Pedro Felipe Froes Silva

Desenvolvimento de Aplicações em Java e de Interfaces de Usuário

Belo Horizonte

2017

Pedro Felipe Froes Silva

Desenvolvimento de Aplicações em Java e de Interfaces de Usuário

Relatório apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para a aprovação na disciplina Estágio Supervisionado.

Centro Federal de Educação Tecnológica de Minas Gerais

Departamento de Computação

Curso de Engenharia de Computação

Orientador: Kecia Aline Marques Ferreira

Empresa: Avenue Code Desenvolvimento e Comércio de Softwares Ltda

Belo Horizonte

2017

Resumo

Este trabalho descreve as atividades desenvolvidas durante o período de Estágio Supervisionado do aluno Pedro Felipe Froes do curso de Engenharia de Computação do CEFET-MG enquanto estagiário na empresa Avenue Code. Parte do programa de Jedi Internship da empresa é descrita, focando nas áreas de aprendizador Java e desenvolvimento de interface de usuários (UI). Ambas as áreas têm sua fundamentação teórica presente no Capítulo 3 deste trabalho, e o Capítulo 4 descreve as atividades desenvolvidas em cada uma delas. Finalmente, o Capítulo 5 conclui o trabalho.

Palavras-chave: Estágio supervisionado. Java. *User interface*.

Lista de ilustrações

Lista de abreviaturas e siglas

API	Interface de programação de aplicações (*application programming interface*)
CSS	Folhas de estilo em cascata (*cascading stylesheets*)
DAO	Objeto de acesso a dados (*data access object*)
JDBC	Java Database Connectivity
HTML	Linguagem de marcação de hipertexto (*hyper text markup language*)
UI	Interfaces de usuário (*user interfaces*)
WORA	Escreva uma vez, rode em qualquer lugar (*Write once, run anywhere*)

Sumário

1 Introdução

O presente trabalho é um relatório da disciplina de Estágio Supervisionado pertencente ao curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG), realizado pelo discente Pedro Felipe Froes na empresa Avenue Code Desenvolvimento e Comércio de Software Ltda. Este relatório reflete o período de Março a Maio de 2017 durante o estágio do aluno na empresa, contemplando parte do Programa de Estágio Jedi Internship.

O Programa de Estágio Jedi Internship da Avenue Code corresponde a uma rotação dos participantes por diferentes tecnologias presentes na área de Computação. Durante o período do programa, o estagiário passa por cinco áreas distintas, obtendo um aprendizado de linguagens de programação como (i) Java e (ii) Ruby, (iii) do *framework* .NET, (iv) de conceitos de garantia de qualidade de software, e (v) de conceitos e *frameworks* para o desenvolvimento de interfaces de usuário (*user interface*, UI). O estudante exercita o aprendizado de cada área por meio de projetos internos da empresa, e apresenta um *workshop* com o conteúdo aprendido ao final de cada etapa.

O objetivo desse trabalho é relatar as experiências do discente nas áreas em que o mesmo participou durante o período do Estágio Supervisionado, que correspondem ao aprendizado de Java e de conceitos de UI. O aprendizado em cada uma das áreas é apresentado por meio do processo de desenvolvimento de um migrador de banco de dados em Java e da construção de interfaces de usuário por meio de *frameworks* de UI.

O restante desse trabalho está organizado de forma que o Capítulo 2 apresenta a empresa e o seu Programa de Estágio, enquanto o Capítulo 3 aponta conceitos básicos de Java e de UI. O Capítulo 4 detalha as atividades em cada uma das áreas, enquanto o Capítulo 5 conclui o trabalho.

2 Estágio supervisionado

Neste capítulo, a empresa concedente Avenue Code e o Programa de Estágio Jedi Internship são apresentados. Tópicos envolvendo a história, especialidade e iniciativas da empresa, bem como a estrutura do programa de estágio são ilustrado nas seções a seguir.

