Trabalho AEDS III - CRUD Conta Bancária

João Victor Dos Santos[1], Leonardo de Oliveira Carvalho[2], Igor Franco Corrêa[3]

[1]PUC Minas – Pontifícia Universidade Católica de Minas Gerais (PUCMG)
30.535-901 – Belo Horizonte – MG – Brazil

1. Introdução

Com o desenvolvimento dessa aplicação, foi possível ver em prática o funcionamento de um algoritmo que trabalhe com arquivos de texto. Para a elaboração do CRUD das contas e leitura dos arquivos. Para a realização desse trabalho foi utilizado a linguagem de programação java.

2. Desenvolvimento

Para o desenvolvimento da aplicação, foi separado em arquivos para melhor organização˜ e aproveitamento do codigo. Esses arquivos foram: CRUD, Conta, Interface Entidade e Menu.

2.1. Conta

O arquivo Contas, representa a classe que será a entidade/registro do projeto. Nela estão as propriedades de uma Conta, pelo qual são: int idConta, String nomePessoa, String email, String nomeUsuario, String senha, String cpf, String cidade, int transferenciasRealizadas, float saldoConta;

2.2. CRUD

O arquivo de CRUD, armazena todos os métodos que vão gerenciar as operações feitas no arquivo. Elas são: Create, Read, Update, Delete.

2.3. Interface Entidade

```
public class Conta implements Comparable<Conta> {
    protected int idConta;
    protected String nomePessoa;
    protected String email;
    protected String nomeUsuario;
    protected String senha;
    protected String cpf;
    protected String cidade;
    protected int transferenciasRealizadas;
    protected float saldoConta;
```

Figure 1. Imagem retirada do Visual Studio Code

A interface Entidade tem como objetivo determinar os métodos que obrigatoriamente devem ser implementados na classe Conta.

2.4. Menu

O nosso menu foi desenvolvido com o intuito de fazer as chamadas dos metódos do CRUD. Com ela o usuário pode selecionar o que deseja faazer como por exemplo criar ou consultar uma conta, além de poder realizar transferências e consultar seu saldo.

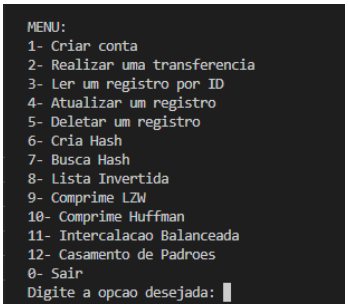

Figure 2. Imagem retirada do Visual Studio Code

2.5. Ordenação

Para realizar ordenação foi se utilizado ordanção externa e Lista invertida.

A ordenação externa é um método de ordenação de dados que é utilizado quando os dados a serem ordenados são muito grandes para caber em memória principal. Ela lê e escreve os dados sequencialmente em disco, em vez de acessá-los aleatoriamente na memória principal.

Uma lista invertida é uma estrutura de dados que armazena uma lista de itens em ordem inversa. Ela é útil em algoritmos de ordenação e busca, pois permite que os itens sejam inseridos, removidos e encontrados rapidamente no início da lista.

2.6. Compreensão

Para realizar compreesão foi se utilizado LZW e Huffman.

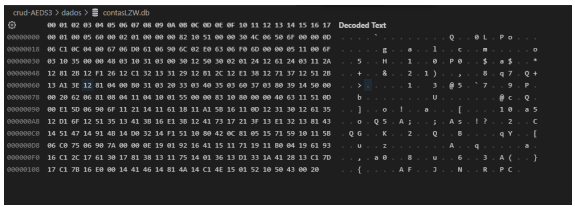

Figure 3. Imagem retirada do Visual Studio Code, pasta ”contasLZW.db” contas comprimidas utilizando algoritimo LZW

O algoritmo LZW funciona criando uma tabela de códigos que mapeia sequências de caracteres para códigos numéricos mais curtos. Esses códigos são então usados para representar as sequências de caracteres no arquivo comprimido, resultando em uma representação mais compacta dos dados.

O algoritmo Huffman funciona criando uma árvore de codificação que mapeia caracteres para códigos binários mais curtos. Os códigos são então usados para representar os caracteres no arquivo comprimido, resultando em uma representação mais compacta dos dados. Quando o arquivo é descompactado, a árvore de codificação é usada para recriar os caracteres originais.

2.7. Criptografia

Para realizar criptogradia estamos utilizando One Time Pad.

