Gustavo Viegas - 3026

Atividade Prática 2 - Biblioteca Drone

Documentação de Atividade Prática - AP2

Universidade Federal de Viçosa - Campus Florestal

Programação Orientada a Objetos

Ciência da Computação

Florestal

14 de outubro de 2018

Lista de ilustrações

Sumário

1 Introdução

Esta documentação descreve a atividade proposta para prática de algoritmos utilizando os conceitos de orientação a objetos na linguagem Java. A atividade proposta foi o desenvolvimento de um sistema de gerenciamento de biblioteca que possui um drone que realiza a entrega àos clientes.

O sistema deveria ter a funcionalidade de autenticação, consulta e pesquisa de livros e empréstimos.

Juntamente com a entrega desta documentação, também está incluso todo o código fonte desenvolvido, o artifato JAR final com o programa e as bibliotecas utilizadas. Um usuário teste pode ser utilizado com matrícula: *3026* e senha: *secret*.

2 Ambiente

Para o desenvolvimento do trabalho foi utilizada a IDE **IntelliJ IDEA 2018.2** e o sistema operacional **macOS High Sierra**.

A compilação do código foi feita pela própria *IDE*, utilizando-se das seguintes versões do Java:

- Java 10 2018-03-20
- Java(TM) SE Runtime Environment 18.3 (build 10+46)
- Java HotSpot(TM) 64-Bit Server VM 18.3 (build 10+46, mixed mode)

A execução do artefato final (JAR) foi realizada pelo próprio sistema operacional, e o desenvolvimento foi feito utilizando o software **SceneBuilder** para prototipar o wireframe, e a IDE para executar e compilar o projeto.

3 Funcionamento

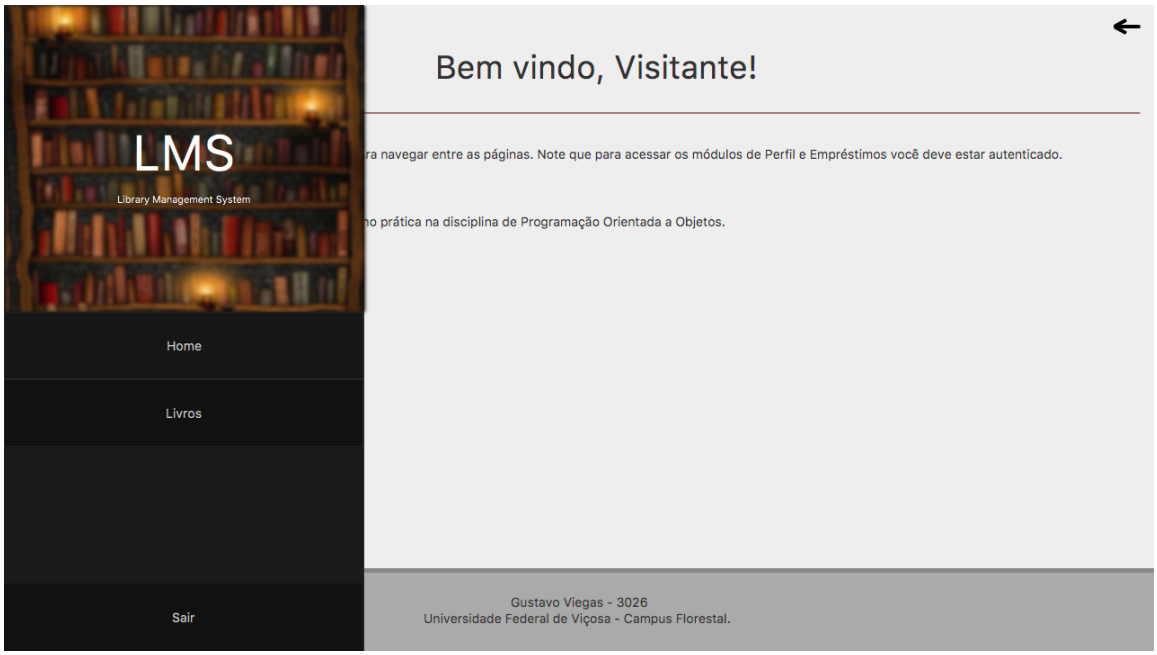
3.1 Autenticação

Ao abrir a aplicação, a página de autenticação é carregada. Nesta página o usuário pode preencher sua matrícula e senha ou continuar como visitante. O formulário valida os campos obrigatórios e se as credenciais estão corretas antes de direcionar para a página principal.

