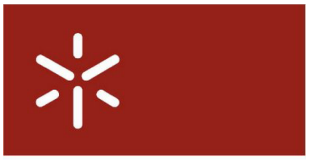
Universidade do Minho

Departamento de Informática



Universidade do Minho

Laboratórios de Informática III

Sistema de Gestão e Consulta de Voos e Reservas

Grupo 62 - Fase 1

Henrique Pereira

Mariana Morais

Tomás Valente

Henrique Morais Pereira A100831
Mariana Filipa Morais Gonçalves A100662
João Tomás Gonçalves de Sousa Carneiro Valente A100540

Novembro de 2023

Conteúdo

1 Introdução

Este projeto foi apresentado no presente ano letivo no âmbito da unidade curricular de Laboratórios de Informática III do curso de Licenciatura em Engenharia Informática, com o objetivo de consolidar os conteúdos teóricos e práticos enriquecendo os conhecimentos adquiridos nas UCs de Programação Imperativa e de Algoritmos e Complexidade.

Nesta primeira fase do projeto tivemos a oportunidade de desenvolver um projeto em larga escala com o objetivo de criar uma aplicação que simule um sistema de Gestão e Consulta de voos e reservas, tentando manter sempre a proteção de dados através do encapsulamento dos mesmos e a utilização eficiente de memória. É também fulcral desenvolver o programa tendo em conta a modularidade(separação do código coerentemente), um códigos reutilizável e uma aplicação de estruturas adequadas para a manipulação dos dados.

Temos em mente com este projeto uma evolução no nosso pensamento crítico tanto na escolha dos métodos de otimização como na escolha de estruturas adequadas ao desafio.

2 Esquema Do Projeto

Sendo uma aplicação de um projeto em larga escala, a aplicação aceita a entrada de quatro ficheiros de texto: um ficheiro com uma listagem de utilizadores, um ficheiro com uma listagem de voos um ficheiro com uma listagem das reservas e ainda um ficheiro com a listagem dos passageiros; Implementou-se uma estrutura semelhante para o tratamento de dados de todos os ficheiros, uma vez que a catalogação se divide em quatro catálogos distintos.

No entanto, de modo mais específico, com o intuito de otimizar o trabalho prático decidimos implementar Hash-Tables de structs.

Esta decisão foi tomada pelo grupo com base nas seguintes características:

- A capacidade de armazenar um alto número de elementos
- Rápida procura
- Facilidade na remoção de elementos
- Eficiência na inserção de elementos

