

# BOXBUSTER - LOCADORA

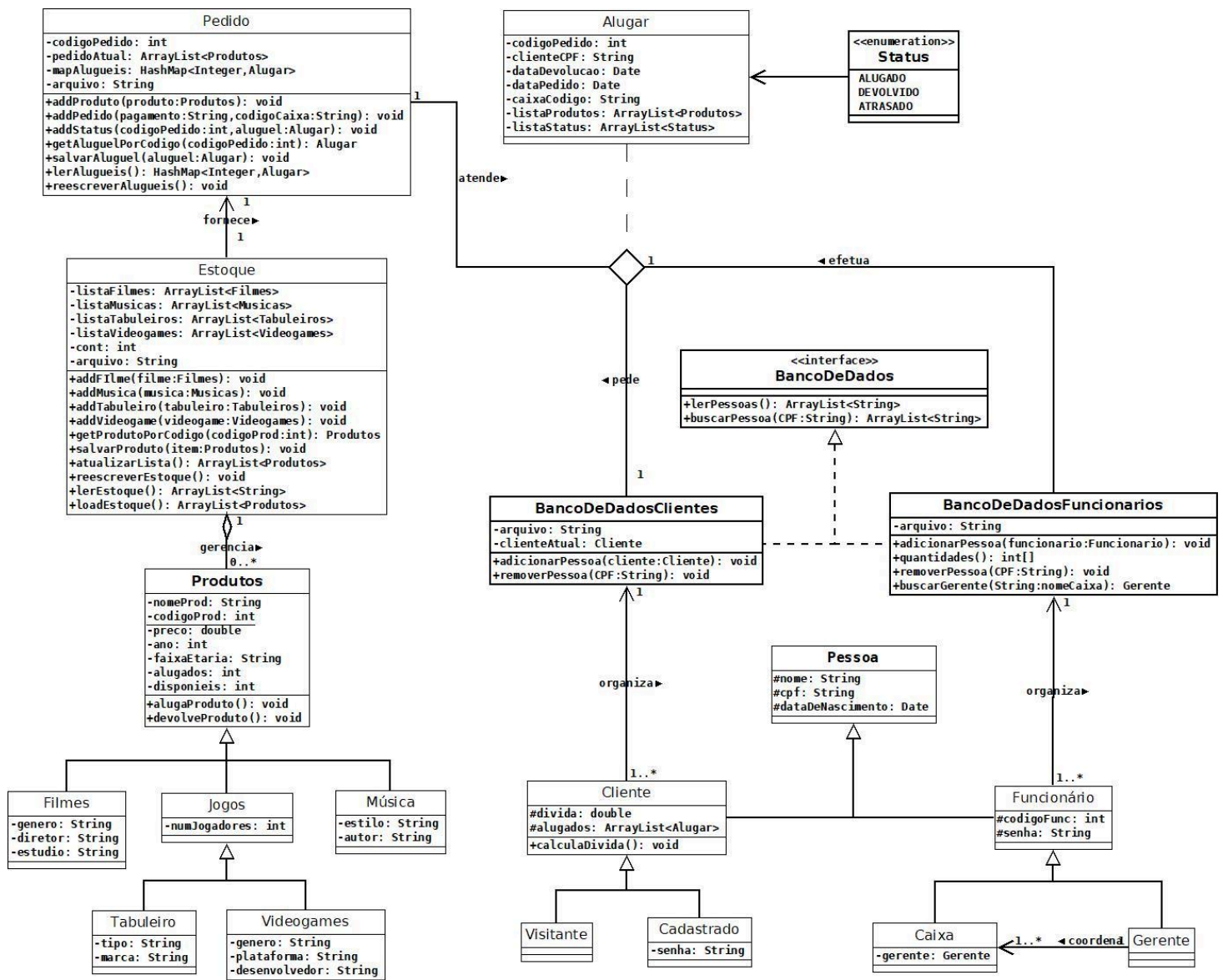
Elis Rodrigues Borges 231018875  
Henrique Cerqueira Ramos Sales 231034841  
José Edson Martins Bezerra da Silva 231003380

Universidade de Brasília, Dep. de Ciências da Computação, Brasil  
Técnicas de Programação 1 - CIC 0197 - 2024/1 - Grupo 5  
Turma 03 - Prof<sup>a</sup>: Roberta Barbosa Oliveira

## 1. Descrição do problema

A primeira locadora foi aberta em 1975, para o aluguel de fitas de vídeo. Com o passar dos anos, as locadoras do mundo todo viram a ascensão de novas mídias e novos formatos, e, eventualmente, se tornaram cada vez menos comuns com a popularização de serviços de *streaming*. No entanto, utilizando o paradigma de Orientação a Objetos, pode ser possível restaurar a cultura das locadoras à sua antiga popularidade e oferecer ainda mais funcionalidades e serviços, sem perder as referências estéticas do início dos anos 2000. Foi com esse intuito que a Locadora Boxbuster foi fundada, e a aplicação que criamos tem o papel de gerenciar cada produto do estoque, os dados de cada aluguel e as funções que os clientes e os funcionários são capazes de realizar.

