UNIVERSIDADE FEDERAL DE SANTA MARIA
COLÉGIO POLITÉCNICO
CURSO TÉCNICO EM INFORMÁTICA

Junior de Souza

O USO DE APIS NO PROCESSO DE DESENVOLVIMENTO EM LINGUAGEM JAVA: APLICAÇÃO NO DESENVOLVIMENTO DE UM SOFTWARE DE GESTÃO DE INFORMAÇÕES EM PROGRAMA DE EXTENSÃO DA UFSM

Santa Maria, RS
2020

Junior de Souza

O USO DE APIS NO PROCESSO DE DESENVOLVIMENTO EM LINGUAGEM JAVA: APLICAÇÃO NO DESENVOLVIMENTO DE UM SOFTWARE DE GESTÃO DE INFORMAÇÕES EM PROGRAMA DE EXTENSÃO DA UFSM

Trabalho de Conclusão de Curso apresentado no curso Técnico em Informática do Colégio Politécnico, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de **Técnico em Informática.**

Orientador: Prof. Dr. Giani Petri

Santa Maria, RS
2020

Junior de Souza

O USO DE APIS NO PROCESSO DE DESENVOLVIMENTO EM LINGUAGEM JAVA: APLICAÇÃO NO DESENVOLVIMENTO DE UM SOFTWARE DE GESTÃO DE INFORMAÇÕES EM PROGRAMA DE EXTENSÃO DA UFSM

Trabalho de Conclusão de Curso apresentado no curso Técnico em Informática do Colégio Politécnico, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de **Técnico em Informática**.

Aprovado em de de 2020

_____

Giani Petri, Dr. (UFSM)
(Presidente/Orientador)

_____

Banca 1 (UFSM)

_____

Banca 2 (UFSM)

Santa Maria, RS
2020

AGRADECIMENTOS

Gostaria de agradecer a Deus pela oportunidade de concluir este curso. Agradeço também a minha família, que teve muita compreensão durante todo o tempo dedicado aos estudos e a conclusão deste trabalho.

Agradeço aos professores responsáveis pela condução do Curso Técnico em Informática, em especial ao meu orientador Giani, pelas orientações e cobranças necessárias.

Agradeço, também, aos amigos do DTG Noel Guarany, que sempre estiveram presentes, tanto no período de formação, quanto nas horas de lazer que, nas quais, sem dúvidas, foram onde surgiu a ideia desse TCC.

RESUMO

O USO DE APIS NO PROCESSO DE DESENVOLVIMENTO EM LINGUAGEM JAVA: APLICAÇÃO NO DESENVOLVIMENTO DE UM SOFTWARE DE GESTÃO DE INFORMAÇÕES EM PROGRAMA DE EXTENSÃO DA UFSM

AUTOR: Junior de Souza
ORIENTADOR: Dr. Giani Petri

Este trabalho tem como foco principal abordar o uso de APIs no desenvolvimento de software em linguagem Java, sobretudo no que se refere a aplicações desktop. O objetivo deste trabalho é pesquisar e analisar as funcionalidades das APIs existentes para a linguagem Java e aplica-las no desenvolvimento de um software de gestão de informações, para ser usado em um programa de extensão da Universidade Federal de Santa Maria. O presente estudo consiste em pesquisa de caráter descritivo, com resultados tratados de maneira qualitativa, com a coleta de dados em fontes secundárias. A partir da condução do processo de pesquisa foi possível concluir que o uso de APIs no processo de desenvolvimento de um software ajuda o programador a cumprir os requisitos levantados. Muitas vezes, torna simples tarefas que eram complexas, cabendo ao programador usar as APIs conforme a sua necessidade.

Palavras-chave: API. Java. Desktop. Swing. Java SE.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API – *Application Programming Interface*

CRUD – Acrônimo para *Create*, *Read*, *Update* e Delete

DTG – Departamento de Tradições Gaúchas

GUI – *Graphic User Interface*

GUJ – Grupo de Usuários Java

Java EE – Plataforma Java *Enterprise Edition*

Java ME – Plataforma Java *Micro Edition*

Java SE – Plataforma Java *Standard Edition*

JDK – *Java Development Kit*

JFC – *Java Foundation Classes*

JVM – *Java Virtual Machine*

SIG – Sistema de Informações Gerenciais

SMS – *Short Message Service*

SO – Sistema Operacional

UFSM – Universidade Federal de Santa Maria

SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

A linguagem Java foi oficialmente anunciada pela *Sun Microsystems* em uma conferência em maio de 1995 (SANTOS, 2014). Logo, teve grande interesse da comunidade de negócios, pois possuía um potencial muito alto para Web, uma vez que permitia adicionar conteúdo dinâmico às páginas como, por exemplo, interatividade e animações (DEITEL; DEITEL, 2010).

Ao mesmo tempo em que Java é um termo usado para a linguagem de programação, também pode ser usado para definir uma plataforma. Segundo Oracle (2012), a tecnologia Java é de alto-nível e orientada a objetos, possuindo uma sintaxe e estilo particulares podendo ser dividida em quatro plataformas, ou seja, o que cada plataforma disponibiliza como ferramentas de desenvolvimento. As quatro plataformas são: Java Standard Edition (Java SE), Java Enterprise Edition (Java EE), Java Micro Edition (Java ME) e JavaFX.