2.1 Sobre a empresa: Avenue Code

A Avenue Code Desenvolvimento e Comércio de Softwares Ltd é uma empresa consultora de softwares especializada no ramo de *e-commerce* da indústria varejista. Fundada em 2008 pelo CEO Zeo Solomon na cidade de San Francisco (Califórnia, EUA), a Avenue Code atendia clientes americanos, abrindo seu primeiro escritório no Brasil somente um ano depois, na cidade de Belo Horizonte. Nos anos seguintes, a empresa expande e inaugura um escritório na cidade de São Paulo, além de adicionar Amir Razmara e Chase Hill ao time de CEOs. Em 2017, a Avenue Code é formada por mais de 230 consultores em sua equipe, além de inaugurar um quarto escritório, agora na cidade de Nova York (AVENUE CODE, 2017).

A Avenue Code é especialista no desenvolvimento e utilização de diversos tipos de tecnologias da área de Computação, como aplicações Web e móveis, automação de infraestruturas, sistemas de *backend*, implementações de plataformas, *coaching* Agile e DevOps e integrações corporativas. A Metodologia Ágil, oriunda do Manifesto Ágil para o Desenvolvimento de Software (FOWLER; HIGHSMITH, 2001), é altamente aplicada na empresa, que busca maximizar sua eficiência através da utilização de princípios Agile no desenvolvimento de projetos (AVENUE CODE, 2017).

Prezando tanto pela qualidade da tecnologia utilizada quanto pelo ambiente de trabalho dos consultores, a Avenue Code possui uma gama de empresas parceiras e de prêmios obtidos em sua história. Dentre as empresas de tecnologia parceiras da Avenue Code, figuram a Mulesoft [1], SAP Hybris [2], CHEF [3], Oracle [4], Amazon Web Services [5] e Adobe [6]. Já entre os prêmios conquistados pela empresa, estão o reconhecimento pelo

[1] MuleSoft: Integration platform for connecting SaaS and enterprise applications. Disponível em: <https://www.mulesoft.com/>
[2] SAP Hybris: E-commerce solutions. Disponível em: <https://www.hybris.com/en/>
[3] Chef: automate IT infrastructure. Disponível em: <https://www.chef.io/chef/>
[4] Oracle: Integrated cloud applications and platform services. Disponível em: <https://www.oracle.com/>
[5] Amazon Web Services: Cloud computing services. Disponível em: <https://aws.amazon.com/>
[6] Adobe: Creative, marketing and document management solutions. Disponível em <www.adobe.com/>

LoveMondays [7], InfoMoney [8] e San Francisco Business Times's Fast 100 [9] em 2016, além de ser agraciada como uma das Melhores Empresas para Trabalhar [10] em 2016 (*Great Place to Work*) (AVENUE CODE, 2017).

Por fim, a Avenue Code participa de iniciativas de inclusão digital por meio do programa AC Social, que oferta aulas de introdução à tecnologias e computação em escolas carentes. A empresa também oferece cursos ministrados pelos próprios consultores por meio do AC Community, além de ofertar dois programas de estágios distintos, o AC Wonder Women e o Jedi Internship, que será apresentado na seção seguinte (AVENUE CODE, 2017).

2.2 Sobre o estágio: Programa de Estágio Jedi Internship

O Programa de Estágio Jedi Internship consiste de uma rotação (*job rotation*) por cinco áreas de diferentes tecnologias presentes na área de Computação. São elas:

- Conceitos e *frameworks* de *user interface* (UI)
- Conceitos e *frameworks* de garantia de qualidade de software
- Linguagem de programação Java
- Linguagem de programação Ruby
- *Framework* .NET

Cada área dura cerca de três meses, e em cada uma delas o estagiário aprende conceitos introdutórios e avançados da tecnologia, aplicando-os em projetos internos da empresa. Cada área possui consultores que atuam como mentores para os participantes e, ao final da área, o estagiário elabora um *workshop* para ser apresentado tanto para seus mentores quanto para os outros participantes do Programa, exibindo conceitos, aplicações desenvolvidas e desafios encontrados ao longo dos três meses.

Esse trabalho apresentará os conceitos aprendidos e aplicações desenvolvidas pelo autor ao longo das áreas de Java e UI, sendo que uma fundamentação teórica para ambas as áreas é exibida no capítulo seguinte.

