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

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

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 Glória

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:

Marcos Valverde de Mira Ferreira, RA 22001184
William Alves Chaves, RA 22000255
Antonio Domingues Neto, RA 22001699
Douglas Vinicius Nobrega, RA 22000041
Gustavo Henrique Tomaz, RA 22001161
Caio Rodrigues Stopa, RA 22000689
Bruno Henrique do Prado Franco, RA 22001702

Monitor:

Caio Grilo da Cunha, RA 22000246
Fábio Luiz Barbosa Filho, RA 22000941
Altair dos Santos Santana, RA 22000691

SÃO JOÃO DA BOA VISTA, SP
NOVEMBRO 2022

ISSN 1983-6767

SUMÁRIO

ISSN 1983-6767

1. INTRODUÇÃO

A princípio, quando falamos de planejamento financeiro, estamos ligando diretamente ao controle de ganhos e gastos, se organizar, exige muito mais que querer, é necessário saber como fazer e esse método é conhecido como Fluxo de Caixa. De acordo com o IBGE (Instituto Brasileiro de Geografia e Estatística), mais ou menos 60% das empresas vão à falência nos primeiros 5 anos, isso nos mostra que é mais que essencial um planejamento, seja ele, otimista ou pessimista.

O fluxo de caixa é algo fulcral para as empresas, comércios e lojas devido ao alto fluxo de dinheiro, esse controle contribui para que os estabelecimentos tenham um registro detalhado de seus ganhos e gastos para administrar com disciplina e sem erros. Segundo o Sebrae (Segundo o Serviço de Apoio às Micro e Pequenas Empresas) na pandemia, 7% das empresas fecharam por não obter lucro e 20% por falta de capital.

Além disso, não é difícil efetuar esse controle, apenas precisamos armazenar as informações conhecidas entradas e saídas: contas à receber (entrada), contas à pagar (saída), lucros obtidos (entrada), vendas (entrada), folhas de pagamentos (saída), para assim calcular lucro, saldo, fazer projeções para possíveis imprevistos e observações para ter maior precisão na exibição dos resultados daquele mês/ano para um melhor planejamento de seus investimentos e projetos futuros.

Dessa forma, nossa proposta busca desenvolver um software que facilite esse controle para que o cliente, ele receberá as entradas e saídas, comumente conhecido, e mostrará um histórico detalhado, para que o cliente saiba onde está pecando mais. Ademais, nossa proposta também inclui uma facilidade nos registro, para que o nosso cliente saiba achar o que ele quiser, de maneira fácil e rápida, uma projeção através de gráficos para ficar prático e intuitivo.

ISSN 1983-6767

2. DESCRIÇÃO DA EMPRESA

A Fiscon é uma empresa prestadora de serviços para empresas na área contábil, fiscal, financeira e de recursos humanos.

Fundada no ano de 1966, contando com mais de 50 anos de experiência no ramo, a Fiscon visa fazer um trabalho personalizado e voltado para qualidade nos serviços executados, utilizando um sistema que se adapta às necessidades do cliente, aumentando a interatividade e a produtividade com os afiliados.

A empresa tem como objetivo se tornar referência nas áreas contábeis e fortalecer a tradição de ser conhecida por seus serviços prestados com excelência.

3. PROJETO DE CONSULTORIA EMPRESARIAL

Tendo em vista que buscamos nos desenvolver, esse projeto une o essencial de cada módulo e o desafio de fazê-los funcionar juntos. Cada unidade tem sua especificidade.

Primeiramente, foi apresentado a proposta diretamente conjunta com gestão financeira, nessa aula aprendemos o que é um fluxo de caixa, como funciona e porque utilizá-lo. E é basicamente sobre isso que nosso projeto gira.

Além disso, está muito atrelado também ao Programação Orientado a Objeto (POO), que é extremamente necessário caso você queira se tornar um desenvolvedor, onde aprendemos como fazer a parte prática, bem como, armazenar os dados (em conjunto com Modelagem de Dados), formulários para receber dados, que envolve toda a parte por trás do software.

Ademais, a Lógica de Programação busca contato direto com POO, nesta unidade aprendemos sobre a lógica dos códigos e como utilizar o Bootstrap, ferramenta para criar a "frente" do nosso projeto.

