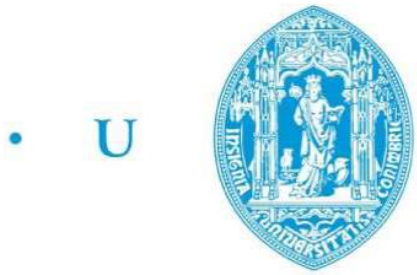


FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Programação Orientada aos Objetos

Saulo José Piccirilo Mendes - 2021235944

Johnny Alexis Lopes Fernandes – 2021190668

7 de dezembro de 2022

Índice

Introdução

O presente relatório tem o objetivo de explicar o funcionamento do projeto desenvolvido para a cadeira de Programação Orientada aos objetos do ano letivo 22/23. O projeto utiliza os conceitos de *data* e *methods* que interagem entre si, através da linguagem java, para atender as necessidades da empresa fictícia StarThrive. A aplicação desenvolvida pretende facilitar a gestão de empresas a cargo da StarThrive, organizando-as em diferentes tipos. Ademais, a aplicação utiliza diferentes ferramentas para reduzir o tempo e os esforços antes necessários para administrar as empresas. Uma das principais características do projeto é o conceito de *user-friendly,* que facilita a utilização da aplicação a todos os utilizadores, graças a interface gráfica (GUI).

Ao longo do trabalho serão também abordadas algumas questões nomeadamente as dificuldades encontradas ao longo do desenvolvimento do projeto bem como as intenções e engenhos no desenvolvimento de certas classes, bem como para a estrutura geral do projeto mencionado supra.

Classes relacionada com a estrutura de dados

A estrutura de dados que prevê a organização das empresas administradas pela StarThrive organiza-se da seguinte forma, tendo um total de 10 classes, das quais 4 são abstratas (e que se encontram a itálico em consonância com o diagrama UML em anexo).

- *Empresa*

A classe principal, ou superclasse, chamada "empresa" possui os atributos comuns: nome, distrito, coordenadas e tipo. Os métodos abstratos "calcularReceitaAnual" e "calcularDespesaAnual" são comuns a todas as subclasses, porém possuem execuções diferentes consoante as características de cada subclasse. O único método concreto, para além dos setter e getters, "calcularLucro", retorna o valor do lucro de cada empresa, ou seja, a diferença entre a receita anual e a despesa anual. O getter mais importante desta classe, o getTipo, é implementado pelas subclasses retorna um inteiro diferente para cada tipo que compõe o projeto.

Faz todo o sentido ter uma classe chamada "empresa" pois o intuito deste projeto é a gestão/administração de empresas a cargo da StarThrive. Todas as classes que se seguem dentro do presente capítulo dizem respeito a classes hierarquicamente inferiores à classe "empresa".

- *Restauração*

A subclasse abstrata restauração, que é uma extensão da classe "empresa", possui os atributos: número de empregados de mesa, custo do salário médio anual por empregado de mesa, e o número médio de clientes por dia. Os métodos constituintes desta classe são somente o construtor e os setters e getters. Nesta classe em particular, como, constante o enunciado, <u>todas</u> as classes hierarquicamente inferiores possuem o atributo relativo ao número médio de clientes, fez todo o sentido colocar este atributo na classe hierarquicamente superior, ou seja, a classe "empresa".

- Café

A primeira classe concreta do trabalho (para a qual existem instâncias de objetos a serem criadas pelo software) - a classe café - contém os atributos intrínsecos das empresas administradas pela StarThrive que são deste tipo, ou seja, o número médio de cafés que vendem por dia e o valor médio de faturação anual por café vendido por dia.

Nesta classe, a despesa anual é calculada através do produto entre o número de empregados e a média de salário anual. Já a receita anual é calculada através do produto entre a média de cafés vendidos diariamente e o valor médio de faturação anual por café vendido por dia. Estes valores são, como veremos mais à frente, úteis na apresentação dos dados tanto na tabela da "JanelaOperacoes" da interface gráfica bem como para os dados estatísticos na "JanelaEstatisticas". Da mesma forma, em classes concretas dos diferentes tipos de empresa, há também métodos úteis e com o mesmo propósito.