[7] Love Mondays: A empresa ideal, avaliada por profissionais como você. Disponível em: <https://www.lovemondays.com.br/>
[8] InfoMoney: Notícias, ações e muito mais sobre investimentos. Disponível em: <www.infomoney.com.br/>
[9] San Francisco Business Time Fast 100. Disponível em: <http://www.bizjournals.com/sanfrancisco/blog/2016/10/bay-area-fast-growing-private-companies-fast-100.html>
[10] Great Place to Work. Disponível em: <www.greatplacetowork.com.br/>

3 Fundamentação teórica

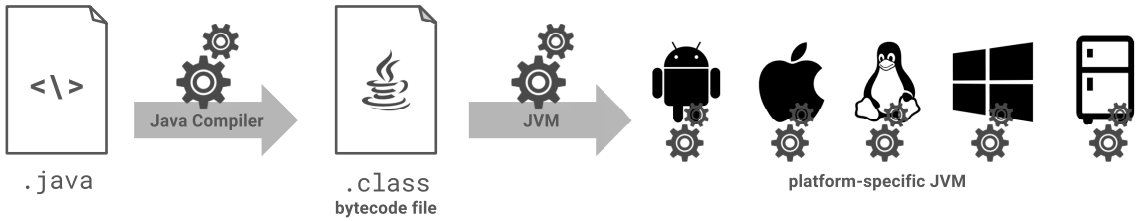
Neste capítulo, a fundamentação teórica das tecnologias abordadas neste trabalho é realizada. Conceitos básicos sobre a linguagem de programação Java são mostrados, abordando tanto os pilares da orientação a objetos presente na linguagem quanto bibliotecas relevantes para as aplicações desenvolvidas na área. Posteriormente, conceitos relevantes para a o desenvolvimento de interfaces de usuário (*user interface*, UI) são apresentados, apontando características e ferramentas utilizada no processo de construção de interfaces.

3.1 Java

Uma das tecnologias presentes no Programa de Estágio Jedi Internship e abordadas nesse trabalho corresponde ao aprendizado e aplicação de conceitos da linguagem de programação Java. Originada inicialmente em 1991 com o codinome de Oak, e nomeada de Java somente em 1995, atualmente a linguagem é uma das mais utilizadas na área de Computação, sendo administrada pela empresa Oracle e estando em sua oitava versão (BOYARSKY; SELIKOFF, 2015).

Em suas primeiras versões, o propósito inicial do Java era conectar diferentes tipos de micro-sistemas da empresa Sun, sendo uma linguagem comum entre eles. A habilidade de escrever um código que pode funcionar em mais de um sistema é conhecida como *write once, run anywhere* (WORA), sendo uma das principais características da linguagem. Ao escrever um código em Java, o compilador Javac processa o arquivo fonte para um arquivo em *bytecode*, e um interpretador JVM específico para a plataforma se encarrega de processar esse arquivo posteriormente, como mostrado na Figura 1 (BOYARSKY; SELIKOFF, 2015).

Figura 1 – Esquema de execução de um código escrito em Java.



Fonte: Próprio autor.

Além do WORA, outra característica marcante da linguagem é a implementação do conceito de orientação a objetos. Para implementar tal conceito, o Java faz uso de

classes e objetos: enquanto uma classe funciona como uma especificação de uma ideia, um objeto corresponde a uma instância, uma materialização dessa ideia, sendo que uma classe pode possuir vários objetos instanciados (BOYARSKY; SELIKOFF, 2015).

A relação entre classes e objetos dá margem para diversos outros conceitos presentes na linguagem Java. O conceito de encapsulamento, por exemplo, faz uso de modificares de acesso nos atributos e métodos de cada classe para controlar quais objetos podem acessá-los, enquanto o conceito de herança entre as classes determina uma relação de pai-filho entre elas, tornando possível que uma herde atributos e métodos da outra (BOYARSKY; SELIKOFF, 2015).

Conceitos de abstração, composição e interfaces também estão presentes na linguagem, possibilitando a criação de classes não instanciáveis, classes compostas por diversos objetos, e a garantia que alguns métodos são implementados em determinadas classes, respectivamente. Por fim, um conceito essencial da linguagem é o polimorfismo, que permite que um objeto seja referenciado por diversas maneiras, como por meio das classes que ele herda, ou de interfaces que implementa (BOYARSKY; SELIKOFF, 2015).