Por último, mas não menos importante, a unidade Modelagem de Dados é um dos pontos chaves do projeto, toda a parte de armazenamento de informações é aprendida aqui, atuando em conjunto com a POO.

Em síntese, une-se um ao outro e todos agem juntos para a criação do Fluxo de Caixa. Passamos por todas as etapas, tanto *"back-end"* quanto *"front-end"*, após aprendermos sobre o que constitui um Fluxo de caixa, fizemos a parte por trás, formulários e etc, também fizemos a parte da frente, a parte do *"style"* e por último o banco de dados para a o armazenamento, onde vamos aplicar na prática com a empresa Fiscon.

3.1 GESTÃO FINANCEIRA

Para Hayrton "A gestão financeira é um conjunto de ações e procedimentos administrativos que envolvem o planejamento, a análise e o controle das atividades financeiras da empresa", com base nisto, podemos dizer que gestão financeira é sem dúvidas algo mais que importante para o controle de gastos, no caso de uma empresa, efetua o registro

de entradas e saídas, alinhando com o planejamento financeiro, além disso, ajuda a administrar o saldo e como investi-lo, para que não haja contratempos.

Ademais, temos também um termo conhecido como "projeções", onde existem 3 tipos: Real, Otimista e Pessimista.

Otimista é tendo em vista que tudo vai dar certo e não haverá problemas, e se espera que tenha um aumento significativo comparado com o “real”, que por sua vez é quando mantém-se a mesma coisa.

Além disso, temos também o pessimista, que por sua vez, é quando há contratempos e devido a eles, há uma perda significativa de capital.

3.1.1 FLUXO DE CAIXA

Segundo Braga (1995) “Fluxo de Caixa é a estimativa dos fluxos de pagamentos e recebimentos, distribuídos durante a vida útil do projeto e constitui o ponto de partida do orçamento de capital”. Pensando nisso, nosso projeto busca receber dados financeiros de uma empresa como entrada e saída, registrá-las e mostrar em forma de análise, para o nosso cliente.

Esses termos de “entrada” e “saída” são bastante utilizados em gestão financeira, onde a entrada é o capital que entra na sua empresa, como uma venda, por exemplo, e saída é o que sai da sua empresa, como por exemplo um pagamento, ou uma compra. A partir disso, temos tambem̀ o caixa da empresa, que constitui-se na soma de entradas e saídas.

3.2 PROGRAMAÇÃO ORIENTADA A OBJETO

Em primeiro lugar, temos que explicar o que é Programação Orientada a Objetos(POO), do inglês *“Object Oriented Programming”*(OOP), é um método de programação que busca a aproximação dos mundos, virtual e real, estruturado através da interação de 2 “objetos”, por exemplo: Herança de classes. Em paralelo a isso, podemos afirmar também que POO é construído utilizando “objetos” e “classes” contendo “métodos” e “atributos”.

Ademais, para nosso projeto, utilizamos em conjunto com o POO, a linguagem php, para a parte “back-end” do site.

3.2.1 CLASSES E OBJETOS

Pensando em conceitos básicos, podemos dizer que, as classes podem ser pensadas como uma "categoria" para os objetos, cada objeto é construído a partir de uma classe. Além disso, de acordo com o desenvolvedor Matthew Weier, as classes são utilizadas para descrever entidades que possuem comportamentos conhecidos, como mensagem para funções e objetos, para agrupar ou estados relacionados e fornecem uma maneira de descrever hierarquias e heranças em um aplicativo.

Nessa parte de classes, em paralelo com os conceitos do fluxo de caixa, criamos por exemplo a classe " class Caixa{ ", nela definimos o que vai receber e armazenar os objetos.

Falando em objetos, em php, "[...] é um tipo de dado composto (junto com *"arrays"*). Valores de mais de um tipo podem ser armazenados juntos em uma única variável. Objeto é uma instância de uma classe interna ou definida pelo usuário. Além das propriedades, a classe define a funcionalidade associada aos dados" (Lathkar, 2020, tradução nossa).

Bem como, de acordo com o que foi apresentado nos pelo professor Mauro Glória, criamos os objetos "private $valor;", "private $datar;", "private $saldo;" e "private $descricao;". nesses objetos foram definidos variáveis que recebera valores em conjunto com o mySQL.

