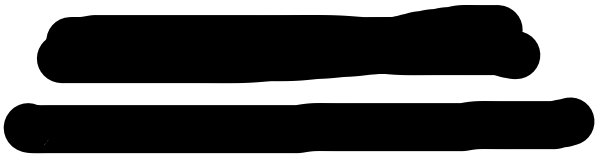

DESENVOLVIMENTO E ANÁLISE DE UM DIAGRAMA DE CLASSES UML PARA UM PROBLEMA PROPOSTO

Itajaí

2024

RESUMO

O relatório a seguir mostra um projeto da disciplina de programação orientada a objeto, com o propósito de mostrar os conhecimentos obtidos em aulas, o projeto tem como base mostrar um problema real feito pelos alunos, especificamente nesse trabalho foi feito pelos alunos Douglas Scheffer Lopes e Edson Luiz Sartori Junior, tendo os requisitos de criar uma implementação de um diagrama de classes UML, criando um problema real e resolver o problema feito por eles mesmo, seguindo os requisitos apresentados pelo Prof. Marcos Carrad.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Nesse relatório pretendemos apresentar um problema real e requisitos propostos pelo professor, entre eles apresentar uma lista de requisitos funcionais da aplicação/problema, ter um diagrama de classe da UML, que contenha no mínimo, 4 classes, apresente herança e algum tipo de composição/agregação. E uma implementação na linguagem Java, atendendo aos requisitos propostos**.**

2 PROBLEMA REAL E REQUISITOS FUNCIONAIS

Um empresário de Itajaí está tendo problemas para criar uma academia (musculação), um dos principais problemas que ele está tendo é como irá criar um sistema para suportar os requisitos que ele pretende fazer, que entre eles é um sistema para os instrutores e para os alunos, portando para resolver esse problema ele chamou programadores da Univali para resolver, perante a chamada os programados mostraram uma lista de funcionalidades, que foi aceito previamente pelo empresário de forma imediata, perante a lista estão as seguintes funcionalidades que pretendem fazem:

Lista de Requisitos Funcionais:

- Sistema de registro: o sistema será onde instrutor e alunos deverão fazer o registro, com as seguintes informações: nome, CPF, data de nascimento, endereço, para professor especificamente ter que ter um id só para ele e para o aluno a mesma coisa.

- Sistema de Treino: o sistema será utilizado pelos instrutores e alunos, o instrutor irá criar os treinos, modificar e poderá remover, e o aluno apenas irá utilizar os treinos que o professor fez, as informações que terá no treino serão, id treino, nome do treino, data do começo e fim do treino, frequência e intensidade e quando criar um treino terá que ter informações especificas, entre elas, o tipo de exercício, número de series, número de repetições, carga e tempo de descanso.

2.1 DIAGRAMA DE CLASSE DO PROBLEMA

O diagrama de classe abaixo demostra como que foi criado com os requisitos propostos:

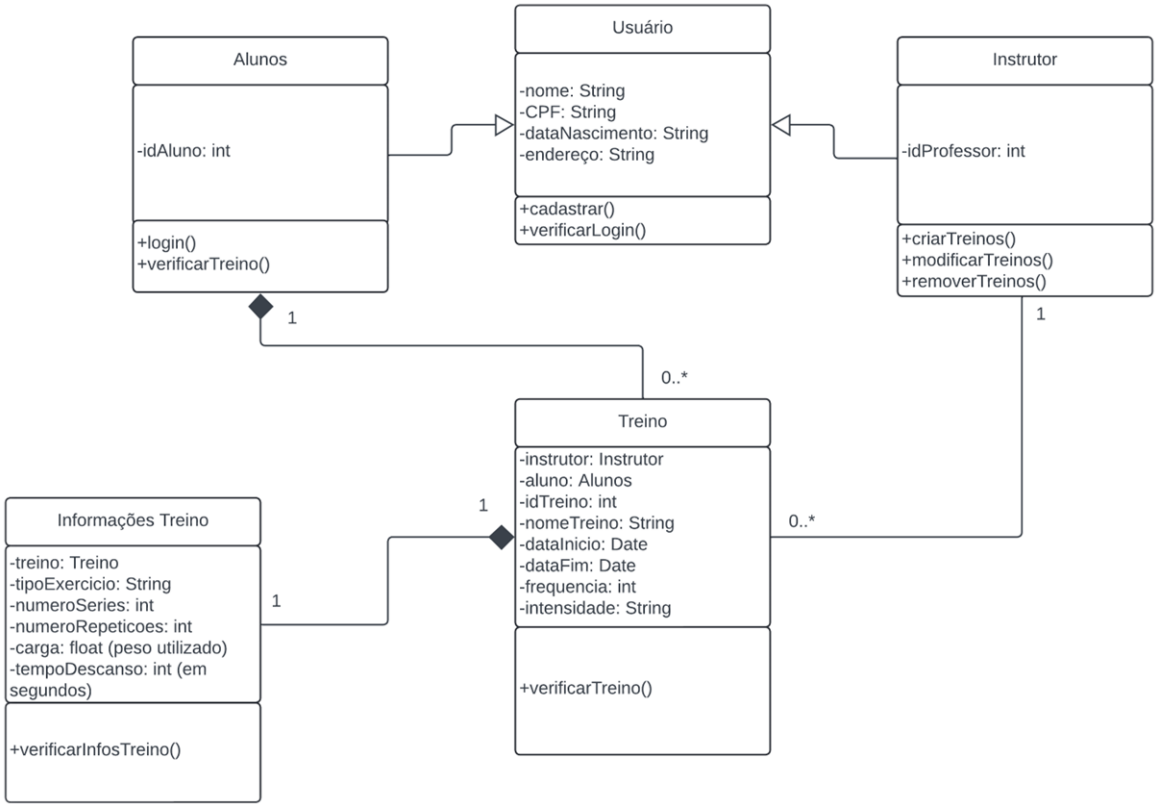

Figura 1 – Fonte: Autor

2.2 IMPLEMENTAÇÃO

A implementação do trabalho foi feita na linguagem Java, usando como IDE o Visual Studio Code, como não podemos demostrar todo o trabalho, iremos mostrar partes da implementação feita.