## 2. Diagrama de classes



Algumas das mudanças feitas foram:

- Adicionadas listas para cada tipo de produto ao “Estoque”
- Atributo “dataPedido” adicionado à classe “Alugar”
- Atributo “CPF” da classe “Pessoa” mudado de int para String
- Adicionada interface “BancoDeDados”
- Adicionada Classe “BancoDeDadosClientes”
- Adicionada Classe “BancoDeDadosFuncionarios”
- Removido “codigoCliente” da classe “Cadastrado”
- Mudanças nos tipos de relacionamentos
- Métodos adicionados

### 3. Regras de negócio

- É possível alugar filmes, músicas, jogos de mesa e videogames.
- Na loja, são apresentados os detalhes de cada produto e a disponibilidade deles.
- Cada cliente pode ter vários aluguéis ativos simultaneamente.
- Cada pedido diz respeito a um ou mais produtos, tem um prazo máximo de 30 dias para devolução, e possui um caixa associado a ele.
- Clientes cadastrados têm direito a um desconto de 10% no valor de cada produto.
- Só é possível alugar produtos que estejam disponíveis no estoque, com eles sendo retirados do estoque ao serem alugados e adicionados de volta ao serem devolvidos.
- O gerente é a única pessoa que tem acesso direto ao estoque e à equipe, e pode cadastrar e editar produtos e funcionários.
- O caixa tem acesso ao histórico de pedidos efetuados por ele.
- Caso o prazo para devolução do produto não seja respeitado pelo cliente, ele terá que pagar uma multa de metade do preço do produto (também com um desconto de 10% se o usuário for cadastrado), e enquanto essa multa não for paga e o produto não for devolvido, ele não poderá fazer mais pedidos.

### 4. Descrição das classes

#### 4.1. Classes de pessoas

##### + Pessoa

É uma superclasse abstrata da qual derivam os funcionários e clientes da locadora.

##### Atributos:

- (String) Nome: pode conter uma ou mais palavras, aceitando apenas o primeiro nome, nomes compostos ou nome e sobrenome.
- (String) CPF: deve ser único para cada pessoa, já que é a principal maneira de identificar os clientes.
- (Date) Data de Nascimento: é usado para calcular a idade dos usuários do sistema.

##### Métodos:

A classe só possui métodos construtores e o método toString().

##### + Funcionario

É uma subclasse abstrata que estende a classe Pessoa e caracteriza a equipe da locadora, composta por gerentes e caixas, que usam ela como uma superclasse.

##### Atributos:

Herda os atributos da classe Pessoa.

- (String) Senha: campo usado para realizar o login.
- (int) codigoFunc: será usado para identificar funcionários e para fazer login

##### Métodos:

Além de herdar os métodos da classe Pessoa, ela só possui métodos construtores, getters, setters e o método toString().

## **+ Caixa**

É uma subclasse que estende a classe Funcionario e os designa a trabalhar como caixas, responsáveis pelos pedidos dos clientes.

### **Atributos:**

Herda os atributos das classes Pessoa e Funcionario.

- (Gerente) gerente: é usado para designar o gerente responsável por cada caixa.

### **Métodos:**

Além de herdar os métodos das classes Funcionario e Pessoa, a classe só possui métodos construtores, getters, setters e o método toString().

## **+ Gerente**

É uma classe que estende a classe Funcionário e os designa como gerentes, responsáveis pelo cadastro de caixas e de outros gerentes e por adicionar itens ao estoque.

### **Atributos:**

Apenas herda atributos das superclasses Funcionario e Pessoa.

### **Métodos:**

A classe herda a maioria de seus métodos, só possuindo os métodos construtores e o método toString().

## **+ Cliente:**

É uma classe abstrata que estende a classe Pessoa e caracteriza os clientes da locadora. É dividida entre Cadastrados e Visitantes.

### **Atributos:**

Herda os atributos da classe Pessoa.

- (double) divida: armazena quanto o cliente deve à loja por produtos atrasados.

- (ArrayList<Alugar>) alugados: contém uma lista de todos os alugueis daquele cliente, com suas informações.

### **Métodos:**

Além dos métodos herdados, ela possui os métodos construtores, getters e setters e toString().

+ (abstract void) calculaDivida(): é um método abstrato que será usado para calcular a dívida pendente de produtos atrasados.

+ (void) addAlugado(Alugar): adiciona um objeto Alugar à lista de alugados.

## **+ Cadastrado**

É uma classe que estende a classe Cliente e que representa os usuários cadastrados.

### **Atributos:**

Herda os atributos das classes Pessoa e Cliente.

- (String) senha: é usado para realizar o cadastro e login do cliente cadastrado.

### **Métodos:**

Além dos métodos que a classe herda, ela possui os métodos construtores, getters e setters e toString().