- Pastelaria

A classe "Pastelaria" contém os atributos: o número de bolos que vendem por dia e o valor médio de faturação anual por bolo vendido por dia. A despesa anual é calculada através do produto entre o número de empregados e a média de salário anual. A receita anual é calculada através da média de bolos vendidos diariamente X valor médio de faturação anual por bolo vendido por dia. Esta classe é também útil para a criação de novas instâncias da classe, onde serão guardados os dados exclusivos de cada pastelaria.

- *Restaurante*

A classe abstrata restaurante, que é uma extensão da classe hierarquicamente superior - Restauração - possui o atributo número de dias de funcionamento por ano. Além disso possui apenas os getters e setters para esse mesmo atributo.

A classe Restaurante é abstrata pois é, consoante o enunciado, uma categoria geral e da qual derivam duas classes de restaurantes – uma de restaurantes de comida rápida e outra de gastronomia local.

- Restaurante Local

A classe “Local” é uma extensão da classe “Restaurante”, e possui os atributos: o número de mesas interiores, número de mesas na esplanada, custo de licença anual por mesa de esplanada e pelo valor médio de faturação de cada mesa por dia. A despesa anual é calculada pela soma do produto entre o número de empregados e a média de salário anual com o produto entre o número de mesas na esplanada e o custo de licença anual por mesa de esplanada. Já a receita anual é dada pelo produto entre a soma do número de mesas interiores mais o número de mesas esplanada com o valor médio de faturação de cada mesa por dia e o número de dias de funcionamento por ano.

- Restaurante Fast-Food

A classe “Fastfood” é uma extensão da classe Restaurante, e possui os atributos: o número de mesas interiores, o valor médio de faturação de cada mesa por dia, o número médio diário de clientes *drive-thru*, e o valor médio diário de faturação por cada cliente *drive-thru*. A receita anual é calculada através do produto entre o número de mesas interiores, o valor médio de faturação de cada mesa por dia mais o número de médio diário de clientes drive-thru, o valor médio de faturação por cada cliente drive-thru, e o número de dias de funcionamento por ano . Já a despesa anual é calculada através do produto entro o número de empregados e a média de salário anual.

Neste caso em particular, optou-se por não se colocar o número de mesas interiores na classe mãe “Restaurante” uma vez que à diferença da classe “Fastfood”, a classe “Local” também possui mesas de esplanada – por uma questão de organização e leitura de código, optou-se assim por coloca-los nas classes a que dizem respeito.

- *Mercearia*

A classe abstrata mercearia, que é uma extensão da classe Empresa, possui o atributo: o custo anual de limpeza. Esta classe é similar em utilidade à classe Restaurante, pois tanto “Restaurante” como “Mercearia” são as classes hierarquicamente diretamente abaixo da classe “Empresa”.

- Frutaria

A classe frutaria, que é uma extensão da Mercearia, possui os atributos: o número de produtos, e o valor médio de faturação anual por produto. A despesa anual consiste no valor do custo anual de limpeza do estabelecimento e receita anual obtém-se através do produto entre o número de produtos e o valor médio de faturação anual por produto.

- Mercado

A classe mercado, que também é uma extensão da classe Mercearia, possui os atributos: o tipo (min, super, hiper), pela área de corredores (m^2), pelo valor médio de faturação anual por m^2 . A despesa anual consiste no valor do custo anual de limpeza do estabelecimento e receita anual obtém-se através do produto entre a área de corredores em m^2 X valor médio de faturação anual por m^2 .

- Considerações finais do capítulo

Nesta fase do projeto, que concerne à criação de uma estrutura de dados e seguindo o que foi indicado no enunciado:

- Em determinadas classes, atributos comuns a todas as classes em paralelo/na mesma hierarquia, optou-se por passar o atributo à classe mãe exceto nos casos em que a legibilidade e compreensão do código é melhor quando colocadas nas próprias classes.
- Há quatro classes abstratas. Não são criadas instâncias de forma direta destas classes. Todas os objetos criados são de classes hierarquicamente inferiores e irão buscar certos atributos destas classes abstratas, superiores.

Classes relacionadas com o funcionamento do programa

A aplicação funciona graças a três classes “funcionais” (ou de controlo, se quisermos seguir o modelo MVC), denominadas “GerirEmpresas”, “Escritor” e “Leitor”.

A estas três classes, junta-se a classe “Main” responsável pela inicialização e funcionamento de todo o nosso programa.