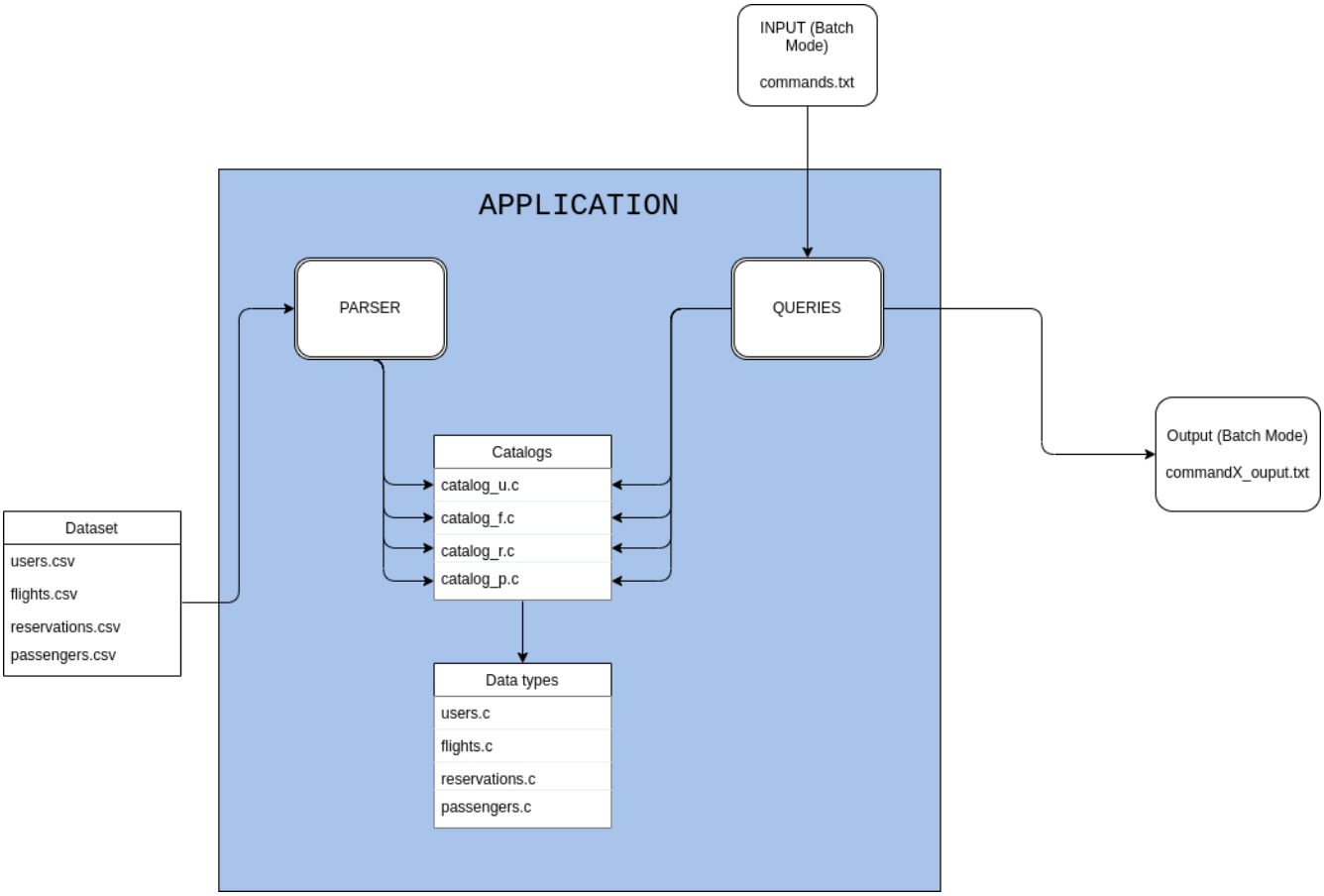
Devemos também referir o uso da Glist uma vez que mantém a ordem dos elementos algo que nos pareceu bastante necessário e útil nas queries apresentadas posteriormente.

2.1 Estrutura de dados

Os dados para a elaboração do trabalho prático estão divididos em quatro ficheiros. Sendo eles:

- **flights.csv**, com voos onde cada linha com a seguinte informação: identificador do voo; companhia aérea; modelo do avião; número de lugares totais disponíveis; aeroporto de origem; aeroporto de destino; data e hora estimada de partida; data e hora estimada de chegada; data e hora real de partida; data e hora real de chegada; nome do piloto; nome do copiloto e observações sobre o voo.
- **passengers.csv**, onde apenas temos a informação do identificador do voo e do identificador do utilizador de cada passageiro.
- **reservations.csv**, sobre as reservas temos a seguinte informação: identificador da reserva; identificador do utilizador; identificador do hotel; nome do hotel; número de estrelas do hotel; percentagem do imposto da cidade (sobre o valor total); morada do hotel; data de início; data de fim; preço por noite; se a reserva inclui pequeno-almoço; detalhes sobre o quarto; classificação atribuída pelo utilizador e comentário sobre a reserva.
- **users.csv**, cada utilizador tem a si associado os seguintes dados: identificador do utilizador; nome; email; número de telemóvel; data de nascimento; sexo; número do passaporte; código do país de residência; morada; data de criação da conta; método de pagamento e estado da conta.

Após analisar os dados que tínhamos para tratar e o que nos era pedido, pensamos no modo de implementação. Posto isto, temos aqui o esquema ilustrativo da nossa aplicação que é o motor e a referência para o desenvolvimento do projeto pretendido .



Esquema ilustrativo da nossa aplicação

3 Estrutura Do Projeto

3.1 Catalogo dos Voos

De modo a tratar os dados que desejamos, criarmos um catálogo para os Voos. É neste módulo que os dados do ficheiro flights.csv são guardados. Os dados são guardados em Hashtables, onde é usado o ID do Voo como chave da mesma. Tendo isto em conta, é realizado o parse do .csv onde abrimos o arquivo, cada linha do arquivo é processada e se validada é inserida na Hashtable usando o ID do voo como chave. Caso contrário segue para o ficheiro .csv de erros correspondente.