Figure 4. Imagem retirada do Visual Studio Code, pasta ”contasHuffman.db” contas comprimidas utilizando algoritimo Huffman

One-time pad (OTP) é um método de criptografia que é considerado totalmente seguro, desde que sejam cumpridas algumas condições específicas. O OTP é um método de criptografia simétrico, o que significa que a mesma chave é usada tanto para criptografar quanto para descriptografar a mensagem.

Figure 5. Imagem retirada do Visual Studio Code

3. Testes e Resultados

Os testes foram feitos em cada procedimento do CRUD, pelo qual e verificado se está certo, conforme o resultado final do arquivo de texto que armazena os dados.

3.1. Método Criar

Para utilizar o método criar e necessário ter a pasta “dados” criada na raiz do projeto. Ao selecionar a opção 1 no menu: “1- Criar conta”, o próximo passo e preencher as informações da conta, da seguinte maneira:

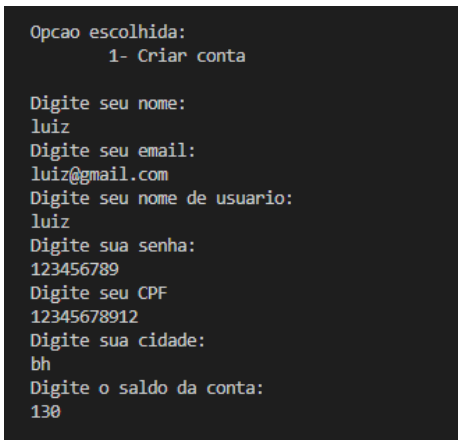

Figure 6. Imagem retirada do Visual Studio Code

Após o time ser criado de maneira correta, o arquivo ”conta.db” ser a possível de se acessar dentro da pasta “dados”, e nele estão armazenadas as informações das contas da seguinte maneira:

3.2. Método Listar

Para fazer a listagem das informações de uma conta, e necessário passar a opção 3 no menu: ”3- Ler um registro por ID”. E depois informar o id da conta que deseja ser achado, o retorno deve ser:

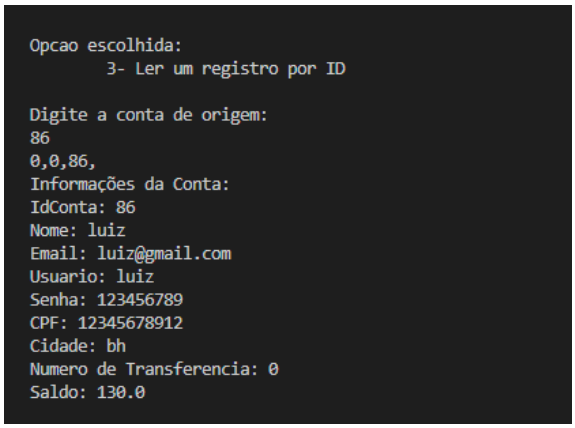

Figure 7. Imagem retirada do Visual Studio Code

3.3. Método Atualizar

Para atualizar os dados de uma conta, no menu e selecionado a opção 4 no menu: ”4- Atualizar um registro” e após isso e indicado o id da conta e as suas novas informações. Ao completar as informações, caso tudo dê certo e printado os novos dados da conta, da seguinte forma:

3.4. Método Excluir

O método excluir pode ser utilizado selecionando a opção 5: ”5- Deletar um registro” no menu e passando o id da conta que deseja ser excluído, dessa forma:

4. Conclusão

Podemos concluir que com esse trabalho, que desenvolver aplicações com memória secundaria e muito interessante e necessário, pois garantimos consistência e persistência dos dados. Fizemos um sistema de CRUD simples, que pode nos ajudar a entender como funciona alguns sistemas que existem no mercado de trabalho e sistema para gerenciamento de equipamentos, em que um certo funcionário pode realizar ações de criação, atualização, pesquisa e deletar certo equipamento. Então, por meio desse trabalho aprendemos as operações básica em um sistema que vai armazenando cada dado em memória secundária. Outra vantagem de se trabalhar em memória secundária e que se por algum motivo ocorresse falta de energia durante a execução do programa nenhum dado seria perdido, já que quando for reiniciado será necessário apenas recarregar o arquivo. Por fim, ainda aprendemos muito sobre a linguagem Java em si, que nunca tínhamos usado antes. Java e uma linguagem que lida muito bem com o fluxo de arquivos, e umas das linguagens mais usadas no mercado de trabalho atualmente, por isso e tão importante saber sua sintaxe e ter projetos com ela.