- Main

A classe “Main” é a principal classe do programa, uma vez que é ela que faz tudo funcionar. Além do método “main” natural de qualquer classe Main, existem ainda os métodos “booting” e “display”. Estes métodos são responsáveis, respetivamente, pela leitura e carregamento dos dados (se existir um ficheiro .dat, será carregado, caso contrário irá apoiar-se no ficheiro .txt, com os dados em plaintext) em memória, sendo guardados num ArrayList<Empresas> existente na classe GerirEmpresas. Além deste método, temos ainda também o “display” responsável pela criação da “JanelaPrincipal”, que contém todas as opções que dão acesso a outras áreas da interface gráfica e funcionalidades do programa. A nossa aplicação usa o tema “Nimbus” do Swing. Em caso de inexistência, usa o tema default (“Metal”).

- Gerir Empresas

A classe gerir empresas contém um ArrayList de objetos do tipo Empresa, ou seja, uma lista com todas as empresas sob a administração da StarThrive. Esta classe possui os métodos necessários para fazer todas as operações necessárias (pesquisar, apagar, adicionar). Ademais, possui métodos que também apresentam estatísticas sobre cada tipo de empresa, como: empresa com maior receita anual, empresa com menor despesa anual, empresa com maior lucro, e as duas empresas do tipo restauração com a maior capacidade de clientes por dia.

- Escritor

A classe escritor é responsável por guardar os dados das empresas e todas as alterações feitas pelo programa. Possui um único método que cria um ficheiro "starthrive.dat" que guarda todas as informações essenciais. Esta função é apenas chamada na classe "Main" para a inicialização do programa, caso não haja um arquivo .dat. Dessa forma, será utilizado o .txt para carregamento dos dados e posteriormente feita a gravação dos dados, com o "Escritor" e ainda na classe JanelaCriaEdita, na qual para qualquer criação de empresa ou edição de empresa, serão escritos os dados relativos ao ArrayList<Empresas> da classe GerirEmpresas no arquivo .dat.

- Leitor

A classe leitor é responsável por ler os dados de empresas pré-existentes e prepará-los para serem modificados pelo programa. Esta classe possui dois métodos: um para ler ficheiros .dat e outro para ler ficheiros .txt. O segundo método é executado apenas caso não exista um ficheiro .dat no sistema. Esta classe é apenas e unicamente utilizada na "Main" para a inicialização do programa. Uma vez feita a leitura dos dados em arquivo, todo o restante processo é feito com o ArrayList<Empresas> guardado em memória.

Para não existir qualquer circunstância de corrompimento de dados, serão gravados após qualquer edição. Caso o programa seja fechado, na reabertura, a leitura dos dados é novamente feita e os dados continuarão a ser trabalhados sobre o ArrayList em memória.

- Considerações finais do capítulo

Nesta fase do projeto, que concerne à inicialização do programa e salvaguarda dos dados:

- É sempre necessário um .txt com dados, caso contrário o programa não inicia (poderá ser um .txt vazio) – na presença de um .dat, terá prioridade em relação ao ficheiro .txt.
- Qualquer alteração e/ou edição de qualquer dado presente na base de dados, relativo às empresas, será seguido de uma ação do escritor para guardar.

Interface Gráfica do Utilizador

- Janela principal

A interface gráfica deste projeto foi construída com recurso a GUI Builder do ambiente de desenvolvimento integrado Netbeans. É constituída por um ecrã principal com uma mensagem de introdução, e dois botões que permitem aceder à(s):