3.2 Catalogo dos Passageiros

Neste módulo que os dados do ficheiro passengers.csv são guardados. Os dados são guardados em Hashtables, onde é usado o ID do passageiro como chave da mesma. Tendo isto em conta, é realizado o parse do .csv onde abrimos o arquivo, cada linha do arquivo é processada e se validada é inserida na Hashtable usando o ID do user e do flight como chave. Caso contrário segue para o ficheiro .csv de erros correspondente.

3.3 Catalogo das Reservas

Neste módulo que os dados do ficheiro reservations.csv são guardados. Os dados são guardados em Hashtables, onde é usado o ID da Reserva como chave da mesma. Tendo isto em conta, é realizado o parse do .csv onde abrimos o arquivo, cada linha do arquivo é processada e se validada é inserida na Hashtable usando o ID da reserva como chave. Caso contrário segue para o ficheiro .csv de erros correspondente.

3.4 Catalogo dos Utilizadores

Neste módulo que os dados do ficheiro users.csv são guardados. Os dados são guardados em Hashtables, onde é usado o ID do Utilizador como chave da mesma. Tendo isto em conta, é realizado o parse do .csv onde abrimos o arquivo, cada linha do arquivo é processada e se validada é inserida na Hashtable usando o ID do user como chave. Caso contrário segue para o ficheiro .csv de erros correspondente.

4 Módulos

Um projeto desta envergadura exige diversos módulos de modo a estruturar e proteger os dados. Uma vez que um dos objetivos é a aplicação de encapsulamento, o projeto está implementado em vários destes. Seguimos com uma breve elucidação.

- ***batch.h*** Módulo que coordena a execução das tarefas relacionadas à gestão de dados dos usuários, voos, reservas e passageiros.
- ***catalog_f.h*** Módulo que guarda os dados associados aos voos.
- ***catalog_p.h*** Módulo que guarda os dados associados aos passageiros.
- ***catalog_r.h*** Módulo que guarda os dados associados às viagens.
- ***catalog_u.h*** Módulo que guarda os dados associados aos utilizadores.
- ***date.h*** Módulo para o manuseamento das datas e horas. É definida uma estrutura de modo a conseguir tratarmos dos dados desejados. E após isto são feitas manipulações de modo a comparar e calcular diferenças entre datas. É também aqui que validamos uma data ou não.
- ***flights.h*** Neste módulo, iremos tratar dos voos. É criada uma estrutura de modo a tratarmos os dados da forma mais conveniente que encontramos. Nesta estrutura são guardados os dados dos mesmos de forma a poderem ser usados posteriormente conforme as necessidades que encontramos.
- ***passengers.h*** Este módulo, semelhante ao anterior, é também elaborado com o pensamento e necessidade de tratar dos dados dos passageiros.
- ***reservations.h*** A partir das Reservas, também é necessário aceder e manipular os seu dados de forma a responder ao pedido. Temos também uma estrutura com todos os dados associados às reservas. Neste módulo já temos algum tratamento de dados, tais como a comparação das datas de duas reservas e a ordenação de reservas com base na data do seu início.
- ***users.h*** Como nos anteriores, será criada a estrutura que permite ter todos os dados associados mas, neste caso, aos utilizadores. É também validada alguma informação pertinente.
  ´
- ***queries.h*** Neste módulo, já entramos no desenvolvimento do que fizemos anteriormente. Iremos manipular os nossos catálogos de acordo aos requisitos pretendidos em cada consulta.
- ***validacao.h*** Este breve módulo garante que certos valores do sistema não sejam zero conforme o que o programa necessita.

5 Queries

5.1 QUERY 1

A query 1 tem como objetivo listar, a partir do seu identificador, o resumo de um user de um voo ou de uma reserva.

Para isso primeiro, a função tenta encontrar um user com o ID fornecido, se não encontrar o usuário, tenta encontrar um voo com o mesmo ID, se também não encontrar o voo, tenta encontrar uma reserva com o mesmo ID.

Se um user for encontrado, verifica se a conta do user está ativa, se a conta estiver inativa, escreve uma string vazia no arquivo. Caso contrário, obtém informações adicionais, como o número de voos, o número de reservas e o total gasto, e converte essas informações numa string

5.2 QUERY 3

A query 3 tem como objetivo apresentar a classificação média de um hotel, a partir do seu identificador.