Além dos conceitos apresentados, o Java ainda conta com bibliotecas como o *collections framework*, que implementam diferentes estruturas de dados utilizadas comumente na programação de computadores. Estruturas como *hashes*, listas de vetores, pilhas e filas podem ser utilizadas através das interfaces `HashSet`, `ArrayList`, `Stack` e `Queue`, respectivamente. Além de eliminar a necessidade do programador construir cada uma das estruturas, a utilização delas é extremamente comum em aplicações construídas em Java, facilitando o trabalho do desenvolvedor (WATT; BROWN, 2001).

Tanto os elementos presentes no *collections framework* quanto os conceitos apresentados previamente podem ser utilizados para diversos tipos de aplicações em Java (WATT; BROWN, 2001). O gerenciamento de banco de dados relacionais em Java, por exemplo, utiliza uma interface de programação de aplicações (*application programming interface*, API) chamada Java Database Connectivity (JDBC). O JDBC abstrai a implementação de banco de dados específicos, criando uma camada única que contribui para a implementação de métodos para estabelecer conexões, criar *queries* de acesso, e extrair resultados de buscas, por exemplo (REESE, 2000).

O JDBC ainda conta com maneiras para gerenciar múltiplas conexões em um servidor através de um *pool* de conexões, e de centralizar o acesso de dados por meio de objetos de acesso a dados (*data access object*, DAO) (REESE, 2000). A utilização de alguns desses recursos é aprofundada na Seção 4.1 do Capítulo 4, que descreve as atividades realizadas durante a área de Java do Programa de Estágio, focando na utilização da linguagem atrelada ao *backend* de sistemas.

3.2 Desenvolvimento de interfaces de usuário

Uma das principais tarefas na construção do *frontend* de um sistema é o desenvolvimento de interfaces de usuário (*user interfaces*, UI). No desenvolvimento de uma aplicação Web, o designer de interfaces cria interfaces gráficas guiado por um designer de experiência de usuário (*user experience*, UX), e passa as interfaces prontas para o desenvolvedor UI. Em posse das telas, cabe ao desenvolvedor transformá-las em código, o que ocorre essencialmente por meio da combinação de três tecnologias diferentes: a Linguagem de Marcação de Hipertexto (*Hyper Text Markup Language*, HTML), as Folhas de Estilo em Cascata (*Cascading Stylesheets*, CSS) e a linguagem de programação de propósito geral JavaScript (OPPERMANN, 2002; STEFANER et al., 2009).

Para desenvolver as interfaces de usuário, é necessário primeiro construir a estrutura da página Web que irá abrigá-las, o que é feito por meio do HTML. O HTML dá estrutura para uma página Web, declarando formulários, *links*, botões e outros elementos que possam estar presentes na interface por meio de *tags*. Cada *tag* pode ter uma gama de atributos botões, por exemplo, podem ter alguma ação disparada quando são clicados por meio do atributo `onClick`. Cada *tag* é representada graficamente por diferentes *browsers* de uma formas distintas, e utiliza-se então o CSS para dar um estilo próprio e único a cada um dos elementos (OPPERMANN, 2002; STEFANER et al., 2009).

O CSS possibilita a estilização de diversas propriedades de cada elemento, como cores, tipografia, posicionamento e disposição na tela. Ainda é possível estilizar os elementos em diferentes estados que eles possam se encontrar, como quando o mouse os clica ou quando somente passa sobre eles. Além disso, é possível estilizar um grupo de elementos baseado em relações de hierarquia por meio de seletores CSS. Também é comum a utilização de pré-processadores de CSS como o SASS, que incluem a utilização de variáveis e uma sintaxe simplificada, por exemplo, para aumentar o desempenho e extender as funcionalidades do CSS (STEFANER et al., 2009).

Por fim, o JavaScript é uma linguagem de programação de propósito geral que busca dar dinamismo para as páginas Web. Por meio dessa linguagem, é possível configurar as ações provenientes do disparo do botão mencionadas anteriormente, além de modificar o comportamento de elementos da página. Embora seja possível desenvolver interfaces de usuário somente por meio das três tecnologias, pode-se fazer uso de algum *framework* JavaScript durante esse processo, como o AngularJS (GOOGLE, 2017; OPPERMANN, 2002).