- (void) calculaDivida(): implementa esse método abstrato da superclasse calculando o valor da dívida com um desconto de 10% pelo cliente ser cadastrado.

### + Visitante

É uma classe que estende a classe Cliente e que representa os usuários visitantes.

#### Atributos:

Seus atributos são apenas aqueles herdados das superclasses Pessoa e Cliente.

#### Métodos:

Além daqueles herdados, possui os métodos construtores, getters e setters e toString().

- (void) calculaDivida(): implementa esse método abstrato da superclasse calculando o valor da dívida sem desconto pelo cliente ser visitante e não cadastrado.

## 4.2. Armazenamento de informações das pessoas

### + BancoDeDados (interface)

É a interface básica para as duas classes que foram criadas para manipulação de arquivos com informações de pessoas. As classes que a implementam escrevem, em um arquivo .txt, uma String referente às informações de objetos da mesma classe, para que seus dados fiquem armazenados mesmo quando o programa não está sendo executado e possam ser recuperados posteriormente.

#### Métodos:

+ (ArrayList<String>) lerPessoas(): retorna uma ArrayList de todos os objetos armazenados no arquivo, em que cada elemento é uma string com todos os dados de um objeto

+ (ArrayList<String>) buscarPessoa(String): busca na String de cada objeto por uma determinada substring e, caso encontre um objeto correspondente à busca, retorna uma ArrayList de Strings em que cada String é um dado desse objeto

### + BancoDeDadosFuncionarios

É uma classe que implementa a interface BancoDeDados, responsável por manipular o arquivo "funcionarios.txt", que armazena as informações de todos os funcionários da locadora.

#### Atributos:

- (static final String) arquivo: cujo valor é "funcionarios.txt", o nome do arquivo em que as informações se encontram.

#### Métodos:

A classe possui um construtor e os métodos sobrescritos da interface BancoDeDados.

+ (void) adicionarPessoa(Funcionario): adiciona uma linha ao arquivo "funcionarios.txt" com as informações de um Funcionário, no formato "Nome\_CPF\_DataDeNascimento\_Senha\_Código\_Cargo", e, caso o cargo seja Caixa, adiciona-se "\_Gerenciado\_por\_NomeDoGerente".

+ (int[]) quantidades(): calcula a quantidade total de gerentes e de caixas para mostrar esse valor na tela AreaGerente.

+ (void) removerPessoa(String): usa um algoritmo semelhante ao buscarPessoa para encontrar uma linha específica do arquivo e removê-la.

+ (Gerente) buscarGerente(String): dado o nome de um Caixa, retorna o Gerente correspondente no arquivo.

### + BancoDeDadosClientes

É uma classe que implementa a interface BancoDeDados, e sua função é manipular o arquivo "clientes.txt", responsável por armazenar as informações de todos os clientes da loja que já usaram o sistema, sejam eles visitantes ou cadastrados.

#### Atributos:

- (static final String) arquivo: esse atributo representa o nome do arquivo trabalhado, e no caso do BancoDeDadosClientes, o nome do arquivo é "clientes.txt"

- (static Cliente) clienteAtual: representa o cliente atual que está logado no software da loja, assim, qualquer tela consegue acessar suas informações por meio desse atributo.

**Métodos :**

A classe possui um construtor e os métodos sobrescritos da interface BancoDeDados.

+ (static Cliente) getClienteAtual e (static void) setClienteAtual(Cliente) são usados respectivamente para recuperar o objeto armazenado no atributo e para alterar suas informações.

+ (void) adicionarPessoa(Cliente): adiciona uma linha no arquivo “clientes.txt” com as informações de um Cliente, no formato “Cadastrado/Visitante\_Nome\_CPF\_DataDeNascimento\_Divida\_Senha”.

+ (static ArrayList<Alugar>) getHistoricoCliente(String): pelo CPF do cliente, esse método estático faz uma busca em todos os pedidos já feitos e acha quais estão ligados ao CPF do cliente que está usando o software. Assim, retorna uma ArrayList com todos os aluguéis já feitos pelo cliente.

+ (void) removerPessoa(String): usa o CPF de um cliente para remover suas informações do banco de dados dos clientes. É usado quando um usuário visitante deseja se tornar cadastrado, ou quando um cliente edita suas informações. Assim, suas informações antigas são deletadas e suas novas informações são inseridas.

A classe também conta com um construtor que tem como único parâmetro uma String com o nome do arquivo e atribui esse nome ao atributo “arquivo”.

### 4.3. Classes de produtos

**+ Produtos**

É uma classe abstrata que serve como superclasse para todos os itens que podem ser alugados, definindo os atributos e métodos que eles compartilham.