i) Lista de empresas e às respetivas operações

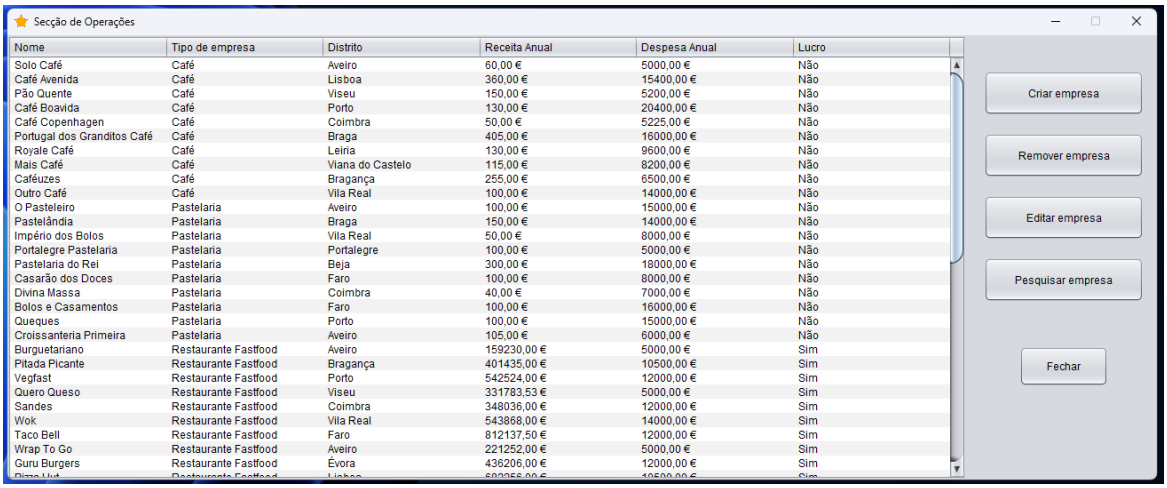
ii) Estatísticas das empresas

Nesta janela, conforme explicado anteriormente no capítulo das classes, há um botão que permite ir para as operação, onde se poderá verificar todos os dados de todas as empresas, organizar os dados, editar, acrescentar ou remover.

Por outro lado, o botão das estatísticas dá-nos acesso a uma janela com várias métricas pedidas, constantes no enunciado, por categoria de empresa, como veremos a seguir.

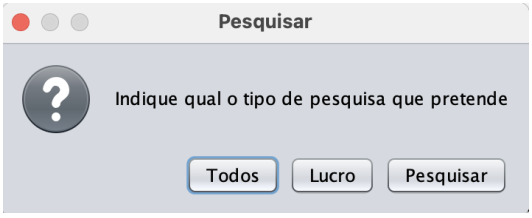
- Janela de operações

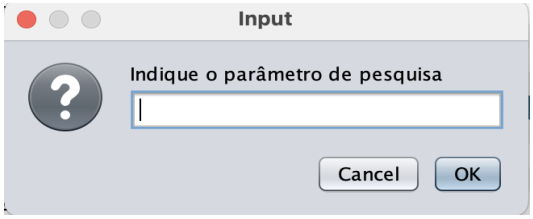
Na janela de operações, temos a nossa tabela de dados, organizada com uma JTable, das quais se optou por mostrar os seguintes dados: Nome da empresa, tipo de empresa, distrito onde a empresa se localiza, a sua receita anual, despesa anual e o lucro (que é calculado pela diferença entre a receita anual e a despesa anual).



Nome	Tipo de empresa	Distrito	Receita Anual	Despesa Anual	Lucro
Solo Café	Café	Aveiro	60,00 €	5000,00 €	Não
Café Avenida	Café	Lisboa	360,00 €	15400,00 €	Não
Pão Quente	Café	Viseu	150,00 €	5200,00 €	Não
Café Boavida	Café	Porto	130,00 €	20400,00 €	Não
Café Copenhagen	Café	Coimbra	50,00 €	5225,00 €	Não
Portugal dos Granditos Café	Café	Braga	405,00 €	16000,00 €	Não
Royale Café	Café	Leiria	130,00 €	9600,00 €	Não
Mais Café	Café	Viana do Castelo	115,00 €	8200,00 €	Não
Caféuzes	Café	Bragança	255,00 €	6500,00 €	Não
Outro Café	Café	Vila Real	100,00 €	14000,00 €	Não
O Pasteleiro	Pastelaria	Aveiro	100,00 €	15000,00 €	Não
Pastelândia	Pastelaria	Braga	150,00 €	14000,00 €	Não
Império dos Bolos	Pastelaria	Vila Real	50,00 €	8000,00 €	Não
Portalegre Pastelaria	Pastelaria	Portalegre	100,00 €	5000,00 €	Não
Pastelaria do Rei	Pastelaria	Beja	300,00 €	18000,00 €	Não
Casarão dos Doces	Pastelaria	Faro	100,00 €	8000,00 €	Não
Divina Massa	Pastelaria	Coimbra	40,00 €	7000,00 €	Não
Bolos e Casamentos	Pastelaria	Faro	100,00 €	16000,00 €	Não
Queques	Pastelaria	Porto	100,00 €	15000,00 €	Não
Croissanteria Primeira	Pastelaria	Aveiro	105,00 €	6000,00 €	Não
Burguetariano	Restaurante Fastfood	Aveiro	159230,00 €	5000,00 €	Sim
Pitada Picante	Restaurante Fastfood	Bragança	401435,00 €	10500,00 €	Sim
Vegfast	Restaurante Fastfood	Porto	542524,00 €	12000,00 €	Sim
Quero Queso	Restaurante Fastfood	Viseu	331783,53 €	5000,00 €	Sim
Sandes	Restaurante Fastfood	Coimbra	348036,00 €	12000,00 €	Sim
Wok	Restaurante Fastfood	Vila Real	543868,00 €	14000,00 €	Sim
Taco Bell	Restaurante Fastfood	Faro	812137,50 €	12000,00 €	Sim
Wrap To Go	Restaurante Fastfood	Aveiro	221252,00 €	5000,00 €	Sim
Guru Burgers	Restaurante Fastfood	Évora	436206,00 €	12000,00 €	Sim