Para tal iremos extrair a lista de todas as reservas do catálogo das reservas, percorrendo a mesma e verificaremos se o ID de cada reserva corresponde ao ID desejado. Caso isto se verifique, o valor da sua classificação será adicionado ao valor total e o número de reservas será incrementado.

Por último iremos proceder ao cálculo da classificação média dividindo o valor total das classificações pelo número total das reservas previamente calculados.

5.3 QUERY 4

A query 4 pretende listar as reservas de um hotel, ordenadas por data de início, com um critério de desempate baseado no identificador da reserva.

De modo a responder a isto, iremos extrair a lista de todas as reservas do catálogo das reservas. Posteriormente, iremos percorrer cada reserva da lista e para cada reserva, o ID dado é associado ao ID de cada hotel adicionado e caso seja igual é adicionado ao nosso novo array.

Após termos o nosso array, vamos ordenar pelas datas das reservas, da mais antiga para a mais recente. Destacando o facto que em caso de empate, é utilizado o identificador da reserva.

5.4 QUERY 5

A query 5 consiste em listar os voos com origem num dado aeroporto, entre duas datas, ordenados por data de partida estimada, da mais antiga para a mais recente.

Para cumprir estes objetivos iremos criar a lista de todas os voos do catálogo de voos. Posteriormente iremos organizar a lista pela data e hora estimada de partida da mais antiga para a mais recente.

De seguida iremos percorrer a lista e verificar se o nome de cada aeroporto corresponde ao nome desejado e se a hora estimada de partida do voo correspondente se encontra dentro das datas limite estabelecidas.

5.5 QUERY 8

Nesta query começamos por listar todas as reservas de modo a que fosse possível selecionar apenas hotéis com o id desejado.

Após isto iremos buscar as datas da reserva e contar o número de noites que a reserva tem durante um período de tempo pré definido pelo input.

Após termos o número de noites reservados num dado hotel num período definido, iremos ver o preço por noite nesse hotel e multiplicar pelo numero de noite, tendo assim a receita que desejamos calcular.

5.6 QUERY 9

De modo a realizar esta query, iremos listar os utilizadores e estes serão alocados num array, após isto iremos percorrer todos os utilizadores e garantir que o tamanho do nome é superior ou prefixo dado.

Após isto iremos retirar do nome o prefixo de igual tamanho ao prefixo dado. Depois estes 2 prefixos são comparados e caso sejam iguais são adicionados ao array.

Nesta fase já temos todos os utilizadores começados pelo prefixo desejado e procedemos à sua ordenação. Esta é realizada peo uma função que compara os nomes e em caso de empate usa o id dos utilizadores.

Tendo assim, nesta fase, a query realizada.

5.7 Testes

Numa fase final, os testes da plataforma fornecida pela equipa docente estavam a dar resultados inconsistentes. De modo a garantir que as alterações efetivamente funcionavam seguimos as instruções do enunciado e fizemos os nosso testes através do Docker. Desta maneira garantimos que o programa compilava e executava ao contrário da informação que tivemos da plataforma algumas vezes.

6 Conclusão

De modo a concluir esta fase inicial, o projeto tem sido bastante desafiante e trabalhoso, no entanto tem sido um bom mecanismo de evolução. Tem sido também um bom motor de busca para novos conceitos que até então não tinham sido despertados totalmente ou da melhor maneira.

Houve uma grande dificuldade na validação, o que gerou alguns conflitos estruturais. A legibilidade e a indentação do código foi algo que acabou por não ser a melhor causando também algumas complicações na comunicação e trabalho de grupo. Para além disso a implementação de queries foi algo que também exigiu muito de nós e que mesmo assim não correspondeu totalmente aos objetivos previsto talvez pela falta de testes realizados ao longo da sua realização.

Contudo, de forma geral, fomos conseguindo realizar o trabalho conscientes que não está o expectável e o que nós projetamos inicialmente.

Posto isto, estes são os aspetos que pretendemos melhorar na segunda fase de forma a conseguir trabalhar melhor para que o resultado também seja melhor. Algo que também queremos melhorar é a gestão de tempo.