Uma das implementações que será apresentada como exemplo será a classe Usuario (Figura 2), principalmente ela recebe as variáveis String Nome, String CPF, String dataNascimento e String Endereço, além disso tem os métodos para o instrutor ou usuário se registrar e outro método para verificar login, as outras classes que têm o nome Instrutores (Figura 3) e Alunos (Figura 4) irão basicamente herdar da classe Usuario, com a única diferença é que Alunos e Instrutores tem id diferentes para melhor separação de usuários, além de que as classes Alunos e Instrutores tem seus próprios métodos.

```java
package Java.M1_AcademiaNova;

//import java.util.*;

public class Usuario {
    private String Nome;
    private String CPF;
    private String dataNascimento;
    private String Endereco;

    public Usuario(String n, String cpf, String dn, String e){
        this.Nome = n;
        this.CPF = cpf;
        this.dataNascimento = dn;
        this.Endereco = e;
    }

    public void cadastrar(){
        System.out.println("Usuario " + Nome + " cadastrado com sucesso!");
    }

    public void verificarLogin(String cpf, String nome){
        if(this.CPF.equals(cpf) && this.Nome.equals(nome)){
            System.out.println(x:"Login realizado com sucesso!");
        } else {
            System.out.println(x:"Falha no login. Verifique suas credenciais.");
        }
    }

}
```

Figura 2 – Fonte: Autor

```java
package Java.M1_AcademiaNova;

import java.util.*;

public class Aluno extends Usuario{
    private int idAluno;

    public Aluno(String n, String cpf, String dn, String e, int idA){
        super(n, cpf, dn, e);
        this.idAluno = idA;
    }

    public void login(){
        Scanner sc = new Scanner(System.in);
        System.out.print(s:"Digite seu CPF: ");
        String cpf = sc.nextLine();
        System.out.print(s:"Digite seu Nome: ");
        String nome = sc.nextLine();
        verificarLogin(cpf, nome);
    }

    public void verificarTreinos(List<Treino> treinos){
        System.out.print("Treinos do aluno " + getNome() + ": ");
            for(Treino treino : treinos){
                if(treino.getAluno().getIdAluno() == this.idAluno){
                    System.out.println("Treino: " + treino.getNomeTreino() + " | Intensidade: " + treino.getIntensidade());
                }
            }
    }

```

Figura 3 – Fonte: Autor

```java
package Java.M1_AcademiaNova;

import java.util.*;

public class Instrutor extends Usuario{
    private int idProfessor;

    public Instrutor(String n, String cpf, String dn, String e, int idP){
        super(n, cpf, dn, e);
        this.idProfessor = idP;
    }

    public void criarTreinos(List<Treino> treinos, Treino treino){
        treinos.add(treino);
        System.out.println("Treino " + treino.getNomeTreino() + " criado com sucesso!");

    }

    public void modificarTreinos(Treino treino, String novoNome, Date novaDataInicio, Date novaDataFim, int novaFrequencia, String novaIntensidade){
        treino.setNomeTreino(novoNome);
        treino.setDataInicio(novaDataInicio);
        treino.setDataFim(novaDataFim);
        treino.setFrequencia(novaFrequencia);
        treino.setIntensidade(novaIntensidade);
        System.out.println("Treino " + treino.getNomeTreino() + " modificado com sucesso!");
    }

    public void removerTreinos(List<Treino> treinos, Treino treino){
        treinos.remove(treino);
        System.out.println("Treino " + treino.getNomeTreino() + " removido com sucesso!");
    }

```

Figura 4 – Fonte: Autor

3 CONCLUSÃO

Em resumo, o trabalho proposto pelo professor contribuiu significativamente para a nossa compreensão de diagramas de classes. Fomos capazes de apreciar a importância da criação de diagramas de classes, uma ferramenta amplamente utilizada em diversas áreas de trabalho. É um sistema prático e de fácil entendimento, essencial para nós, programadores, e para futuros profissionais que pretendem utilizá-la para representar sistemas que planejamos desenvolver.

REFERÊNCIAS

Carrad, Marcos. UML – Unified Modeling Language. 2024. 15 slides. Notas de aula da disciplina Estrutura de Dados. UNIVALI, Itajaí. Disponível em: material didático.

Carrad, Marcos. UML – Diagrama de Classes. 2024. 18 slides. Notas de aula da disciplina Estrutura de Dados. UNIVALI, Itajaí. Disponível em: material didático.

YOUTUBE. Tutorial de Diagramas de Classes UML. Disponível em: <https://www.youtube.com/watch?v=rDidOn6KN9k&t=549s>. Acesso em: 25/05/2024