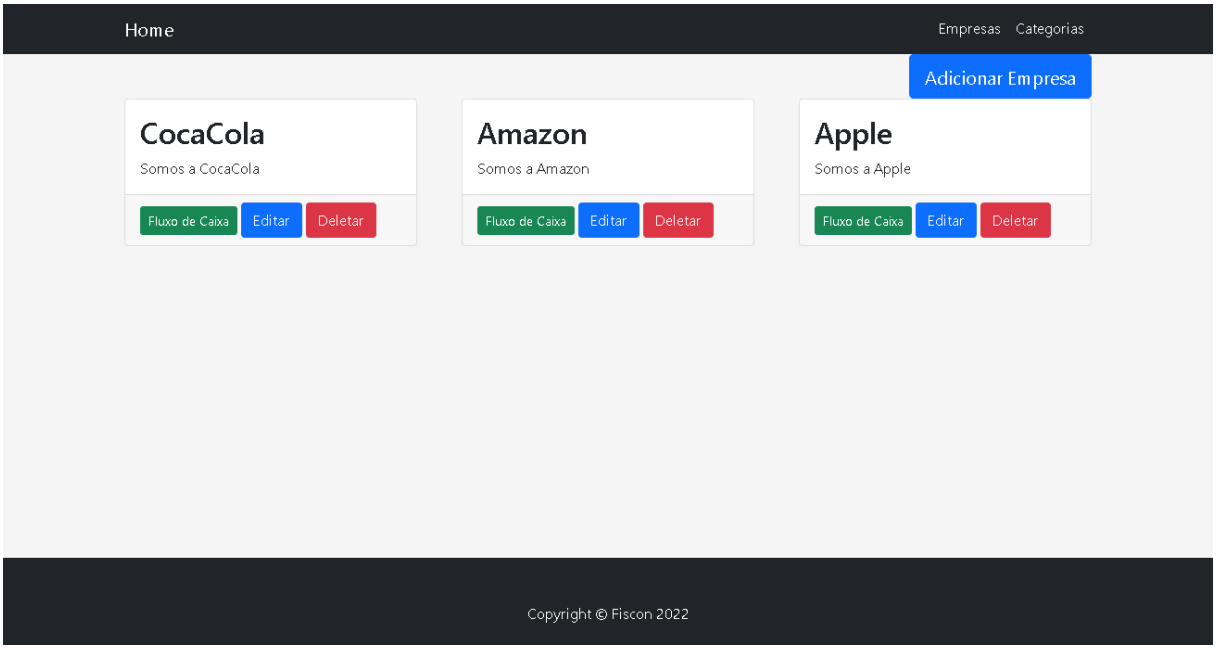
Figura 12 - Tela categorias



Fonte: Produção própria dos autores

- Tela de empresas, onde o administrador poderá cadastrar, alterar e excluir as empresas que farão parte do sistema:

Figura 13 - Tela empresas



Fonte: Produção própria dos autores

- E, por fim, a tela de fluxo de caixa, onde a empresa poderá fazer o cadastro de suas rendas e despesas e planejar seus movimentos capitais:

Figura 14 - Tela fluxo de caixa

Fonte: Produção própria dos autores

ISSN 1983-6767

5. CONCLUSÃO

O projeto de gestão empresarial para criação de um sistema que realiza o controle de fluxo de caixa para a empresa FISCON foi uma experiência muito construtiva, nos concedendo a possibilidade de colocar ensinamentos passados neste módulo na prática.

O principal desafio que enfrentamos foi a integração entre banco de dados MySQL e código PHP e HTML, por se tratarem de tecnologias das quais detemos pouca experiência. O que se demonstrou extremamente eficiente, tanto no desenvolvimento do projeto como na fixação do conteúdo ministrado, foram os diversos alinhamentos entre os professores e o grupo, onde pudemos estabelecer um direcionamento certeiro para atingirmos os objetivos definidos.

O resultado foi satisfatório, tendo em vista todo o trabalho realizado e conhecimento adquirido. Acreditamos que o projeto realizado atende às necessidades e expectativas da empresa FISCON.

REFERÊNCIAS

BRONZERI, Gizele Bronzeri. **Benefícios reais de um controle de Fluxo de Caixa. Loopa Digital,** 2021. Disponível em: https://blog.loopa.digital/controle-de-fluxo-de-caixa/. Acesso em: 17 mar. 2021.

DE SOUZA OLIVEIRA, Igor. **Programação Orientada a Objetos e Programação Estruturada: Vamos abordar neste artigo qual tipo de programação deve ser utilizada no decorrer do desenvolvimento:** orientada a objetos ou programação estruturada. Devmedia, 2015. Disponível em: https://www.devmedia.com.br/programacao-orientada-a-objetos-e-programacao-estruturada/32813Acesso em: 17 out. 2022.

HENRIQUE , João. **POO: o que é programação orientada a objetos?:** Programação orientada a objetos e programação estruturada. **alura,** 2022. Disponível em: https://www.alura.com.br/artigos/poo-programacao-orientada-a-objetos. Acesso em: 11 maio 2022.

HENRIQUE. **Os 4 pilares da Programação Orientada a Objetos:** Conheça nesse artigo os 4 principais pilares, bem como as diferenças para programação estruturada e as principais vantagens da POO.. devmedia, 2014. Disponível em: https://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264.

HOUSE DIGITAL, Equipe. **Modelagem de dados: como gerar informações coerentes, estratégicas e de valor para os negócios: A modelagem de dados constrói estruturas para armazenar e recuperar informações, dando coerência e forma a dados que, soltos, não fariam sentido**.. Blog Digital House, 2022. Disponível em: https://www.digitalhouse.com/br/blog/modelagemdedados/. Acesso em: 19 abr. 2022.

