Programação Orientada aos Objetos

2017/2018

LCC

Relatório do trabalho prático

Grupo nº 7:

A80298; Arlindo da Fonte Torres



A80584; Nelson José Dias Teixeira



A81241; José Alberto Martins Boticas



**Conteúdo**

1. **Introdução 3**
2. **Arquitetura de classes utilizada 5**
3. **Funcionalidades 6**
4. **Possibilidades de alterações ao projeto 7**

**1 Introdução**

No âmbito da Unidade Curricular de Programação Orientada aos Objetos (POO), foi proposto que se realizasse um trabalho prático, recriando, com as devidas alterações, a plataforma e-fatura, que disponibiliza aos contribuintes informações sobre as faturas emitidas em seu nome e lhes permite, entre outros, verificar o montante de deduções fiscais e associar um tipo de despesa a uma fatura.

Para este projeto, deve desenvolver-se uma plataforma que permite:

1. registar contribuintes, quer sejam individuais ou coletivos (empresas);
2. validar o acesso à plataforma, utilizando as credenciais (NIF e password), por parte de diferentes entidades;
3. criar faturas associadas a despesas feitas por um contribuinte individual, sendo esta funcionalidade responsabilidade das empresas;
4. aos contribuintes individuais, consultar as suas faturas;
5. aos contribuintes individuais, associar uma atividade económica às faturas emitidas em seu nome e que ainda não tenham uma associada;
6. aos contribuintes individuais, visualizar o valor acumulado de deduções fiscais;
7. corrigir a classificação de atividade económica de uma fatura, sendo deixado um rasto para posterior rastreamento;
8. obter a listagem das faturas de uma determinada empresa, ordenada por data de emissão ou por valor;
9. às empresas, obter as listagens das faturas por contribuinte num intervalo de datas;
10. às empresas, obter as listagens das faturas por contribuinte ordenadas por valor decrescente da despesa;
11. indicar o total faturado por uma dada empresa, num determinado intervalo de tempo;
12. ao administrador da aplicação, determinar a relação dos 10 contribuintes que mais gastam em todo o sistema;
13. ao administrador da aplicação, determinar a relação das X empresas que mais faturas emitiram em todo o sistema e o montante de deduções fiscais que as despesas registadas por essas empresas representam;
14. gravar o estado da aplicação em ficheiro, para seja possível retomar mais tarde a execução.

Para além destes requisitos, foram impostas regras à forma como se emite uma fatura, sendo estas as seguintes:

1. se uma empresa tiver apenas uma atividade económica, então a despesa fica automaticamente catalogada e deve determinar-se qual o valor a acrescentar às deduções fiscais do contribuinte;
2. se a empresa tiver mais do que uma atividade económica, deve ser o contribuinte a catalogar a despesa e só aí deve ser calculado o valor de dedução fiscal;
3. enquanto o contribuinte não validar a despesa, esta fica no estado pendente, não deixando de ser associada a esse contribuinte;
4. se o contribuinte não validar a despesa, esta não é considerada para efeitos de deduções.

O cálculo das deduções fiscais dependerá sempre, essencialmente, do setor de atividade económica associado à fatura, mas também poderá ser influenciado pela empresa que a emite ou pelo agregado familiar do cliente, havendo bonificações para empresas do interior ou famílias numerosas (com mais de quatro filhos). Uma empresa terá bonificação de empresa do interior se se inserir num concelho com incentivo fiscal.

**2 Arquitetura de classes utilizada**

**3 Funcionalidades**

**4 Possibilidades de alterações ao projeto**

Em relação ao que foi implementado no projeto apresentado pelo nosso grupo, poderia-se ter mudificado ou acrescentado alguns métodos. Entre eles destacam-se:

1. Acrescentar às classes Fatura, Contribuinte e Coletivo métodos de ordenação (ordem natural, entre outros) e criar um Map que associasse um nome de um comparador (Chave) ao respetivo comparador (Valor).

Consequentemente seria necessário adiciona o método que dado o nome do comparador returnava o respetivo comparador. Desta maneira, a classe Sistema sempre que necessita-se de um método para ordenar um TreeSet de elementos das classe Fatura, Contribuinte ou Coletivo, bastaria aceder ao comparador pretendido. Desta forma, uma vez que apenas se criava cada comparador uma vez, esta seria uma solução mais eficiente.

1. Possibilidade de registar as faturas numa variavel de instância do tipo Set. Desta forma, garantidamente não existiriam faturas repetidas na classe Sistema.