O AngularJS é um *framework* JavaScript que busca simplificar o desenvolvimento Web, possibilitando a utilização de uma arquitetura *Model-View-Controller* (MVC) em uma aplicação Web. Criado por Hevery e Abrons em 2009, o *framework* hoje encontra-se na versão 1.7, possuindo ainda uma versão 2 em fase de testes. Por meio do AngularJS, é

possível conectar os dados exibidos na interface com um modelo constituído de variáveis e um controlador codificado em JavaScript que define métodos e lógica para a interface. A essa conexão dá-se o nome de *data binding*, e uma das principais características do *framework* em sua primeira versão é possibilitar o *two-way data binding*, isso é, ocorre a conexão tanto do modelo com a interface quanto da interface com o modelo (GOOGLE, 2017).

Além do *data binding*, o AngularJS também disponibiliza outras funcionalidades a fim de aumentar o dinamismo em aplicações Web, como diretivas, templates e filtros que podem ser incorporados diretamente no HTML por meio de atributos ou dentro de um arquivo JavaScript que contenha a lógica da interface (GOOGLE, 2017). Todas essas funcionalidades contribuem para o desenvolvimento de um projeto legível, com fácil manutenção e reutilização de código, sendo todas elas utilizadas nas atividades realizadas durante a área de UI do Programa de Estágio, mostrada na Seção 4.2 desse trabalho.

4 Atividades desenvolvidas

As atividades desenvolvidas durante o período do Estágio Supervisionado são descritas nesta seção. Primeiramente, uma aplicação construída para migrar dados de uma banco para outro por meio da linguagem Java é descrita, fazendo uso de conceitos apresentados na Seção 3.1. Posteriormente, um projeto utilizando o *framework* Angular é apresentado, fazendo uso dos conceitos de UI apresentados na Seção 3.2.

4.1 Migração de bancos de dados em Java

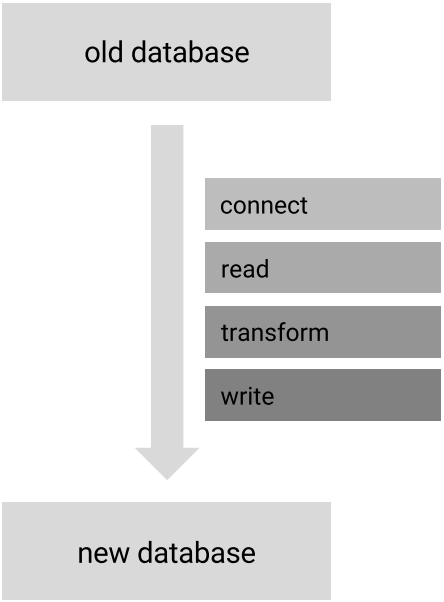
O Avenue Code Events Billing Administration (AC-EBA) é um projeto interno da empresa que visa o gerenciamento de gastos monetários e de pessoal em eventos realizados pela Avenue Code. Originalmente desenvolvido na filial de São Paulo, uma segunda versão do sistema começou a ser projetada a fim de adicionar funcionalidades e melhorar o seu desempenho. A nova versão do AC-EBA começou a ser construída com novas linguagens de *backend* e *frontend* e, portanto, não foi desenvolvida sobre o código da primeira versão. Dessa forma, era necessário migrar os dados da primeira versão para a segunda, e optou-se por construir uma aplicação em Java para auxiliar nesse processo.

O Java Database Connectivity (JDBC) é uma *application programming interface* (API) da linguagem da Java que possibilita gerenciar banco de dados relacionais por meio da abstração da implementação de bancos de dados específicos, criando uma camada intermediária com diversos métodos para o estabelecimento de conexões, leitura, escrita de dados, e extração de resultados a partir de *queries*. Utilizando diversos métodos providos pelo JDBC, uma aplicação para a migração dos dados do banco da primeira versão do AC-EBA foi desenvolvida, e a ela foi dado o nome de AC JDB Migrator (AC-JDBM).

O AC-JDBM foi projetado em uma *pipeline* de quatro etapas diferentes, mostradas na Figura 2. As etapas da *pipeline* dessa aplicação são: (i) conectar-se ao banco de dados do projeto original, (ii) ler os dados presentes nesse banco, (iii) transformar os dados de acordo com as tabelas presentes no novo sistema, e (iv) utilizar esses dados transformados para a escrita das *queries* para popular o banco de dados do novo sistema.