**Atributos:**

- (String) nomeProd: atributo que armazena o nome e certos detalhes do produto, sendo necessário que ele seja único.
- (double) preco: representa o preço base do produto, que pode receber um desconto caso o usuário seja cadastrado.
- (int) ano: informa o ano de criação do produto em questão.
- (int) codigoProd: é um número de identificação único para cada produto e que é utilizado para recuperar informações do item e para ordená-los.
- (int) faixaEtaria: informa a classificação indicativa do produto.
- (int) disponiveis: armazena a quantidade de cópias daquele produto disponíveis para serem alugadas.
- (int) alugados: armazena quantas cópias do produto em questão já foram alugadas.

**Métodos:**

A classe possui os métodos construtores, getters e setters, toString() e equals(Object).

+ (double) getPreco(): foi modificado para já retornar o valor do preço com desconto caso o usuário seja cadastrado.

+ (void) alugaProduto(): diminui a quantidade de disponíveis daquele produto por um e adiciona um à quantidade de alugados.

+ (void) devolveProduto(): faz o inverso do alugaProduto(), aumentando o disponíveis e diminuindo o alugados por um.

**+ Filmes**

É uma subclasse que estende a classe Produtos e que define os atributos específicos para os filmes disponíveis na locadora.

**Atributos:**

Herda os atributos da classe Produtos.

- (String) genero: informa o gênero do filme em questão.
- (String) estudio: informa o estúdio que produziu o filme.
- (int) diretor: informa o diretor que dirigiu o filme.

**Métodos:**

possui métodos construtores, getters, setters e toString().

## **+ Musicas**

É uma subclasse que estende a classe Produtos e que define os atributos específicos para os discos de música disponíveis na locadora.

### **Atributos:**

Herda os atributos da classe Produtos.

- (String) estilo: informa o estilo do álbum de música em questão.
- (String) autor: informa o autor do álbum, podendo definir se mais de um estão envolvidos.

### **Métodos:**

Herda os métodos da classe Produtos e possui métodos construtores, getters, setters e toString().

## **+ Jogos**

É uma subclasse da classe Produtos e que também serve como superclasse para os jogos de tabuleiro e videogame, definindo um atributo compartilhado por essas duas classes.

### **Atributos:**

Herda os atributos da classe Produtos.

- (String) numJogadores: informa o número de jogadores.

### **Métodos:**

Herda os métodos da classe Produtos e possui métodos construtores, getters, setters e toString().

## **+ Tabuleiros**

É uma subclasse que estende a classe Jogos e que define os atributos específicos para os jogos de mesa disponíveis na locadora.

### **Atributos:**

Herda os atributos das classes Produtos e Jogos.

- (String) tipo: informa se o jogo de mesa envolve um tabuleiro, cartas ou é de outro tipo.
- (String) marca: informa a marca que produziu o jogo.

### **Métodos:**

Herda os métodos das classes Produtos e Jogos e possui métodos construtores, getters, setters e toString().

## **+ Videogames**

É uma subclasse que estende a classe Jogos e que define os atributos específicos para os videogames disponíveis na locadora.

### **Atributos:**

Herda os atributos das classes Produtos e Jogos.

- (String) genero: informa o gênero do jogo em questão.
- (String) plataforma: informa para qual tipo de plataforma o jogo foi feito (console ou computador).
- (String) desenvolvedor: informa o nome da equipe que desenvolveu o jogo.

### **Métodos:**

Herda os métodos das classes Produtos e Jogos e possui métodos construtores, getters, setters e toString().

## 4.4. Armazenamento de informações dos produtos

### + Estoque

É a classe que armazena as informações de todos os Produtos da locadora, mantendo listas estáticas de cada tipo de produto que são iniciadas por meio do arquivo “estoque.txt” e que sempre são atualizadas com base nas modificações realizadas pelos clientes ou funcionários, armazenando essas mudanças no arquivo. Também é do estoque que o pedido recebe os produtos que o cliente coloca no carrinho.

#### Atributos (todos estáticos):

- (static final String)arquivo: define o nome do arquivo que será lido, nesse caso o “estoque.txt”.
- (static ArrayList<Filmes>) listaFilmes, (static ArrayList<Musicas>) listaMusicas, (static ArrayList<Tabuleiros>) listaTabuleiros e (static ArrayList<Videogames>) listaVideogames: listas que mantêm todos os objetos de seus respectivos tipos, os quais podem ser substituídos, adicionados ou removidos quando modificações são realizadas.
- (static ArrayList<Produtos>) listaProdutos: uma lista que reúne os objetos das outras quatro e que é usada para ordená-los e apresentá-los na tela do gerente.
- (static int) cont: uma variável que é usada para definir o código do próximo produto a ser adicionado.