Figura 1 – Página de autenticação de usuário

Se o usuário continuar como visitante, ele só terá acesso a lista e busca de livros. Se autenticado, poderá consultar os livros e escolher um para tentar solicitar empréstimos e gerenciar os seus empréstimos.

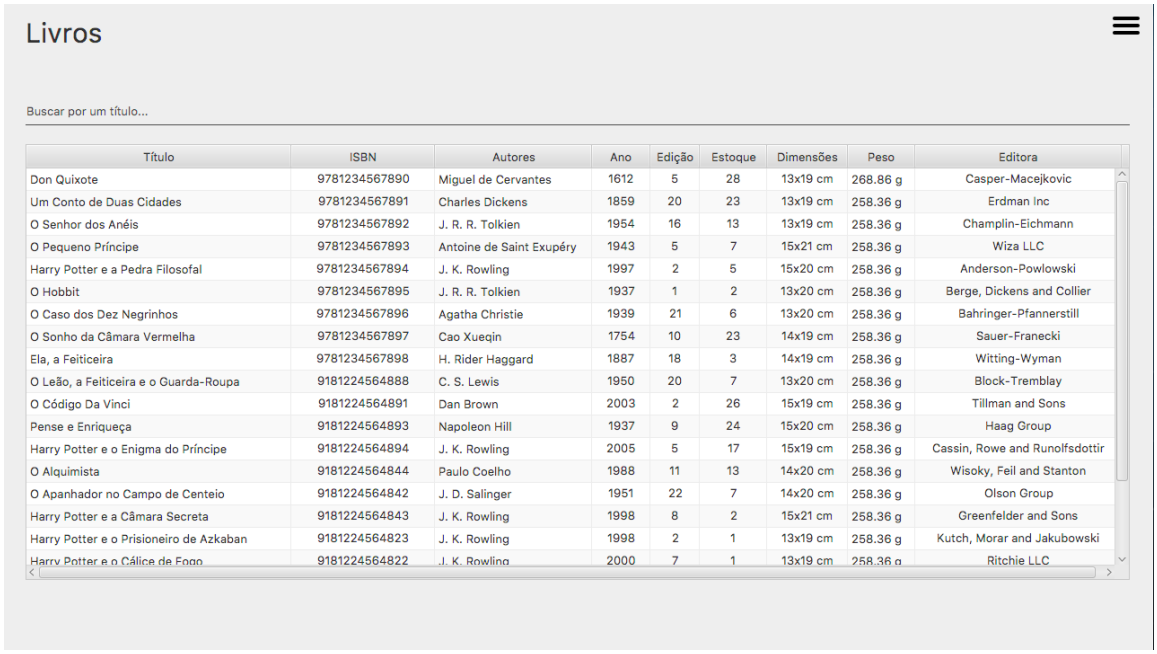
Figura 2 – Página inicial para visitante e o menu lateral com opções restritas



3.2 Listagem e Busca de Livros

Ao entrar na página de livros, é exibida uma tabela com os livros cadastrados no sistema e seus respectivos dados. A ordem padrão é a do mais vendido para o menos vendido, por simples conveniência da ordem descrita no arquivo de leitura.