A primeira etapa da aplicação é constituída pela classe `DbConnector`, que busca realizar uma conexão com um banco de dados. Para isso, ela recebe um arquivo de extensão `Property` como parâmetro, que contém as credenciais necessárias para a conexão com um banco. A classe foi feita de modo genérico, sendo que é possível criar um objeto para conectar-se tanto com o banco de dados original quanto o do novo sistema, e métodos de conexão do JDBC foram utilizados em seu desenvolvimento.

Figura 2 – Esquema de execução da aplicação AC JDB Migrator.



Fonte: Próprio autor.

A etapa seguinte corresponde à leitura dos dados do banco original, sendo realizada pela classe `DbReader`. *Queries* para a leitura de todas as tabelas presentes no banco original (por exemplo, `select * from EVENT`) foram codificadas, e os resultados dessas *queries* é obtido por meio de objetos do tipo `ResultSet`. Esses objetos foram então mapeados para objetos do tipo `Table`, que correspondiam às tabelas do antigo banco de dados. A classe `Table` é constituída de um `Set` de objetos do tipo `Row`, e tanto `Table` quanto `Row` foram criados para armazenar os dados lidos de acordo com a tabela correspondente e, posteriormente, passá-los para a próxima etapa da *pipeline*.

A terceira etapa consiste na transformação dos dados lidos para se adequarem às tabelas do banco de dados do novo projeto. O `DbTransformer` foi desenvolvido para receber o conjunto de objetos do tipo `Table` correspondente às tabelas do antigo banco, e transformar os dados recebidos de acordo com as tabelas no novo sistema, que diferiam do sistema original por meio da adição e/ou remoção de algumas colunas, por exemplo. Dessa forma, o `DbTransformer` é constituído de outros objetos, como `EventTransformer` ou `BudgetTransformer`, e cada um deles retorna objetos do tipo `Set` contendo uma interface construída para armazenar os dados transformados, a `SQLWritable`.

Finalmente, a última etapa da *pipeline* de migração do dados é responsável por conectar-se ao banco do novo sistema e escrever as *queries* para a inserção dos dados transformados. Para realizar a conexão, foi utilizada uma instância do `DbConnector` com um arquivo `Property` com os dados do banco do novo sistema, e a interface `SQLWritable`

teve o método `toSQL()` implementado em cada uma de suas instâncias. Esse método era responsável por escrever as *queries* para a inserção dos dados de cada uma das tabelas lidas para as novas tabelas.

No desenvolvimento do AC-JDBM foram utilizados diversos conceitos de orientação a objetos, de padrões de projeto e boas práticas de programação. A criação do projeto como uma *pipeline*, a manipulação de interfaces ao invés de tipos concretos, o encapsulamento dos dados e a separação de responsabilidades pelas classes ao longo do projeto são algumas das características que contribuíram não só para o sucesso da aplicação, mas também para o aprendizado do autor na utilização da linguagem Java aplicada ao *backend* de sistemas.

4.2 Desenvolvimento de interfaces de usuário em uma aplicação Web

A fim de promover um produto de um dos clientes da empresa (não revelado por motivos confidenciais), desenvolveu-se um site responsivo fazendo uso do *framework* AngularJS. O início do desenvolvimento das interfaces dá-se com o recebimento do design da interface por uma empresa terceirizada de design. Em posse das telas, é possível iniciar o desenvolvimento do site, construindo elementos para abrigar os elementos presentes nas telas. Inicialmente, os elementos são construídos de maneira estática, ou seja, fazendo uso somente de HTML e CSS, sendo este último utilizado por meio do SASS.

A preocupação em fazer um site responsivo e de comportamento priorizado para dispositivos móveis (*mobile-first*) ocorre desde o estágio de elaboração do design das interfaces, dado que são enviadas telas correspondentes a como o site deve ser exibido em telas de tamanho *desktop*, *tablet* e *mobile*. Entretanto, ao invés de declarar o mesmo elemento para cada tamanho de tela, faz-se uso das *media queries* em CSS para adequar o estilo do elemento de acordo com a tela do usuário, evitando repetição de código e aumentando a facilidade de manutenção e de mudança do mesmo.