#### Métodos (além dos métodos getters e setters, sendo todos estáticos):

- + (static void) addFilme(Filmes), (static void) addMusica(Musicas), (static void) addTabuleiro(Tabuleiros), (static void) addProduto(Produto): adicionam um item a suas respectivas listas.
- + (static Produtos) getProdutoPorCodigo(int): retorna um objeto Produtos com base em no código do produto.
- + (static void) salvarProduto(Produtos): escreve as informações do produto recebido no final do “estoque.txt”.
- + (static ArrayList<Produtos>) atualizarLista(): atualiza e ordena a lista de produtos geral com base nas outras, retornando-a.
- + (static void) reescreverEstoque(): usa a lista de produtos geral para reescrever o “estoque.txt” por completo, salvando qualquer informação modificada.
- + (static ArrayList<String>) lerEstoque(): retorna uma lista de Strings com as informações de cada produto.
- + (static ArrayList<Produtos>) loadEstoque(): após chamar o lerEstoque(), usa as Strings recebidas para reconstruir os objetos dos produtos e suas listas, como também prepara o código para o próximo produto.

## 4.5. Realização dos aluguéis e armazenamento de suas informações

### + Pedido

É a classe que coordena os pedidos feitos na locadora, armazenando um mapa que relaciona cada código de pedido ao seu objeto Alugar correspondente, como também guarda o pedido que está sendo preparado pelo cliente.

#### Atributos:

- (static final String) arquivo: é definido sempre como “pedidos.txt”, que é o arquivo em que as informações dos pedidos são armazenadas.
- (static int) codigoPedido: usado para enumerar cada pedido, começando por 1 e sempre aumentando à medida que mais pedidos são feitos ou encontrados no arquivo.
- (static ArrayList<Produtos>) pedidoAtual: é a lista que armazena os produtos do pedido sendo feito pelo cliente no aplicativo.
- (static HashMap<Integer, Alugar>) mapAlugueis: faz a relação do código de cada pedido com seu aluguel correspondente.

#### Métodos:

A classe possui os métodos construtores, os métodos getters e setters como static e toString().

- + (static void) addPedido(String, String): recebe como parâmetros a forma de pagamento e o código do caixa, e é chamado quando um pedido é finalizado, salvando o pedido atual do cliente ao mapa de aluguéis e esvaziando o pedido atual.
- + (static void) replaceAluguel(int, Alugar): recebe como parâmetros o código de um pedido e um objeto de aluguel. É usado para adicionar ou substituir um pedido na lista de aluguéis, geralmente quando o status de um produto precisa ser atualizado.
- + (static Alugar) getAluguelPorCodigo(int): recebe como parâmetro o código do pedido e retorna o aluguel correspondente a esse código.
- + (static void) salvarAluguel(Alugar): recebe como parâmetro um aluguel e o escreve no arquivo “pedidos.txt”.

+ (static HashMap<Integer, Alugar>) loadAlugueis(): tem a função de ler o arquivo “pedidos.txt” e retornar um HashMap com base nas informações salvas, sendo chamado para carregar o histórico de pedidos ao iniciar o aplicativo, entrar na área do cliente ou do caixa, e para confirmar que a lista de pedidos está atualizada.

+ (static void) reescreverAlugueis(): tem a função de reescrever por completo o arquivo com as informações do mapa de pedidos, para salvar as informações caso alguma mudança tenha sido feita.

## + Alugar

É uma classe associativa responsável por reunir, em um objeto, cada produto alugado a seu respectivo cliente, o caixa que realizou o pedido e salvar outras características do aluguel

### Atributos:

- (int) codigoPedido: por meio desse atributo é possível ligar um aluguel a um pedido.
- (Date) dataPedido: é o atributo que guarda a data em que o aluguel foi feito.
- (Date) dataDevolucao: é o atributo que guarda a data máxima em que o produto deve ser devolvido.
- (String) pagamento: caracteriza o tipo de pagamento feito no aluguel.
- (String) clienteCPF: conecta o aluguel a um cliente por meio do CPF do cliente.
- (String) caixaCodigo: guarda a informação de qual caixa realizou o processo de aluguel por meio de seu código.
- (ArrayList<Produtos>) listaProdutos: armazena a lista de produtos alugados, no aluguel associado.
- (ArrayList<Status>) listaStatus: armazena a lista dos Status dos produtos alugados no aluguel associado, representando se o produto está alugado, foi devolvido ou se sua devolução está atrasada.

### Métodos:

A classe possui dois métodos construtores que reúnem as informações necessárias, como também os métodos getters e setters e toString().

- + Status getProdutoStatus(int): utiliza o código de cliente informado para retornar o status dele no aluguel em questão.
- + (void) setProdutoStatus(int, Status): ao receber o código de um produto, define seu status para aquele informado nos parâmetros.

## + Status (enumerador)

É um enumerador com os valores “ALUGADO”, “DEVOLVIDO” e “ATRASADO”, que são usados para determinar o estado de cada produto de um pedido.

## 5. Telas

### 5.1. Tela Principal

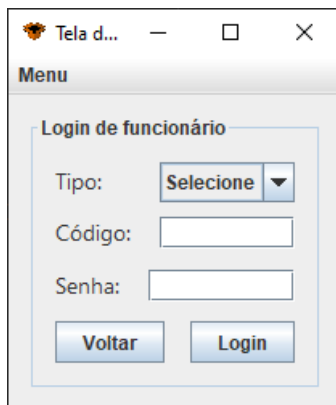
É a tela que aparece quando o aplicativo é iniciado, apresentando o nome e a logo da locadora de botões que o usuário pode usar para navegar pelo programa. Sendo que, desses botões, o primeiro leva à tela da Loja, o segundo leva para a tela de Login do Cliente ou direto para a Área do Cliente caso ele já tenha feito login, o terceiro leva à tela de Login do Funcionário e o último fecha o programa.