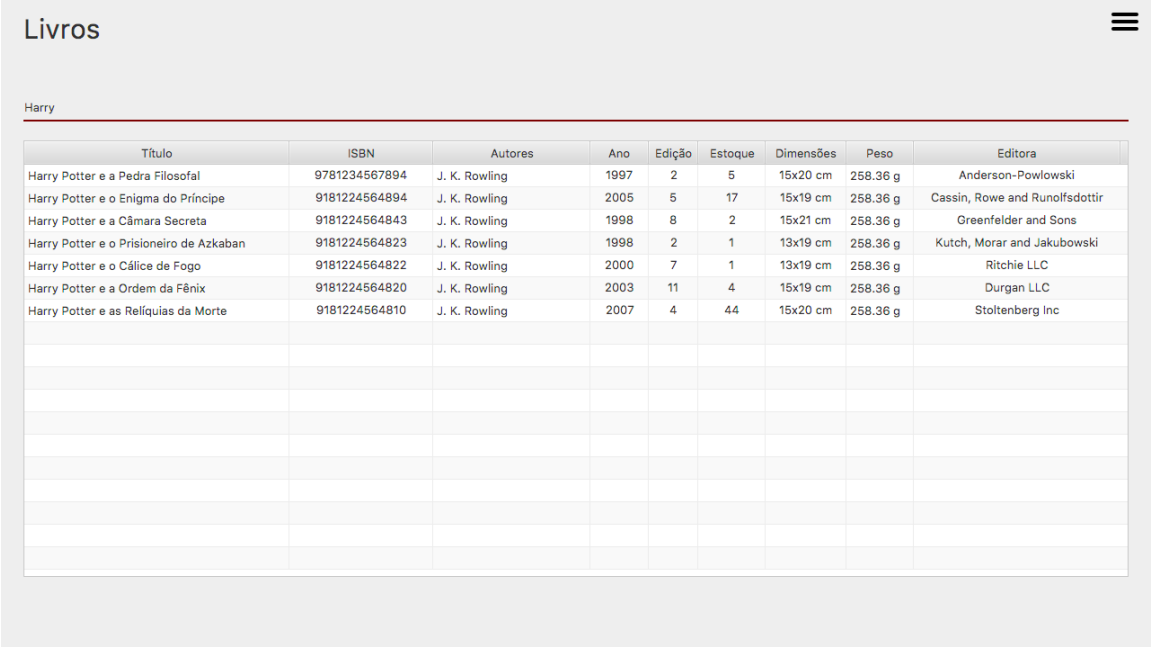
Figura 3 – Página de livros



Título	ISBN	Autores	Ano	Edição	Estoque	Dimensões	Peso	Editora
Don Quixote	9781234567890	Miguel de Cervantes	1612	5	28	13x19 cm	268.86 g	Casper-Macejkovic
Um Conto de Duas Cidades	9781234567891	Charles Dickens	1859	20	23	13x19 cm	258.36 g	Erdman Inc
O Senhor dos Anéis	9781234567892	J. R. R. Tolkien	1954	16	13	13x19 cm	258.36 g	Champlin-Eichmann
O Pequeno Príncipe	9781234567893	Antoine de Saint Exupéry	1943	5	7	15x21 cm	258.36 g	Wiza LLC
Harry Potter e a Pedra Filosofal	9781234567894	J. K. Rowling	1997	2	5	15x20 cm	258.36 g	Anderson-Powlowski
O Hobbit	9781234567895	J. R. R. Tolkien	1937	1	2	13x20 cm	258.36 g	Berge, Dickens and Collier
O Caso dos Dez Negrinhos	9781234567896	Agatha Christie	1939	21	6	13x20 cm	258.36 g	Bahringer-Pfannerstill
O Sonho da Câmara Vermelha	9781234567897	Cao Xueqin	1754	10	23	14x19 cm	258.36 g	Sauer-Franecki
Ela, a Feiticeira	9781234567898	H. Rider Haggard	1887	18	3	14x19 cm	258.36 g	Witting-Wyman
O Leão, a Feiticeira e o Guarda-Roupa	9181224564888	C. S. Lewis	1950	20	7	13x20 cm	258.36 g	Block-Tremblay
O Código Da Vinci	9181224564891	Dan Brown	2003	2	26	15x19 cm	258.36 g	Tillman and Sons
Pense e Enriqueça	9181224564893	Napoleon Hill	1937	9	24	15x20 cm	258.36 g	Haag Group
Harry Potter e o Enigma do Príncipe	9181224564894	J. K. Rowling	2005	5	17	15x19 cm	258.36 g	Cassin, Rowe and Runolfsdottir
O Alquimista	9181224564844	Paulo Coelho	1988	11	13	14x20 cm	258.36 g	Wisoky, Feil and Stanton
O Apanhador no Campo de Centeio	9181224564842	J. D. Salinger	1951	22	7	14x20 cm	258.36 g	Olson Group
Harry Potter e a Câmara Secreta	9181224564843	J. K. Rowling	1998	8	2	15x21 cm	258.36 g	Greenfelder and Sons
Harry Potter e o Prisioneiro de Azkaban	9181224564823	J. K. Rowling	1998	2	1	13x19 cm	258.36 g	Kutch, Morar and Jakubowski
Harry Potter e o Cálice de Fogo	9181224564822	J. K. Rowling	2000	7	1	13x19 cm	258.36 g	Ritchie LLC