3.2.2 ATRIBUTOS, MÉTODOS, ENCAPSULAMENTO, HERANÇA E POLIMORFISMO.

Falando mais a fundo sobre termos que usamos em POO, temos os atributos, que são basicamente, as variáveis contidas na classe, existindo 2 tipos, *"Private"* e *"Public"*.Dito isso, a principal diferença entre eles, é que o *"Private"* só pode ser acessado dentro da classe em questão, já o *"Public"* pode ser acessado por qualquer outro código externo.

Ademais, falando em métodos, existem o *"get"* e o *"set"*. Falando em *"get"*, é um "array" associativo de variáveis que são passadas para um um script através de parâmetros URL(query string). No caso do *"set"*, é uma sequência de valores únicos(variáveis).

Nessa mesma linha, temos o encapsulamento, que tem a intenção de proteger acessos aos membros internos de um objeto. "Toda classe é responsável pelos seus atributos, assim podendo acessar esses mesmos atributos utilizando métodos da própria classe"(Kadu

Oliveira), dito isso, os atributos não devem ser acessados fora da classe, para os seus que seus valores se mantenham seguros.

Por meio disso, existem também o termo "herança", nele uma subclasse derivada de um classe herda todos seus métodos, sendo muito útil pois permite a implementação de uma funcionalidade adicional aos seus objetos sem necessariamente ter que adicionar novamente todas as funcionalidades compartilhadas.

Em último mas não menos importante, temos o polimorfismo que nos permite classes derivadas de uma classe “pai” (superclasse) tenham métodos iguais mas comportamentos diferentes. Abaixo deixaremos um exemplo prático de cada.

Ex:

```
//tudo que está em chaves{} é chamado de encapsulamento
class Caixa{ // definindo uma classe

private $valor;
//metodo set para alterar o valor do atributo privado $valor
public function setValor($valor){
   $this -> valor = $valor;
}
//metodo get para acessar o atributo privado $valor
public function getvalor(){
   return $this -> valor;
}
//Abaixo um exemplo de herança
<?php

class A {
   public int $prop;
}
class B extends A {
   public readonly int $prop;
}
?>
//Temos abaixo um exemplo de polimorfismo
<?
class Mamifero extends Animal
{
   public function locomover()
```

```
    {
        echo "Mamíferos correm!";
    }
}
//aqui ocorre o polimorfismo
class Ave extends Animal
{
    public function locomover()
    {
        echo "Aves voam!";
    }
}
Mamifero->locomover();
Ave->locomover();
```

3.2.3 MÉTODOS ESTÁTICOS, PÚBLICOS E PRIVADOS

Aprofundando em métodos, como foi citado anteriormente temos os públicos("public") e privados(“private”), que são variáveis contidas numa classe. Em POO, temos uma terceira, que é conhecida como estática (“static”), nela é definido que um atributo ou método, pertence a classe e não à uma instância dela, sendo assim, pode ser acessada sem instanciar um objeto;

Ex: Private

```
class Caixa{
    private $valor;
    private $datar;
    private $saldo;
    private $descricao;
```

Ex: Public

```
class MinhaClasse
{
    public $publica = 'Public';
    function imprimeAlo()
    {
        echo $this->publica;
    }
}
```

```
$obj = new MinhaClasse();
echo $obj->publica; // Funciona
```

Ex: Static

```
class Foo
{
   public static $meu_estatico = 'foo';
   public function valorEstatico() {
      return self::$meu_estatico;
   }
}
```

3.3 LÓGICA DE PROGRAMAÇÃO

A priori, pensando em lógica de programação, é um conceito pensado na organização de um código voltado à resolução de um problema, ou a criação de um software. Mesmo que cada linguagem tenha suas particularidades, sintaxe e orientação, mas no conceito lógica todas seguem a mesma linha de base, usado tanto em *"back-end"* quanto *"front-end"*.

De acordo com Steve Jobs "Everybody in this country should learn how to program a computer because it teaches you how to think." - (Todo mundo neste país deveria aprender a programar porque isso te ensina a pensar)", por isso o conceito de lógica se torna tão importante, é além de só programação, lógica e organização mental.

Ademais, pensando em *"front-end"*, temos a parte de prototipação e parte de template. No caso da prototipação, feita no figma, é parte mais "superficial" do projeto, nela a nossa ideia é mostrar a ideia que temos para o projeto, como ele ficará visualmente falando e onde cada botão ficará localizado, para assim facilitar o desenvolvimento.