## 5.2. Login do Funcionário

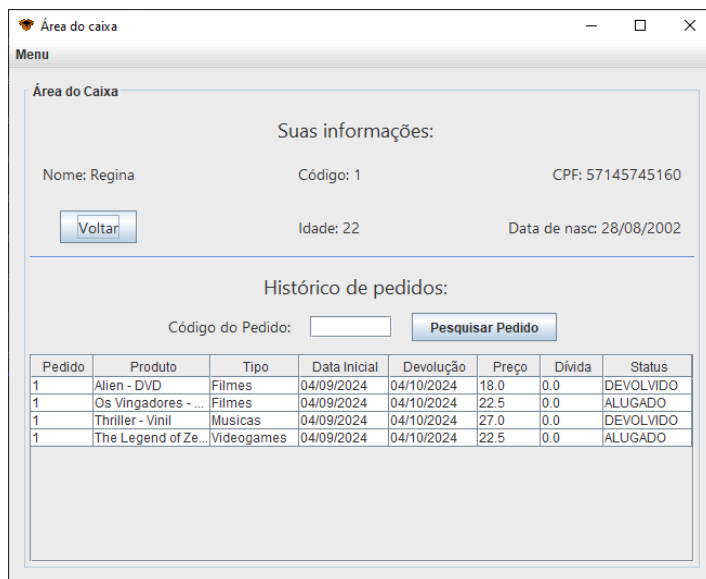
Ao tentar acessar a área do funcionário essa tela aparece, permitindo que o funcionário faça login ao informar seu cargo (caixa ou gerente), código de funcionário e senha. O login só é feito caso o funcionário tenha sido previamente cadastrado no sistema. Caso o código informado não exista ou o tipo ou senha informados estejam incorretos, uma janela pop-up informará isso ao usuário para que ele possa corrigir seus dados. Ao fazer login como caixa, o funcionário é redirecionado para a Área do Caixa, enquanto ao fazer login como gerente, o redireciona para a Área do Gerente.



A interface de login para funcionários. No topo, há uma barra de menu com o ícone de uma caixa e o texto "Tela d...". Abaixo, o título "Menu" é seguido por "Login de funcionário". O formulário contém um campo "Tipo:" com uma lista suspensa mostrando "Selecione", um campo "Código:" e um campo "Senha:". Na base, há dois botões: "Voltar" e "Login".

## 5.3. Área do Caixa

Esta tela mostra os dados do caixa que fez login, como também uma tabela contendo o histórico de pedidos efetuados por ele. Cada linha da tabela corresponde a um produto de um pedido, e contém as informações do aluguel. O caixa pode pesquisar pelo código do pedido para mostrar na tabela apenas os produtos de um mesmo pedido.



A interface da Área do Caixa. No topo, há uma barra de menu com o ícone de uma caixa e o texto "Área do caixa". Abaixo, o título "Menu" é seguido por "Área do Caixa". O formulário mostra "Suas informações:" com campos para "Nome: Regina", "Código: 1", "CPF: 57145745160", "Idade: 22" e "Data de nasc: 28/08/2002". Há um botão "Voltar". Abaixo, o título "Histórico de pedidos:" é seguido por um campo "Código do Pedido:" e um botão "Pesquisar Pedido". A tabela abaixo mostra o histórico de pedidos.

Pedido	Produto	Tipo	Data Inicial	Devolução	Preço	Dívida	Status
1	Alien - DVD	Filmes	04/09/2024	04/10/2024	18.0	0.0	DEVOLVIDO
1	Os Vingadores - ...	Filmes	04/09/2024	04/10/2024	22.5	0.0	ALUGADO
1	Thriller - Vinil	Musicas	04/09/2024	04/10/2024	27.0	0.0	DEVOLVIDO
1	The Legend of Ze...	Videogames	04/09/2024	04/10/2024	22.5	0.0	ALUGADO

## 5.4. Área do Gerente

Essa tela após um gerente realizar login e é dividida em duas abas: uma para fazer o controle do estoque e outra para fazer o controle da equipe.

O controle da equipe é feito por meio de uma tabela em que cada linha corresponde a um dos funcionários cadastrados no sistema. O gerente pode, a partir disso e de campos de texto, adicionar um funcionário, visualizar e editar os dados de cada um deles, bem como pesquisar por quaisquer campos que ele deseje, para que a tabela só mostre os funcionários cujos dados contêm aqueles especificados nos campos de busca.