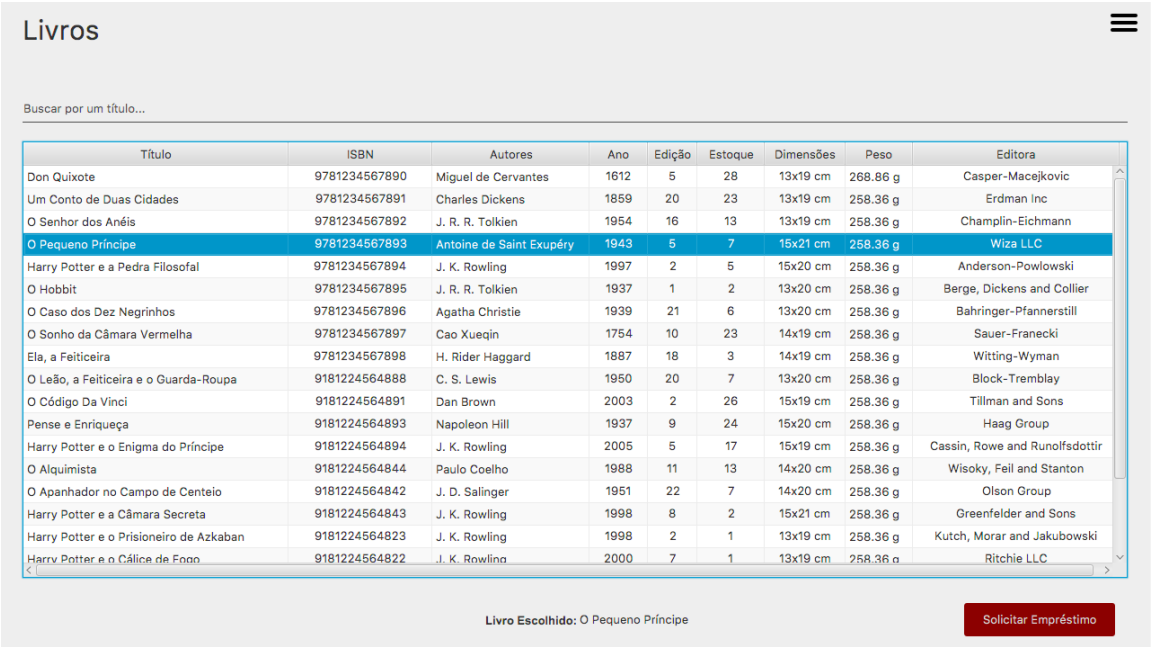
A busca é feita pelo campo de texto logo acima da tabela. Ao digitar uma string e apertar a tecla enter, a tabela é filtrada com os títulos parecidos com o texto fornecido.

Figura 4 – Busca de livros que casem com um critério



Se o usuário estiver autenticado, ao clicar em uma linha da tabela, aparecerá um campo no canto inferior mostrando o título do livro escolhido e um botão para solicitar o empréstimo deste livro. O sistema validará as condições após a confirmação de empréstimo do usuário, e registrará este empréstimo, caso as condições sejam atendidas.