Ainda falando de *"front-end",* existe o template, feito em conjunto com o bootstrap, é parte visual a partir da prototipação, nela é realmente feita, codificada e mostrada para o cliente como ficará. Diferente do protótipo que é mostrar as ideias, no template, é executar as ideias, portanto assim, mostrar a parte visual funcionando em conjunto com a *"back-end"* da Programação Orientada a Objetos (POO).

3.3.1 LÓGICA DE PROGRAMAÇÃO O.O.

Como lógica de programação e POO são ligadas a criação do mapa de classes, serve como uma linha de base para a criação do projeto. Falando especificamente em php, nosso projeto criamos várias classes relacionadas com fluxo de caixa, para receber valores, conhecidos como objetos, e armazená-los num banco de dados em conjunto com o sql.

3.3.2 PROTOTIPAÇÃO

Criação de protótipo na plataforma Figma para conseguir visualizar como a é a ideia que foi mapeada do cliente. Assim construir uma representação daquilo que acreditamos ser a solução para as necessidades do cliente

Nessa parte do projeto, a equipe junto ideias, e utilizando o figma, criamos a idealização das telas, nessas telas nós buscamos colocar tudo que achávamos necessário para assim, poder criar o juntamente com o template, o site do fluxo de caixa.

Na home buscamos adicionar tudo de mais importante num fluxo de caixa, como, o saldo geral do caixa, transações esperando ser aprovadas, entradas, saídas, gráficos e o balanço geral.*figura 1*

Ademais, nas telas 1, 2 e 3, é parte de registro, como login, cadastro de email, recuperação e toda parte que envolve registro.*figura 2,3 e 4*

Pensando em algo que una tudo, temos botões que buscam ser interativos e práticos, na home, por exemplo, temos um botão que encaminha nosso cliente para a tela de inserir movimentações, nossa ideia é que todos os dados que forem inserido nessa tela, seja mandando diretamente para a tabela.*figura 5*

Em anexos terá também, o link do figma completo, e as telas de entrada, saída e saldo.

3.3.3 TEMPLATE

Por mais que nosso conhecimento seja o suficiente para a criação de uma página web do zero, usar um template serve para a praticidade e economia de tempo. Template é basicamente um conjunto de códigos pré criado, onde você baixa e modifica ele da forma que você quiser, nele já vem funcionalidades e visual, bastando apenas você adaptar para sua ideia.

Utilizando bootstrap, que é basicamente uma biblioteca de templates, a equipe buscou o que mais encaixaria com a ideia do nosso projeto, após escolhido, modificamos para que fique 100% do nosso jeito. Na aba anexos, terá o link do template original, para a visualização antes das modificações feitas pelo grupo.

3.4 MODELAGEM DE DADOS

Primordialmente, quando pensamos em modelagem de dados, é ligado diretamente ao banco de dados, e isso envolve várias aplicações teóricas e práticas, que visam construir um modelo de dados consistente, para ser aplicado à SGBD(Sistema Geral de Banco de Dados) moderno. Consistindo-se basicamente em desenhar o sistema de informações, concentrando-se entidades lógicas e dependência lógica entre essas entidades.

No quesito dados, existem 2 tipos de modelos, conceituais e lógicos, que são feitos basicamente para idealizar como o banco de dados vai ser constituído.

3.4.1 MODELO CONCEITUAL

A equipe deve entender o problema e propor uma solução para o banco de dados do sistema, demonstrando o modelo por meio de um DER (Diagrama de Entidade Relacionamento), devidamente documentado.

Como foi dito anteriormente, o modelo conceitual, é para discutir aspectos do negócio do cliente e não nos quesitos da tecnologia. Nessa parte identificamos o que é preciso ter no banco e mostramos para o cliente, pois é fácil de entender pois não aplica tecnologia. Deixaremos em Anexos, um exemplo desse modelo.

3.4.2 MODELO LÓGICO

Os estudantes devem implementar o projeto do banco de dados utilizando o SGBD MySQL. Tal banco será utilizado para o sistema.

No caso do lógico, já trata de algo mais técnico, pensando em paralelo com o conceitual, ele já adequa padrões e nomenclaturas, sendo mais voltado aos desenvolvedores do que realmente ao cliente.