De modo parecido, o controle do estoque é feito por uma tabela que contém as principais informações de cada produto, os quais podem ser visualizados com mais detalhes ao serem selecionados. Além disso, a tela possui botões para realizar a adição de novos produtos, a pesquisa por produtos com as informações desejadas, a edição de detalhes dos produtos e a exclusão permanente de produtos do estoque.

**Área do gerente**

Menu

Área do Gerente

Equipe | Estoque

Novo funcionário | Pesquisar | Editar

Total de caixas: 2      Total de gerentes: 1

Código:       CPF:       Tipo:

Nome:       Data de nasc.:       Gerente:

Voltar | Confirmar | Cancelar

Nome	CPF	Data de Nascime..	Código	Cargo	Gerente
Phil	30607579412	1/1/2000	0	Gerente	--
Regina	57145745160	28/08/2002	1	Caixa	Phil
Daniel	98887627029	13/03/2003	2	Caixa	Phil

**Área do gerente**

Menu

Equipe | Estoque

Novo produto | Pesquisar | Editar | Excluir

Total de produtos: 28      Tipo:       Produtos desse tipo: --

Código:       Faixa etária:       Preço:

Nome:       Ano:       Disp./Alugados:

Voltar | Confirmar | Cancelar

Código	Tipo	Nome	Faixa	Ano	Preço	Disp./Alugados
1	Tabuleiros	Dungeons & ...	14 anos	2011	25.0	5/0
2	Musicas	Thriller - Vinil	Livre	1982	30.0	4/0
3	Filmes	De Volta Para ...	Livre	1985	20.0	7/0
4	Musicas	Worlds - DVD	Livre	2014	25.0	5/0
5	Musicas	Elis & Tom - V...	Livre	1974	30.0	9/0
6	Tabuleiros	Coup	9+	2012	20.0	7/0
7	Videogames	Celeste	Livre	2018	25.0	8/0
8	Videogames	Super Mario B...	Livre	1985	30.0	9/0
9	Videogames	Shadow of the...	A13	2005	25.0	7/0
10	Videogames	The Legend o...	A10	2017	25.0	8/1

## 5.5. Login do Cliente

Essa tela é utilizada para fazer o cadastro ou o login do cliente e armazenar suas informações no arquivo “clientes.txt”. Quando ela é aberta o usuário tem 3 opções para selecionar: se ele deseja se cadastrar, se quer continuar como visitante ou se já é cadastrado. Assim, ao selecionar uma dessas informações, os espaços de texto aparecem para o usuário preencher com o que for necessário. Se estiver correto, esse usuário pode seguir para a área do cliente ou para a loja, de forma que ele fica salvo no aplicativo como um usuário logado. Caso o cliente entre em uma conta que possui dívida, ele fica impossibilitado de ir para a loja até pagar a dívida.

**Tela de login**

Menu

Cadastro de Cliente

CLIENTES CADASTRADOS GANHAM DIVERSOS BENEFÍCIOS  
INCLUINDO DESCONTO EM TODOS OS PRODUTOS!

Você quer se cadastrar?

Quero me cadastrar

Nome:

CPF:

Data de nascimento:

Senha:

Voltar | Cadastrar-se | Entrar e ir à loja

## 5.6. Área do Cliente

Essa tela tem como propósito o cliente ter controle de suas informações e dos produtos que ele alugou. Ao ser acessada, o usuário pode ver suas seguintes informações: nome, data de nascimento, CPF, idade, dívida e número de produtos alugados. Além disso, a tela possui uma tabela com o histórico de pedidos do cliente, com as seguintes informações de cada produto alugado: número do pedido, nome do produto, tipo do produto, data de aluguel, data de devolução, preço, dívida por atraso e status do aluguel do produto. Sendo que também é por essa tabela que o cliente pode devolver o produto para a loja e pagar uma possível multa. Por fim, o cliente também pode ir para a loja caso não tenha dívida, deslogar da conta ou ir para a tela Editar Cliente.

A interface 'Área do cliente' apresenta uma barra de menu no topo. Abaixo, a seção 'Suas informações:' contém campos para Nome (João Gabriel), Data de nasc. (17/05/1998), CPF (111.111.111-11), Idade (26), Dívida (R\$ 0.00) e Produtos Alugados (2). Há botões para 'Deslogar', 'Editar' e 'Ir para a loja'. À direita, há uma ilustração de uma caixa aberta com uma explosão amarela saindo. Abaixo, a seção 'Histórico de Aluguel:' possui um formulário para 'Forma de pagamento:' com uma lista suspensa 'Selecione' e um botão 'Devolver'. Uma tabela exibe os dados dos pedidos:

Pedido	Produto	Tipo	Data Inicial	Devolução	Preço	Dívida	Status
1	Alien - DVD	Filmes	04/09/2024	04/10/2024	18.0	0.0	DEVOLVIDO
1	Os Vingadores - ...	Filmes	04/09/2024	04/10/2024	22.5	0.0	ALUGADO
1	Thriller - Vinil	Músicas	04/09/2024	04/10/2024	27.0	0.0	DEVOLVIDO
1	The Legend of Ze...	Videogames	04/09/2024	04/10/2024	22.5	0.0	ALUGADO