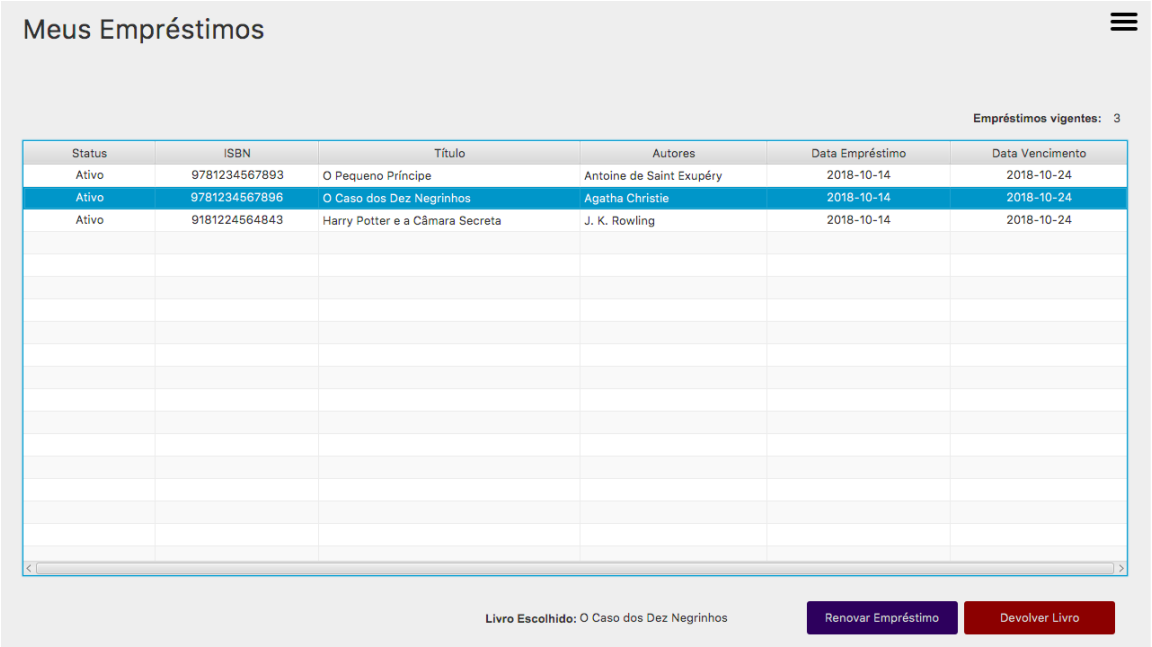
Figura 5 – Seleção de um livro na tabela



3.3 Página de Empréstimos

Para o usuário autenticado, a página de empréstimos lista todos os empréstimos solicitados pelo usuário, seja eles vigentes ou não. Na tabela mostra-se os dados dos empréstimos, e assim como na página de livros, também há um campo com opções de gerenciamento do empréstimo escolhido ao clicar em uma linha da tabela. As opções são de renovar o empréstimo ou devolver o livro.

Figura 6 – Listagem de empréstimos de um usuário



4 Decisões

4.1 Interface Gráfica

Foi decidido utilizar uma interface gráfica para além de aprender mais sobre o desenvolvimento de GUI, poder também ter um sistema mais robusto e amigável do que pela linha de comando. A praticidade de gerar um JAR após o desenvolvimento do projeto também é um outro pró para a criação de uma interface, que seria universal em qualquer sistema operacional.

Foi utilizado a biblioteca **JavaFX** com o auxílio do sistema **SceneBuilder** que permite a prototipagem fácil da interface.