Nesta janela é ainda possível pesquisar todas as empresas para filtrar a tabela, com o botão “Pesquisar empresa”.





Nestas opções, é possível pesquisar por algum parâmetro, seja o nome da empresa, o distrito, o tipo de empresa, ou qualquer outro dado da empresa presente na tabela.

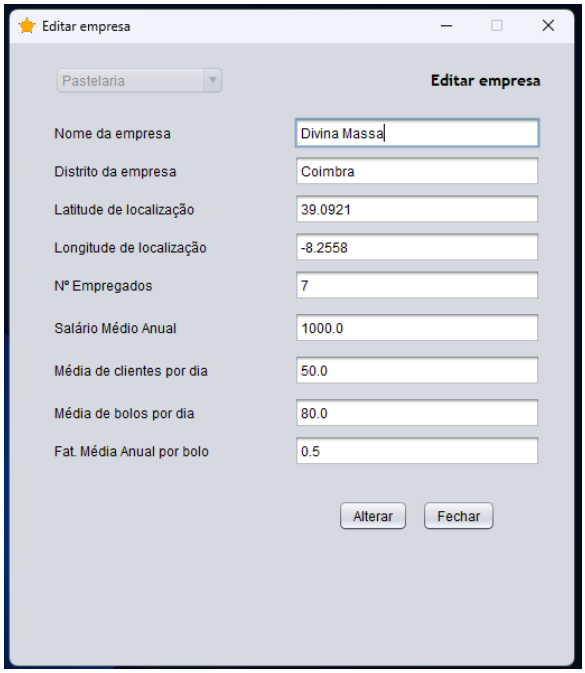
Para reverter a filtragem, existe o botão “Todos” dentro do “Pesquisar empresa”. Além disso, é ainda possível filtrar por lucro, apresentando todas as empresas com “Sim” no campo lucro.

Por outro lado, na secção de “Criar empresa” e “Editar empresa”, o layout/interface é idêntico, havendo apenas uma diferença fulcral. No caso da criação de nova empresa, os campos estarão a branco e a ComboBox é selecionável, para se escolher o tipo de empresa que se pretende.

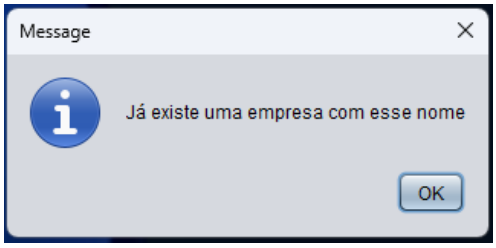
Pelo seu oposto, a opção “Editar empresa” não permite a troca do parâmetro da ComboBox – ou seja, uma empresa que seja do tipo “Café”, continuará a ser até à sua remoção, uma vez que a edição do tipo não é possível.

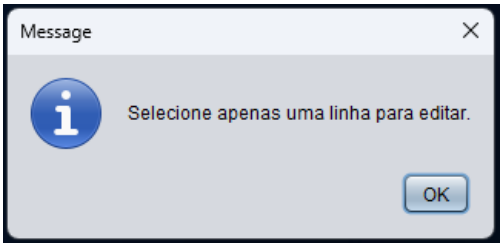
Por outro lado, na janela de “Editar empresa”, todos os dados serão pré-preenchidos, para que o utilizador não tenha de preencher novamente todos os campos. Basta para isso editar apenas os campos que tem interesse em alterar.

No caso da opção de editar, será necessário selecionar uma empresa da lista para editar. Mais do que uma opção selecionada e o programa apresentará uma janela de aviso informando que não pode selecionar mais que uma empresa (não são abertas múltiplas janelas para cada uma das opções selecionadas).