## 5.7. Editar Cliente

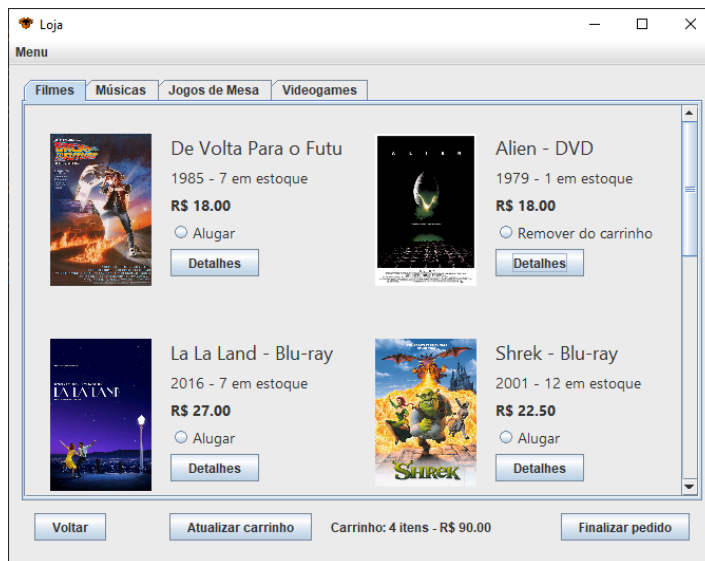
Essa tela só é acessível por meio da Área do Cliente e tem como funcionalidade alterar os dados de um cliente pré-existente, sendo possível alterar seu nome, data de nascimento ou senha que estavam previamente cadastrados, sem afetar seus itens alugados e sendo impossível modificar seu CPF e sua dívida.

A interface 'Editar perfil' possui uma barra de menu. A seção 'Editar Informações' contém campos de entrada para 'Nome:' (João Gabriel), 'Data de Nasc.:' (17/05/1998) e 'Senha:'. Abaixo, há botões para 'Voltar' e 'Salvar'.

## 5.8. Loja

É a tela que apresenta todos os produtos que a locadora oferece, sendo dividida em quatro abas, uma para cada tipo de produto: filmes, músicas, jogos de mesa e videogames. Essas abas apresentam os itens presentes no estoque, os quais podem apresentar imagens de suas capas caso tenham títulos que coincidam com o nome de certos produtos, sendo que há um limite de 6 itens para a aba de jogos de mesa e um de 10 itens para as outras. Navegando pelo catálogo, o usuário

pode clicar nos botões de detalhes para ver uma descrição completa dos produtos e pode usar o botão de atualizar carrinho para adicionar ou remover itens de seu carrinho com base no que ele selecionou, sendo que o número de itens do carrinho e o valor total da compra aparecem na parte de baixo da tela. Se não houver nenhum item selecionado, o usuário pode seguir para a tela de Finalizar Pedido.



## 5.9. Finalizar Pedido

Nessa tela é realizada a finalização do pedido, com áreas para confirmar qual usuário está realizando a compra, quais itens estão no carrinho, qual forma de pagamento será usada e qual caixa confirmou o aluguel (considerando que o aplicativo seria utilizado em uma loja física).

Na parte de identificação da tela, o cliente pode selecionar se quer se cadastrar, se já é cadastrado ou se quer continuar como visitante e preencher suas informações para não ter o trabalho de passar pela área de Login do Cliente antes de ir para a loja.

Pela visualização do carrinho, o usuário pode realizar mudanças de última hora ao seu pedido, podendo retirar itens que ele escolher.

Na área de finalização, são apresentados os detalhes do aluguel e o usuário pode escolher sua forma de pagamento (de maneira simplificada) e qual caixa foi responsável por confirmar o pedido (que no contexto de uma loja física, seria quem liberaria os produtos que o cliente alugou).

Se todas as informações estiverem corretas, o cliente pode finalizar a compra, o que o levará para a Área do Cliente, já com suas informações e o novo aluguel em seu histórico.

A interface 'Pedido' é dividida em três seções principais: Identificação, Carrinho e Finalização.

**Identificação:**

Já possui cadastro? Selecione uma opção

Nome:

CPF:

Data de nasc:

**Carrinho:**

Nº	Título	Preço
1	Alien - DVD	R\$ 18.00
2	Os Vingadores - Blu-ray	R\$ 22.50
3	Thriller - Vinil	R\$ 27.00
4	The Legend of Zelda: Breath of the ...	R\$ 22.50

**Finalização:**

Usuário atual: João Gabriel

Forma de pagamento: Cartão

Caixa: 1-Regina

Data atual: 04/09/2024

Filial: UnB - Darcy Ribeiro

Data máx. de devolução: 04/10/2024

Valor total: R\$ 90.00