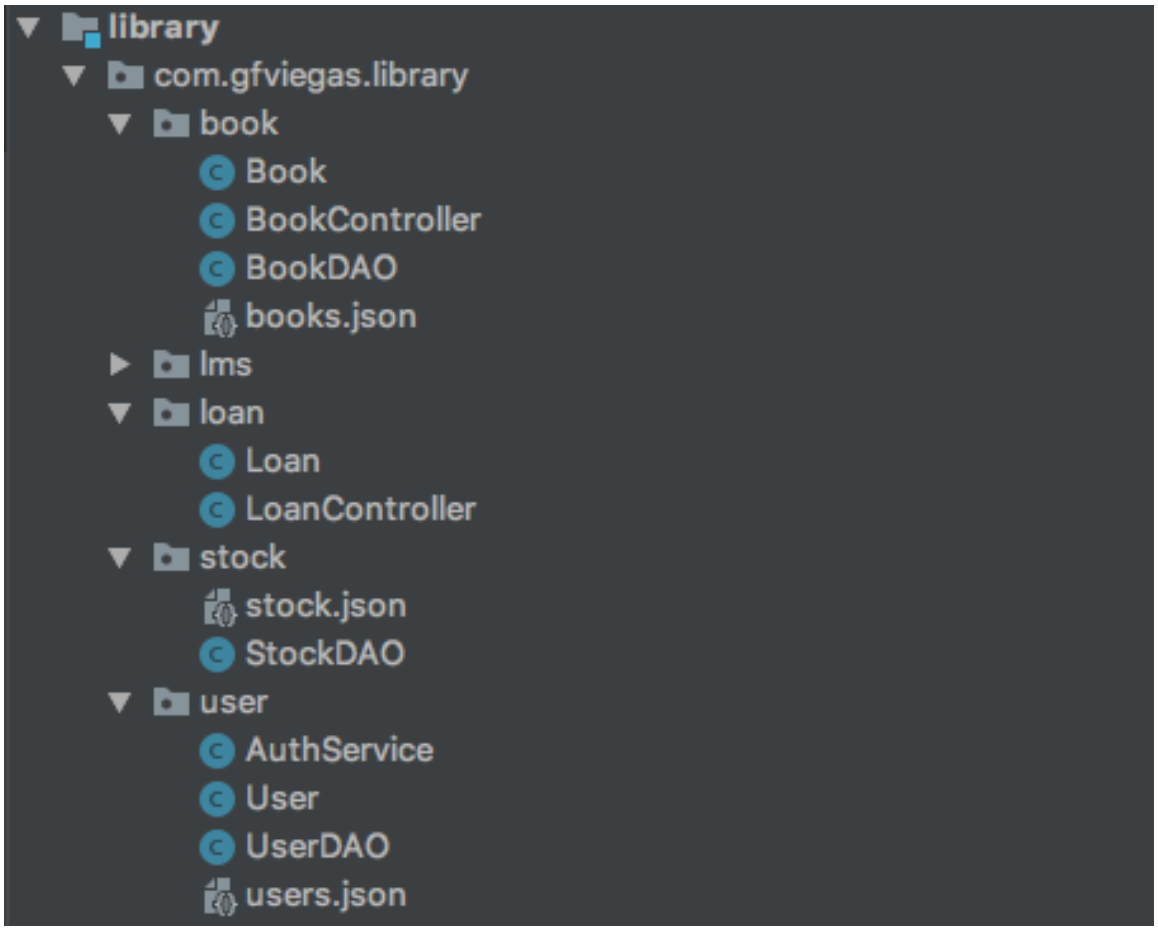
Cada página é um arquivo parecido a um *XML*, na extensão *.fxml*, que se conecta com uma classe Java, padronizada no projeto como uma classe *View*.

4.2 Organização dos Pacotes e Classes

As classes seguiram um padrão parecido ao MVC, tendo classes descrevendo um TAD, controllers e DAOs. Além disso tem as classes *view* que ficaram todas agrupadas em um pacote separado junto com os arquivos de layout, de forma a isolar a visão do resto do sistema.

Os pacotes foram organizados de acordo com cada módulo e não por categoria da classe. Por exemplo, **user** é um módulo que contêm todas as classes referentes ao Usuário, ou seja, seu "modelo", seu controller, seu DAO e seu arquivo de dados.

