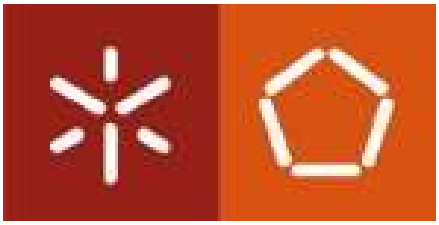

Trabalho Prático

Programação Orientada aos Objetos

(2º ano, Licenciatura em Ciências da Computação)

ImOObiliaria

```
|-------------------------|
|    Bemvindo à Remoox,   |
|  A nossa Imoobiliária!  |
|-------------------------|
```

Relatório do Projeto

Grupo 8

Ângela Fernandes (A72053)

Luís Lopes (A71016)

Sónia Costa (A71506)

Ambiente de Trabalho

O software usado para a criação desta aplicação foi:

- Sistema Operativo: Windows 8.1 | Linux Ubuntu 14.04.2 LTS
- IDE: BlueJ, versão 3.1.4

Introdução

Para a realização deste trabalho em Java, tivemos a necessidade de avaliar o problema e todas as suas necessidades e criar um esquema para tentar perceber de que forma conseguiríamos a sua implementação. Este projeto tem como objetivo o desenvolvimento de uma aplicação que permite fazer a gestão de imóveis. Esta aplicação deve dar suporte a todo o ciclo de vida de um imóvel numa agência imobiliária, desde a sua criação no sistema (sob a forma de anúncio), até ao registo da venda. Para isso será necessária a criação de dois tipos de utilizadores, nomeadamente os vendedores e os compradores.

Os utilizadores terão a possibilidade de aceder apenas a funcionalidades específicas do seu tipo, ou seja, os compradores vão ter a possibilidade de pesquisar imóveis dado um conjunto de características ou identificador do mesmo, não sendo necessário estarem registados na aplicação. Poderão também marcar um imóvel como favorito sendo que para efetuar esta opção o utilizador tem de estar registado e autenticado no sistema como comprador. Os vendedores poderão inserir, consultar e remover anúncios de imóveis, bem como alterar o seu estado (em venda, reservado ou vendido). Estes têm também a possibilidade de aceder à informação atualizada sobre as estatísticas (número de anúncios criados, número de visualizações dos anúncios e número de vendas de imóveis).