Nessa parte, antes dos códigos, criamos as tabelas lógicas para ter uma ideia ampla de como realizar o fluxo de caixa, quais chaves precisam e como eles se uniriam no banco de dados. Além disso, o relacionamento entre eles ajuda bastante quando se trata de entendimento de dados, como por exemplo, criar uma tabela caixa, na qual essa tabela recebe dados e saídas. Deixaremos em anexos, figuras de exemplo.

3.4.3 SQL

Chegando mais afundo, SQL consiste na parte dos códigos, pensando em banco de dados, os modelos conceituais e lógicos servem mais para o entendimento, a parte conceitual, já essa parte, é considerada mais difícil por ser a prática, nela utilizando comandos como o Insert(inserir dados), Update(atualizar dados), Delete(apagar dados) e o Select(selecionar dados), é possível criar um database(banco de dados) e fazer a ligação com o template e php para o recebimento e armazenamento desses dados.

Falando sobre o nosso trabalho, nós criamos a database fluxo de caixa, e uma tabela caixa, nela usamos o insert para inserir dados, e o select para mostrar dados, sendo apenas para popular e ver como ficaria recebendo dados de verdade. Após isso, fomos realmente para o projeto, identificamos que, para o fluxo de caixa, uma tabela precisaria armazenar vários dados, vindo do php, para realizar essa ligação utilizamos os aprendizados com o professor Mauro Glória.

```
CREATE TABLE IF NOT EXISTS `caixa` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `valor` float() DEFAULT '0',
 `data` date() DEFAULT '0',
 `descricao` varchar(100) DEFAULT '0',

 PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8mb4;
```

Nesse código mostramos a criação da database, da tabela e de suas variáveis.

```
<?php
```

```
require_once '../../config/db.php';
$db = new db();
$conexao = $db->conectar();
$id = $_GET['id'];
$sql = " SELECT * FROM caixa where id = " . $id;
$resultado = mysqli_query($conexao,$sql) or die("Erro ao retornar dados");

// Obtendo os dados por meio de um loop while
while ($registro = mysqli_fetch_array($resultado))
{
  $valor = $registro['valor'];
  $datar = $registro['datar'];
  $descricao = $registro['descricao'];



  echo $valor . ' - ' . $datar . ' - ' . $descricao. '<br />' ;
}
mysqli_close($conn);
```

Nesse código é mostrado uma ligação do php, com o sql, e nele a gente já define quais são as variáveis que o banco vai receber, a partir disso, ele tem que armazenar e mostrar utilizando o select.

ISSN 1983-6767

4. CONCLUSÃO

Em suma, sobre o nosso trabalho, podemos dizer que foi feito com excelência, desde a idealização do projeto, no campo de fluxo de caixa, passando pela programação orientada a objetos fazendo a parte “back-end” em conjunto com a lógica de programação, que também nos ajudou bastante tanto no “back-end” quanto no “front-end”, para finalizar na parte de recebimento de dados feito no sql em conjunto como Php do POO.

Nossa equipe se dedicou bastante para esse trabalho, uma das nossas maiores dificuldades foi o manuseio do template mas que foi resolvido rápido pois nos ajudamos uns aos outros. Mesmo que seja um trabalho que a ideia inicial foge um pouco da nossa área, a conclusão final que eu tenho é que ampliar nossos conhecimentos nunca é demais.

Gostaria de agradecer a minha equipe, sem a ajuda de todos esse trabalho não seria possível, fomos resilientes e persistentes, e quero também agradecer aos professores, pois graças também a eles nosso projeto foi realizado da melhor forma possível.

REFERÊNCIAS

$_GET. Disponível em: <https://www.php.net/manual/pt_BR/reserved.variables.get.php>. Acesso em: 21 out 2022.

$_SET. Disponível em: <https://www.php.net/manual/pt_BR/class.ds-set.php>. Acesso em: 21 out 2022.

FILHO, Carvalho Rogério Cláudio. **Encapsulamento em PHP:** Curso de PHP. Disponível em: <http://excript.com/php/encapsulamento-php.html>. Acesso em: 21 out 2022.

Hayrton. **Gestão Financeira:** Relevância para o sucesso empresarial. Disponível em: <http://revistaconexao.aems.edu.br>. Acesso em: 17 out. 2022.