Figura 7 – Organização de módulos do sistema



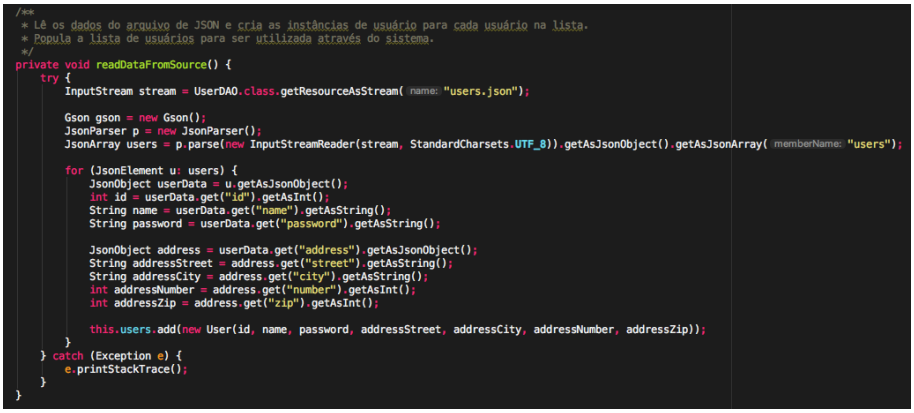
4.3 Arquivos de Dados

Figura 8 – Exemplo de arquivo JSON


```json
{
  "users": [
    {
      "id": "3026",
      "name": "Gustavo Viegas",
      "password": "secret",
      "address": {
        "street": "José Alves Pinto",
        "city": "Betim",
        "number": 21,
        "zip": 32604020
      }
    },
    {
      "id": "3106",
      "name": "Hattie Lamplugh",
      "password": "AVSdKStQy6",
      "address": {
        "street": "Spohn",
        "city": "Maumbawa",
        "number": 5,
        "zip": 3719975
      }
    },
    {
      "id": "2486",
      "name": "Melany Ravenhills",
      "password": "5X5UfYym3mGN",
      "address": {
        "street": "Redwing",
        "city": "Nossa Senhora da Glória",
        "number": 8281,
        "zip": 3250094
      }
    },
```


Os dados são carregados a partir de arquivos **JSON**, utilizando o auxílio da biblioteca **GSON**, da Google, que permite interpretar os arquivos e facilmente carregar os dados nele inseridos. Este tipo de arquivo foi escolhido como forma de aproximar o desenvolvimento do sistema com uma API RESTful do mundo real, de forma que os dados ao invés de serem carregados por um arquivo local, seria servido, na mesma extensão, por requisições.

Figura 9 – Utilização da biblioteca GSON para ler o arquivo



```java
/**
 * Lê os dados do arquivo de JSON e cria as instâncias de usuário para cada usuário na lista.
 * Popula a lista de usuários para ser utilizada através do sistema.
 */
private void readDataFromSource() {
    try {
        InputStream stream = UserDAO.class.getResourceAsStream( name: "users.json");

        Gson gson = new Gson();
        JsonParser p = new JsonParser();
        JsonArray users = p.parse(new InputStreamReader(stream, StandardCharsets.UTF_8)).getAsJsonObject().getAsJsonArray( memberName: "users");

        for (JsonElement u: users) {
            JsonObject userData = u.getAsJsonObject();
            int id = userData.get("id").getAsInt();
            String name = userData.get("name").getAsString();
            String password = userData.get("password").getAsString();

            JsonObject address = userData.get("address").getAsJsonObject();
            String addressStreet = address.get("street").getAsString();
            String addressCity = address.get("city").getAsString();
            int addressNumber = address.get("number").getAsInt();
            int addressZip = address.get("zip").getAsInt();

            this.users.add(new User(id, name, password, addressStreet, addressCity, addressNumber, addressZip));
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Conclusão

Através do trabalho realizado evidencia-se a importância de se tomar cuidado com as decisões referentes a orientação a objetos em um sistema, na linguagem Java, uma vez que o sistema robusto e reutilizável é muito mais facilmente aproveitado do que aquele sem utilizar-se dos paradigmas estudados.

Destaca-se também a relevância do encapsulamento para proteger as estruturas de dados, da herança para reutilização de código facilmente pelo projeto e principalmente das tecnologias do universo Java para o desenvolvimento de um sistema.

O trabalho, portanto, se mostrou útil para reforçar ideias de programação, algoritmos e estruturas de dados e, principalmente, de orientação a objetos.