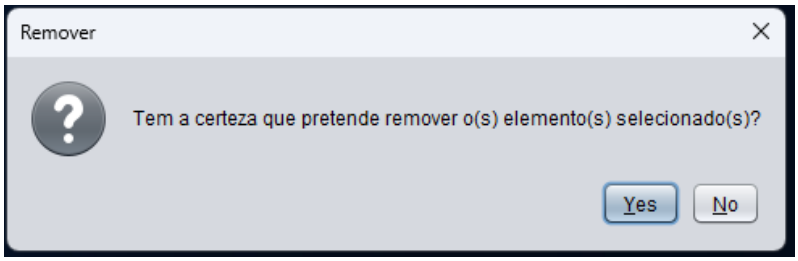


O programa permite apenas uma empresa com determinado nome, sendo impossível editar uma empresa com o nome de outra empresa já existente, bem como criar uma nova empresa com outra empresa de nome igual presente na estrutura de dados.



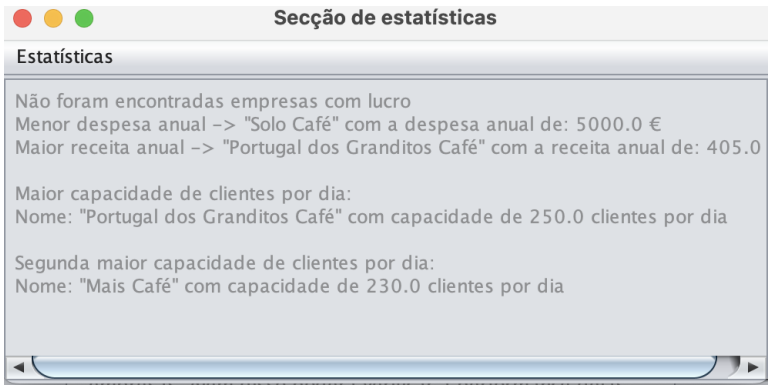


É por fim possível, na janela das operações, remover empresas. As empresas podem ser removidas individualmente ou em lote, originando uma mensagem de confirmação para a remoção das mesmas.



Janela de estatísticas

Por fim, a nossa janela de estatísticas, acessível através da janela principal, permite verificar várias métricas sobre os dados das nossas empresas, nomeadamente a empresa com maior lucro, menor despesa anual, maior receita anual, e capacidade (caso possua número médio de clientes por dia).



Conclusão e considerações finais

O presente projeto levou à necessidade de tomada de decisão perantes diversos cenários de usabilidade, integridade de dados e organização do projeto. Para tal foi desenvolvido um diagrama UML que deverá ser levado em consideração na análise e avaliação deste projeto, tendo sido demonstrada toda a estrutura, hierarquia e divisão dos diversos componentes e áreas deste projeto.

O diagrama UML foi desenvolvido com recurso a uma ferramenta (PlantUML) para o qual se deve referenciar que os símbolos public (+ é uma bola verde) e private (- é um quadrado vermelho) têm uma notação específica nesta ferramenta, para efeitos estéticos. Por outro lado, recorremos também ao IDE Netbeans, tendo sido abordado em aula uma funcionalidade no que concerne ao desenvolvimento GUI. Tanto o Intellij como o Netbeans possuem acesso a uma ferramenta JForms para montagem de GUI.

Decidimos usar estas ferramentas para poder criar uma interface não só mais apelativa, como também organizada e mais profissional. Todo o código gerado foi refatorizado e incluído nas nossas classes existentes. Aconselhamos ainda o acesso ao Javadoc para melhor compreensão de todas as classes existentes no projeto bem como os seus atributos e métodos. Convém notar ainda que por imposição da disciplina na submissão dos arquivos, não usámos packages. Os packages teriam trazido um acréscimo de organização do código-fonte e uma maior simplicidade relacional entre as classes, bem como teria sido mais facilmente demonstrado a utilização do modelo MVC na nossa projeção das diferentes classes. Por fim, mas não menos importante, tentamos seguir no presente projeto todas as boas práticas de programação orientada a objetos conforme lecionado em aula laboratorial.

Consideramos ter chegado a um resultado sem falhas e problemas de performance, cumprindo todos os requisitos, com a certeza de concluirmos o projeto mais enriquecidos no que toca à programação orientada a objetos.