Herança de objetos. Disponível em: <https://www.php.net/manual/pt_BR/language.oop5.inheritance.php>. Acesso em: 21 out 2022.

LATHAKAR, Malhar. **PHP Objects**. Disponível em: <https://www.tutorialspoint.com/php-objects#:~:text=In%20PHP%2C%20Object%20is%20a,defines%20functionality%20associated%20with%20data.>. Acesso em : 18 out. 2022.

MODELAGEM DE DADOS: MODELO CONCEITUAL, MODELO LÓGICO E FÍSICO. Disponível em: <https://www.luis.blog.br/modelagem-de-dados-modelo-conceitual-modelo-logico-e-fisico.html>. Acesso em: 21 out 2022

NOLETO, Cairo. **Tudo sobre Programação Orientado a Objetos!**. Disponível em: <https://blog.betrybe.com/tecnologia/poo-programacao-orientada-a-objetos/>. Acesso em: 18 out. 2022.

O'PHINNEY, Weier Matthew. **What is a Class PHP:** Learning About a Class PHP. Disponível em: <https://www.zend.com/blog/what-php-class#:~:text=A%20PHP%20class%2C%20and%20more,to%20functions%20and%20other%20objects.>. Acesso em : 18 out. 2022

ROHAN, Vats. **OOP vs POP:** Difference Between OOP and POP. Disponível em: <https://www.upgrad.com/blog/oop-vs-pop/>. Acesso em : 18 out. 2022.

ROVEDA, Ugo. **Lógica de programação:** o que é e por que é importante. Disponível em: <https://kenzie.com.br/blog/logica-de-programacao/>. Acesso em: 21 out 2022.

SAKATA, Zak. **If you're really good at OOP**. Disponível em: <https://medium.com/@zak.s_/if-youre-really-good-at-oop-you-re-poop-e76b95811415#:~:text=Object%20Oriented%20Programming%20or%20OOP,the%20ability%20to%20be%20expanded>. Acesso em : 18 out. 2022.

SILVA, Benedito Junior. **Atributos em PHP**. Disponível em: <https://www.criandobits.com.br/atributos-em-php/#:~:text=O%20termo%20atributo%20faz%20menção,post%20veja%20atributos%20em%20PHP.>. Acesso em: 21 out 2022.

SILVESTRE, Gabriel. **Introdução ao SQL**. Disponível em: <https://dev.to/gabrielhsilvestre/introducao-ao-sql-287g>. Acesso em 21 out 2022.

ANEXOS

Figma:
<https://www.figma.com/file/vrNqfPpDEbVMNVBxnYV0PK/Untitled?node-id=0%3A1>