Com algumas das interfaces desenvolvidas estaticamente, é então dado início à elaboração da lógica e dos controladores em JavaScript que irão dar dinamismo ao site. Cada seção do site possui um controlador próprio, e é possível injetar dados presentes nesse controlador para serem exibidos na interface. Além disso, em algumas seções do site, faz-se o uso de arquivos de extensão `JSON` para declarar informações que serão exibidas ao invés de programá-las diretamente no controlador da seção, ou no HTML em si. O controlador da seção recebe o arquivo `JSON` por meio de um Serviço, que por sua vez localiza o arquivo presente em uma das pastas do projeto.

Dentre outras ferramenta do AngularJS que foram utilizadas durante o desenvol-

vimento do site estão inclusas as diretivas, que atuam a fim de adicionar comportamento à algumas seções do site. Além dessas seções poderem ser reutilizadas ao longo de outras partes do projeto, as diretivas podem ser ligadas a um determinado elemento da página de três maneiras diferentes: por meio de uma classe, atributo ou por meio de uma *tag* específica.

Diversos conceitos de AngularJS e desenvolvimento de UI foram utilizados ao longo do projeto, buscando também fazer uso de boas práticas de programação ao longo do processo. Dessa forma, o projeto contribuiu para que o autor adquirisse experiência não só na utilização do *framework* AngularJS, mas também no desenvolvimento *frontend* de sistemas em geral.

5 Conclusão

Por abranger dois espectros bem diferentes no desenvolvimento de sistemas, o período do Estágio Supervisionado foi certamente enriquecedor para o autor. Durante o período do Programa de Estágio na área de Java, exercitou-se diversos conceitos atrelados à orientação a objetos e às características dessa linguagem de programação, como o uso de encapsulamento dos dados e a separação de responsabilidades nas classes presentes no projeto. Além disso, trabalhar com uma API amplamente utilizada por programadores Java no desenvolvimento do *backend* de sistemas dá ao autor a possibilidade de sair do básico da linguagem e aprofundar-se em conceitos mais avançados de Java.

O período de UI também propiciou o aprendizado de diversos conceitos no desenvolvimento de interfaces de usuário, desde os mais básicos relacionados ao HTML, CSS e JavaScript, quanto aos mais avançados atrelados à utilização do AngularJS. A utilização de diversas ferramentas disponibilizadas pelo *framework*, como as diretivas e serviços, contribuíram também para um maior aprendizado no âmbito do desenvolvimento de sistemas Web. Ainda é válido ressaltar que tanto no período de UI quanto no período de Java, houve a preocupação com o desenvolvimento dos projetos seguindo boas práticas de programação, reutilizando código quando possível e obedecendo padrões de projeto consolidados.

O autor pode adquirir experiência tanto no desenvolvimento do *backend* quanto no do *frontend* de sistemas ao longo do período do Estágio Supervisionado. Além disso, as experiências de trabalho em equipe vividas e utilização da metodologia ágil ao longo do desenvolvimento são enriquecedoras para o perfil profissional do autor, que terminou o Estágio Supervisionado com um conhecimento mais amplo no desenvolvimento de sistemas na área de Computação.

Referências

AVENUE CODE. *Avenue Code*: Trusted advisors for e-commerce. 2017. <https://www.avenuecode.com/>. Acesso em: 01/05/2017.

BOYARSKY, J.; SELIKOFF, S. *OCA: Oracle Certified Associate Java SE 8 Programmer I*: Study guide exam 1z0-808. 1. ed. Indianapolis, Indiana (EUA): John Wiley & Sons, Inc., 2015.

FOWLER, M.; HIGHSMITH, J. The agile manifesto. *Software Development*, Miller Freeman, Inc., San Francisco, Califórnia (EUA), v. 9, n. 8, p. 28–35, 2001.

GOOGLE. *AngularJS*: Superheroic javascript mvw framework. 2017. <https://angularjs.org/>. Acesso em: 01/06/2017.

OPPERMANN, R. User-interface design. In: *Handbook on information technologies for education and training*. [S.l.]: Springer, 2002. p. 233–248.

REESE, G. *Database programming with JDBC and JAVA*. 2. ed. Sebastopol, Califórnia (EUA): O'Reilly Media, Inc., 2000.

STEFANER, M. et al. User interface design. *Dynamic taxonomies and faceted search*, Springer, p. 75–112, 2009.

WATT, D. A.; BROWN, D. *Java collections*: An introduction to abstract data types, data structures and algorithms. 1. ed. Nova York, Nova York (EUA): John Wiley & Sons, Inc., 2001.