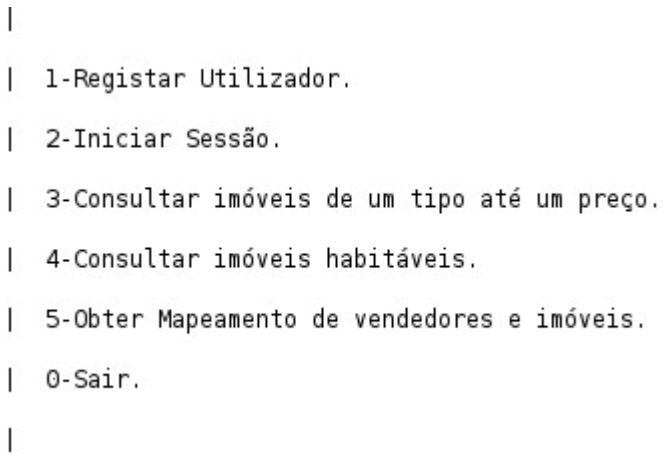

Fig.1 - Menu Inicial

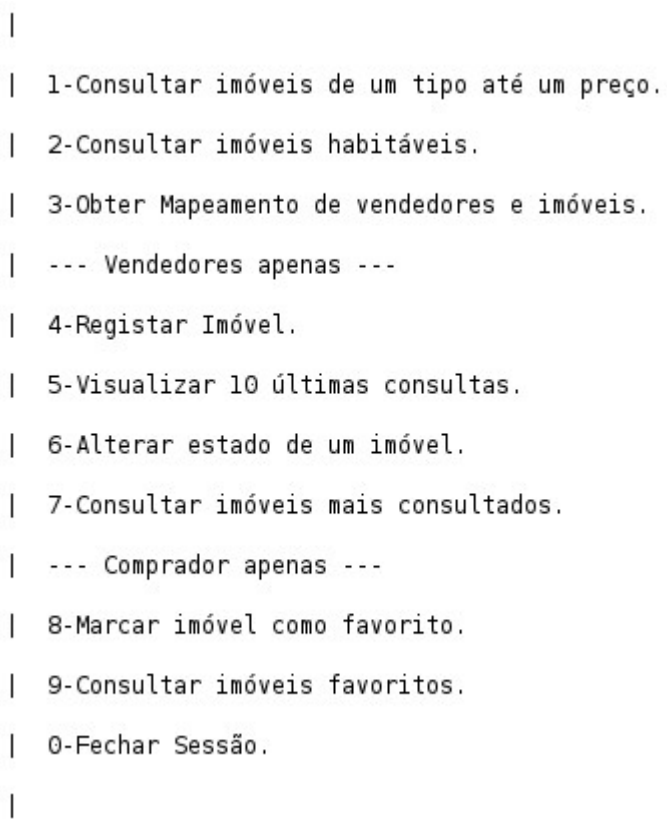

Fig.2 – Menu Utilizador Registado

Especificação

Para o desenvolvimento desta aplicação foi necessária a implementação de várias classes: Utilizador, Vendedor, Comprador, Imovel, Moradia, Apartamento, Loja, Lojahab, Terreno, Consulta, Imoobiliaria, App, ComparatorPreco, ImovelInexistenteException, SemAutorizacaoException, ImovelExisteException, EstadoInvalodoException, UtilizadorExistenteException. Assim como a implementação de uma instancia: Habitavel.