Protótipo usado <https://startbootstrap.com/template/sb-admin>

Figura 1

Figura 2

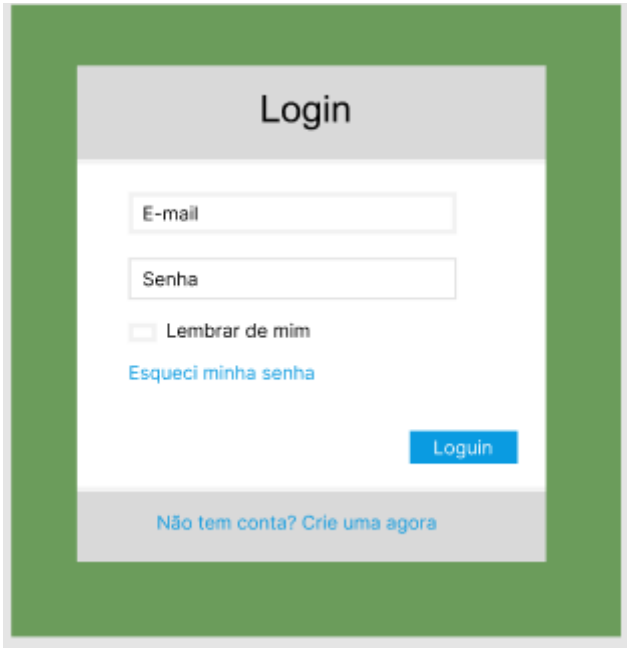


Figura 3

Figura 4

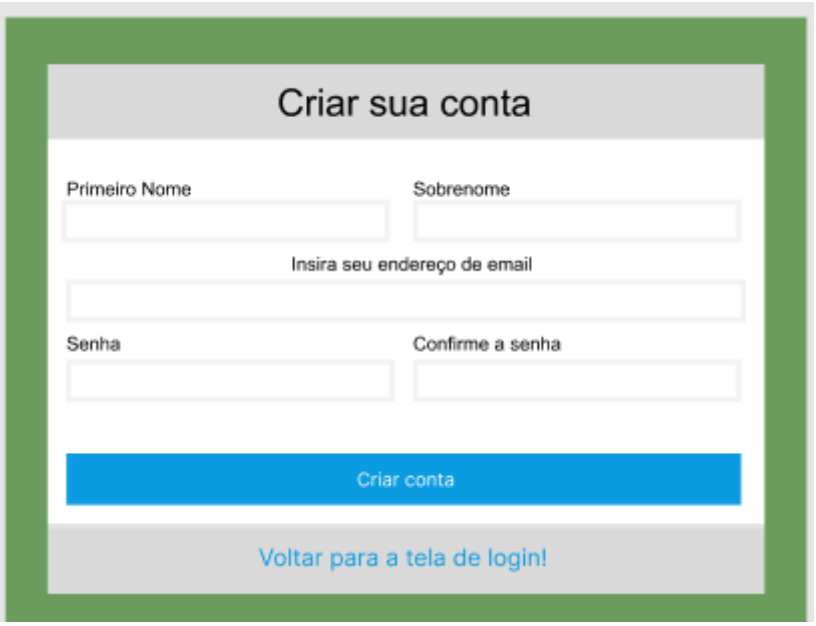


Figura 5

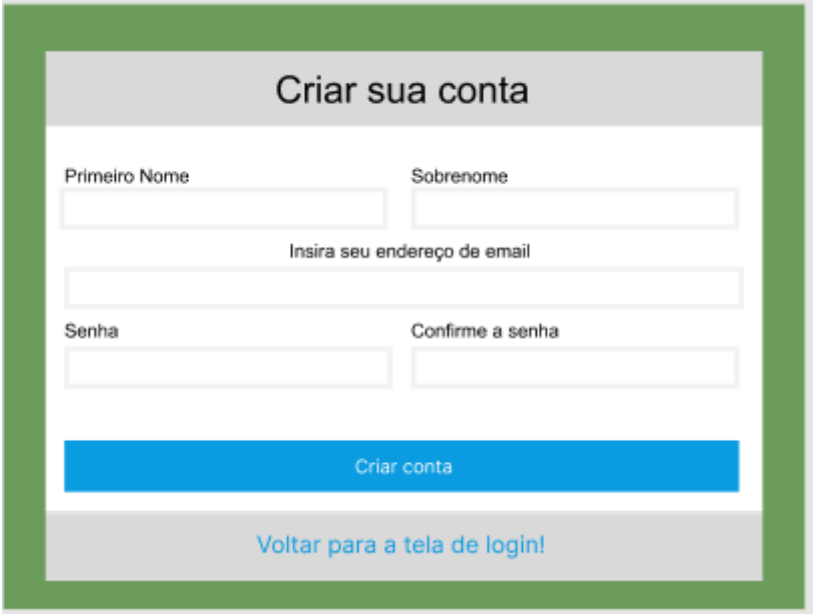
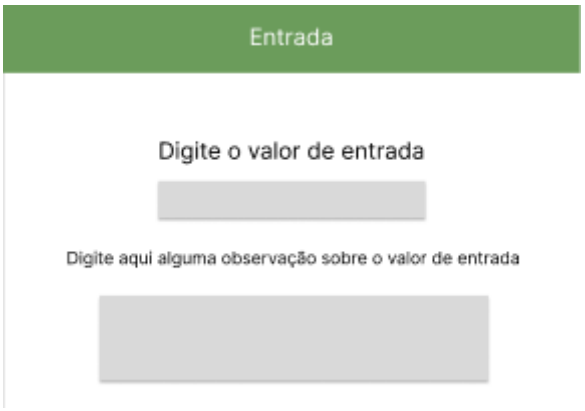
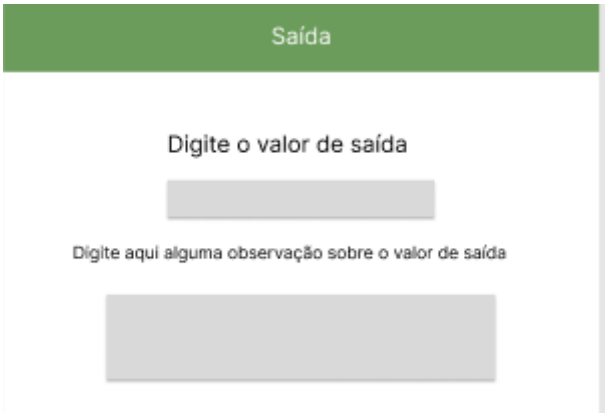