NOLETO, Cairo . **POO: tudo sobre Programação Orientada a Objetos!: Quais são os 4 pilares básicos da POO?.** betty be, 2017. Disponível em: https://blog.betrybe.com/tecnologia/poo-programacao-orientada-a-objetos/#:~:text=abordados%20mais%20adiante.-,O%20que%20%C3%A9%20o%20paradigma%20de%20POO%3F,c%C3%B3digos%2C%20m%C3%A9todos%2C%20entre%20outros.. Acesso em: 20 out. 2022.

RODRIGUES, Ana Luiza. **Fluxo de caixa: o que é, os tipos e quais os benefícios?.** Rede Jornal Contábil, 2022. Disponível em: https://www.jornalcontabil.com.br/fluxo-de-caixa-o-que-e-os-tipos-e-quais-os-beneficios/. Acesso em: 22 jun. 2022.

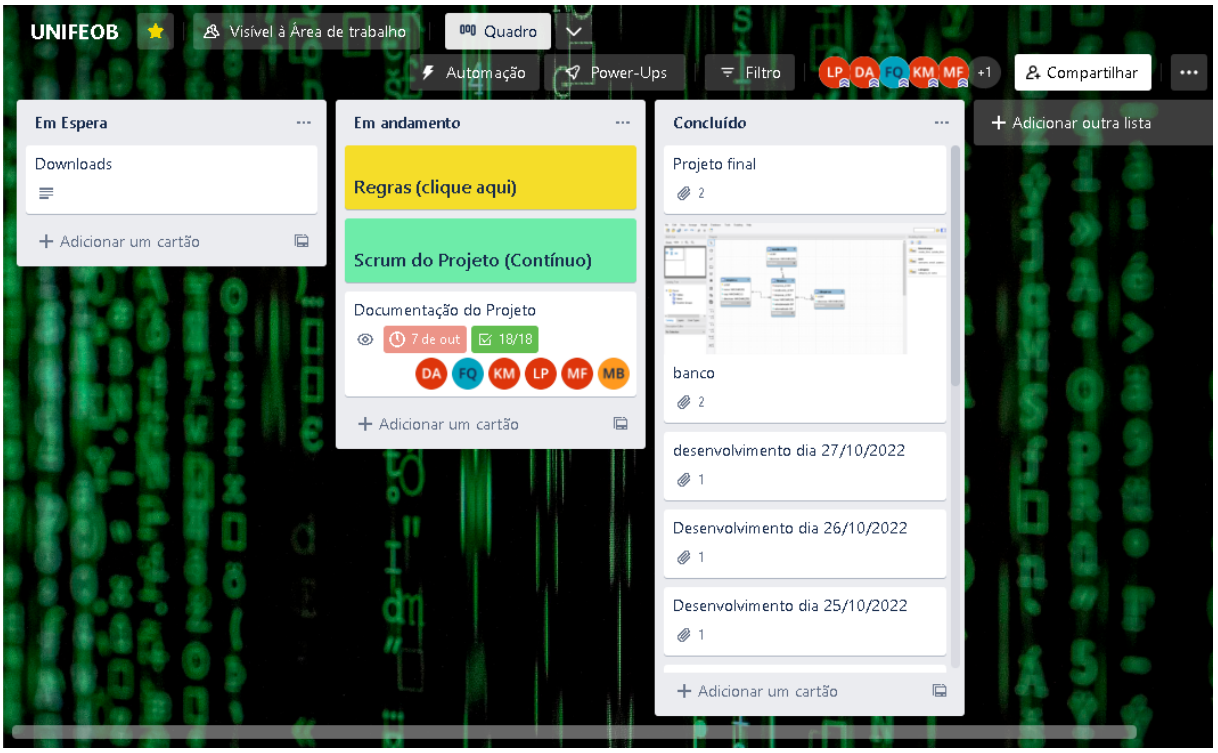
SANTOS DE JESUS, Luciano. **Métodos Get e Set (Encapsulamento)**. teste velocidade. 2020. Disponível em: https://www.testeavelocidade.com.br/metodos-get-e-set-encapsulamento/. Acesso em: 29 maio 2020.

THIAGO. **Trabalhando com métodos em Java:** Veja neste artigo como trabalhar com métodos na linguagem Java, dividindo o código em blocos, facilitando a sua implementação e

manutenção.. devmedia, 2012. Disponível em: https://www.devmedia.com.br/trabalhando-com-metodos-em-java/25917.

WALDERSON. **ENCAPSULAMENTO**. walderson.com, 2014. Disponível em: http://walderson.com/site/wp-content/uploads/2014/08/POO-06-07.pdf.

ANEXOS


UNIFEOB
Visível à Área de trabalho
Quadro
Automação
Power-Ups
Filtro
LP DA FQ KM MF +1
Compartilhar
Em Espera
Downloads
Adicionar um cartão
Em andamento
Regras (clique aqui)
Scrum do Projeto (Contínuo)
Documentação do Projeto
7 de out
18/18
DA FQ KM LP MF MB
Adicionar um cartão
Concluído
Projeto final
2
banco
2
desenvolvimento dia 27/10/2022
1
Desenvolvimento dia 26/10/2022
1
Desenvolvimento dia 25/10/2022
1
Adicionar um cartão
Adicionar outra lista

UNIFEOB | ESCOLA DE NEGÓCIOS