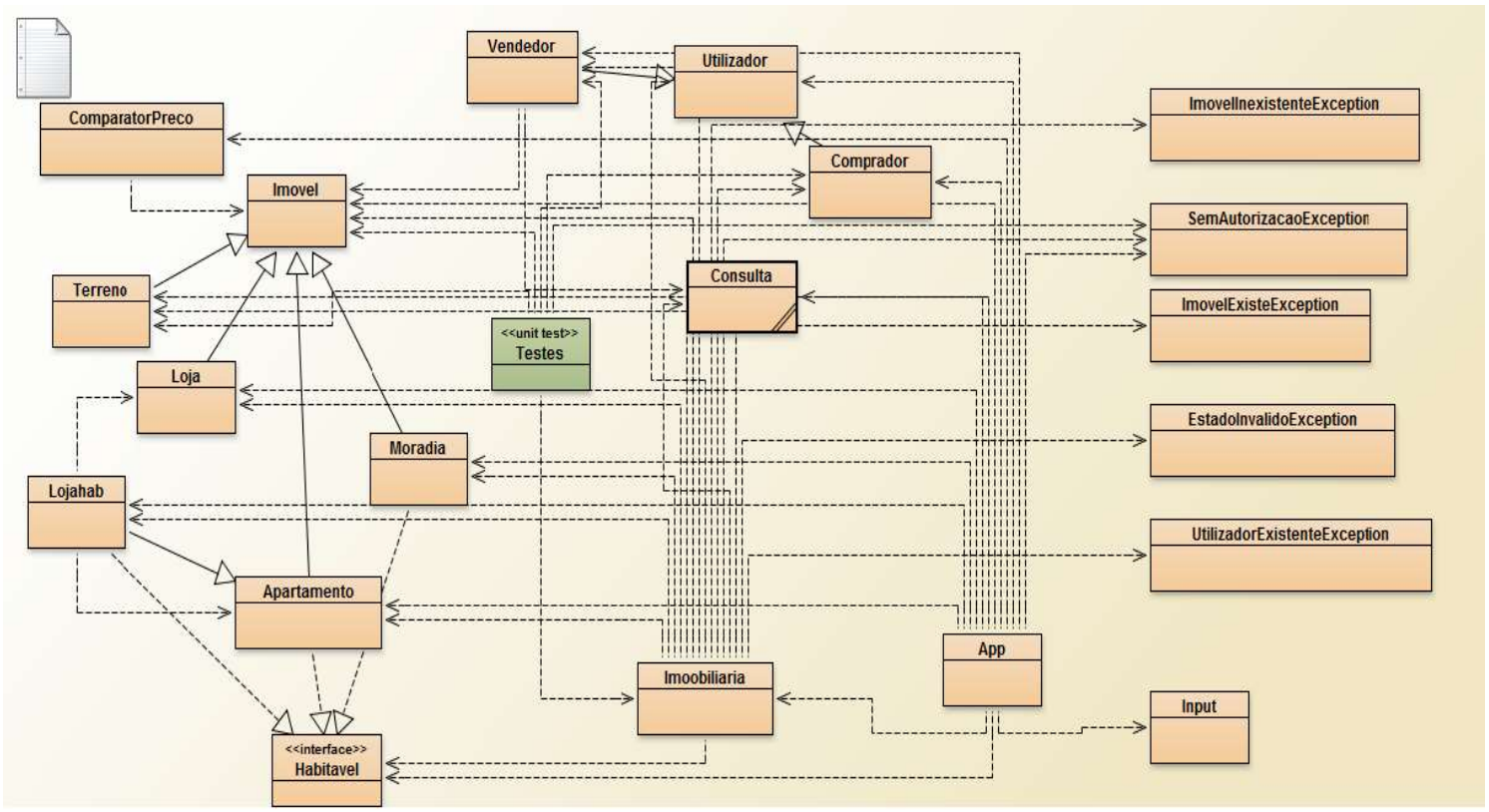

Fig.3 – Esquema da aplicação

Imoobiliaria

Na classe Imoobiliaria encontra-se toda a parte logística da aplicação, nomeadamente, carregamento de dados a partir de um estado previamente guardado, registo de utilizador, início de sessão, registo de imóvel, consulta de imoveis, alteração de imoveis, mapeamento de imoveis e respetivos utilizadores. É também onde se armazenam todos os dados referentes aos utilizadores e aos imóveis.

```
public class Imoobiliaria {

    public static HashMap<String, Imovel> imoveis;
    public static HashMap<String, Utilizador> utilizadores;
    public static ArrayList<Consulta> consultas;
    public static int contador;
```

Fig.4 – Classe Imoobiliaria

Utilizador

A classe Utilizador é uma superclasse que tem como subclasses as classes Vendedor e Comprador.

```
public class Utilizador {

    private int tipo;
    private String email;
    private String nome;
    private String password;
    private String dataNascimento;
    private String morada;
```

Fig.5 - Classe Utilizador

Vendedor

A classe Vendedor é uma subclasse da classe Utilizador. Nesta classe, temos um ArrayList que armazena a identificação de todos os imóveis que o Vendedor tem para venda. Assim como um HashMap para os imóveis já vendidos onde a chave é a identificação do imóvel.

```
public class Vendedor extends Utilizador{

    public ArrayList<String> portfolio;
    public HashMap<String,Imovel> historico;
```

Fig.6 – Classe Vendedor

Comprador

A classe Comprador é uma subclasse da classe Utilizador. Nesta classe está implementado um ArrayList que armazena a identificação dos imóveis favoritos.

```
public class Comprador extends Utilizador implements Serializable{

    public ArrayList<String> favoritos;
```

Fig.7 – Classe Comprador

Imovel

A classe Imovel é uma superclasse cujas subclasses são: Moradia, Apartamento, Loja, Terreno, Lojahab. Os imóveis são comparáveis entre si através da classe ComparatorPreco.

```
public class Imovel implements Serializable{

    public int classe;
    private String idImovel;
    private String rua;
    private double precoPedido;
    private double precoMinimo;
    private String estado;
```

Fig.8 – Classe Imovel

Habitavel

A instância Habitavel permite classificar os diferentes imoveis em função da sua habitabilidade.

App

Na classe App ocorre toda a interação entre o utilizador e o programa.

Como seria possível incluir novos tipos de imóveis na aplicação?

Para incluir novos imóveis seria necessário criar novas classes que seriam sublasses de Imovel. Seria necessário alterar o código na classe App para que estes novos imóveis passassem a ser reconhecidos pela aplicação.

Problemas de Implementação

Tivemos um problema na implementação dos métodos setFavoritos(String idImovel) e getFavoritos() na classe Imoobiliaria no entanto as operações respetivas a estes métodos funcionam com a implementação de métodos na classe App.

Conclusão

No final da realização deste trabalho, conseguimos completar a maioria dos requisitos básicos. A aplicação é iniciada com o carregamento de dados previamente guardados num ficheiro txt. Esta aplicação, permite a todos os seus utilizadores fazer uma consulta dos imóveis e se registarem. Aos utilizadores registados permite também iniciar sessão, além disso, aos vendedores permite, registar imóvel, visualizar as 10 últimas consultas aos seus imóveis, alterar o estado de um imóvel e saber quais os imóveis mais consultados. Os compradores podem também, além de se registar e iniciar sessão, marcar um imóvel como favorito e consultar os seus imóveis favoritos.

Ao realizar este trabalho, foram adquiridos conhecimentos sobre a linguagem Java. Atingimos os objetivos pretendidos tanto a nível de trabalho realizado, como a nível do conhecimento e compreensão das matérias abordadas.