Figura 6

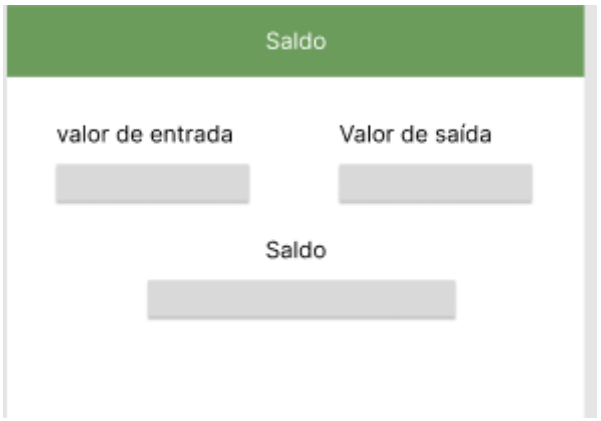
Tela de Entrada



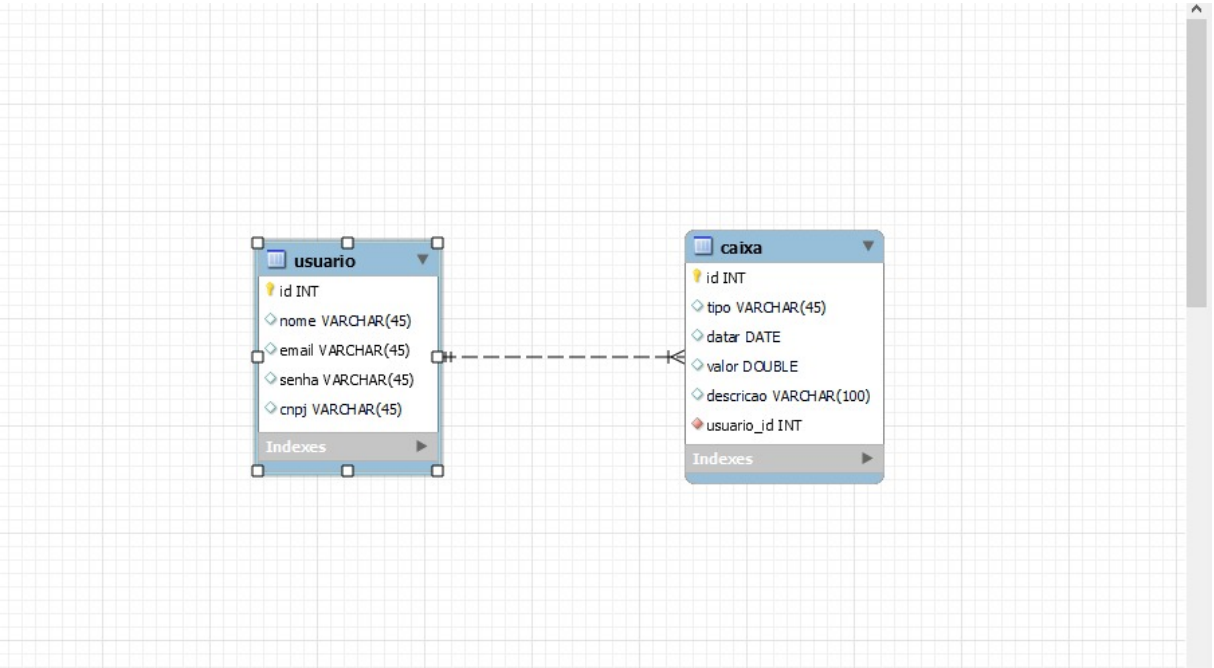
Tela de Saída



Tela de Saldo



Modelo Lógico



Modelo Conceitual