Relatório do Projeto eDOE
Laboratório de Programação 2

Anderson Filipe Clemente Silva
Bruno Roberto Silva de Siqueira
João Pedro Santino Espíndula
Vítor Braga Diniz

13 de dezembro de 2018

Sumário

1 Design Geral

O design do projeto foi arquitetado para minimizar o acoplamento do sistema de forma que foram utilizados vários níveis de abstração. No nível mais alto, há o Mediator, que tem a função de criar, controlar e integrar os controladores ....

2 Caso de Uso 1

No primeiro caso, deve ser criado um CRUD (*Create*, *Read*, *Update* e *Delete*) de usuários, os quais podem ser doadores ou receptores de itens. Para isso, a equipe de desenvolvimento implementou duas classes: "UsuarioConroller" – que tem como funções principais criar e administrar conjuntos de usuários – e "Usuario" – que é a abstração de um doador ou receptor no sistema.

Em "UsuarioController", foi criado um mapa (LinkedHashMap, uma vez que a ordem de cadastro importa) de usuários, cujo identificador único é o CPF ou CNPJ e, também, um set de classes, tal que as classes são a categoria a qual um usuário pode pertencer, como igreja, ONG, sociedade, pessoa física. Além disso, foram implementados os métodos adicionaDoador() para cadastrar usuário doador, atualizaUsuario() e removeUsuario() para atualizar os atributos de um usuário qualquer e remover um usuário, respectivamente, além dos métodos de pesquisa para encontrar um usuário a partir de um parâmetro e lerReceptores() para cadastrar os receptores.

Já "Usuario" possui id, nome, e-mail, telefone, categoria e uma variável indicadora se o usuário é doador ou receptor. Há, também, um conjunto de métodos get – para informar às classes externas os valores dos atributos – e set – para alterar os atributos.

3 Caso de Uso 2

O caso 2 pede para que os doadores possam inserir os itens a serem doados no sistema. Dessa forma, foram criadas a classe Item (que é a abstração de um item no sistema), a classe abstrata ItemController, a qual tem como objetivo gerenciar todos os itens do sistema e a classe DoadosController, que gerencia os itens doados.

Na classe ItemController, foram implementados 2 atributos, um contador, que contém o identificador (ID) único do item, e um mapa de mapa– o qual relaciona um Usuário a um outro mapa que, por sua vez, relaciona o id do item ao próprio item –, uma vez que é necessário que o item esteja externamente relacionado ao seu id único e ao seu respectivo usuário. Foram implementados métodos para cadastro, atualização e remoção de itens. Aquele gera o id único incrementando 1 para o próximo item a ser cadastrado e grava os valores necessários no mapa. Além disso, foi incorporado o método listaTodos() que lista todos os itens cadastrados em uma ordem específica solicitada pelo usuário.

Na classe DoadosController, que extende de ItemController – afinal, é, também, um controlador de itens, mas específico àqueles doados –, foram implementados um *set* de descrições, indentificando quais as descrições de itens já existentes no sistema, e métodos para cadastrar e exibir item e descritores.

Já na classe Item, foram implementados os seguintes atributos necessários para descrever um item com todas as suas propriedades: ID, descrição, quantidade, tags e usuário. Além disso, há um conjunto de métodos *get* e *set* para transmissão de dados para outras classes.

4 Caso de Uso 3

O caso 3 demanda uma ferramenta de pesquisa de itens a serem doados