“Um dos pontos fortes do Java é seu rico conjunto de classes predefinidas que você pode reutilizar em vez de “reinventar a roda”” (DEITEL; DEITEL, 2010, p. 37). Com o uso de bibliotecas já existentes é possível a reutilização de código, tornando o trabalho do programador um pouco mais simples, visto que não é necessário implementar algoritmos que já tenham sido previamente codificados.

Um exemplo de biblioteca muito utilizada é a *Math* (DEITEL; DEITEL, 2010). Ela contém vários métodos para a realização de cálculos matemáticos, como: potenciação, radiciação, funções trigonométricas. Outro exemplo é a classe String, que faz parte do pacote java.lang. Com a classe String é possível manipular textos no Java, sem se preocupar de que forma eles são implementados, simplesmente invocando os métodos que fazem parte dessa classe (SANTOS, 2014).

Hoje em dia, milhares de bibliotecas estão disponíveis para uso nos projetos a serem desenvolvidos. Existem bibliotecas que são disponibilizadas com a plataforma Java, outras provêm de terceiros, ou seja, pessoas ou empresas desenvolvem uma biblioteca e colocam-na à disposição da comunidade em geral para que faça uso de classes e métodos já utilizados e testados, nos mais variados campos de aplicação.

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Este trabalho tem como objetivo geral pesquisar e analisar as funcionalidades das APIs existentes atualmente, para linguagem Java, voltadas para uma aplicação desktop, e exemplificar seu uso no desenvolvimento de um software em um estudo piloto. O software a ser desenvolvido no estudo piloto refere-se a um sistema de gestão de informações em um programa de extensão da Universidade Federal de Santa Maria (UFSM).

1.2.2 Objetivos Específicos

- Pesquisar as APIs existentes para linguagem Java, voltadas para ambiente desktop;
- Analisar o funcionamento de cada API que for selecionada para o estudo piloto;
- Aplicar as APIs pesquisadas no desenvolvimento de um software de gestão de informações;

1.3 METODOLOGIA

Para a realização deste trabalho, será feita uma pesquisa aplicada de caráter descritivo, a qual, segundo Gil (2009, p. 42) “[...] tem como objetivo primordial a descrição das características de determinada população ou fenômeno [...]”. Essa pesquisa visa analisar as funcionalidades das APIs existentes na linguagem Java, voltadas para uma aplicação desktop, desenvolvendo um software de gestão de informações.

Com isso, os resultados serão apresentados de forma qualitativa, porque segundo Godoy (1995, p. 21) “um fenômeno pode ser melhor compreendido no contexto em que ocorre e do qual é parte, devendo ser analisado numa perspectiva integrada”. A coleta de informações virá de fontes secundárias, que englobam desde buscas no Google e Google Acadêmico, a autores como James Gosling e Herbert Schildt. A consulta realizada na internet usará palavras-chave, dentre elas: java, api, desktop, swing, java se. Além do buscador do Google, serão analisados fóruns de discussão sobre programação, por conterem um vasto acervo sobre o tema, como, por exemplo, o Stack Overflow e GUJ, sites sobre a linguagem Java, como o Java Code Geeks e o próprio site da Oracle, responsável pela supervisão da linguagem Java.

2 APIs NO DENSENVOLVIMENTO DE SOFTWARE USANDO A LINGUAGEM JAVA

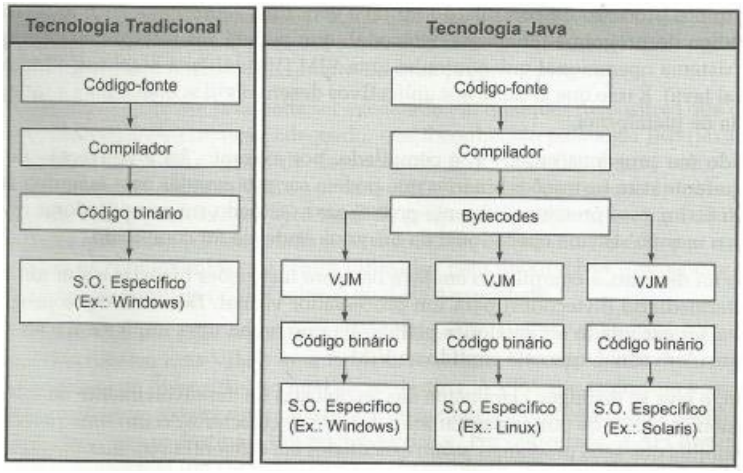
2.1 A LINGUAGEM DE PROGRAMAÇÃO JAVA

Segundo Gosling et al. (2020, p. 17),

> Java é uma linguagem de programação de propósito geral, baseada em classes e orientada a objetos. Foi projetada para ser simples o suficiente para que muitos programadores pudessem obter fluência na linguagem.

A terminologia "Escreva uma vez, execute onde quiser" (tradução literal para a sentença em inglês *write once, run anywhere*) foi usada pela *Sun Microsystems* para descrever uma das características da linguagem, a portabilidade (W3SCHOOLS, 2018). Para que essa portabilidade seja possível, o Java, diferentemente de outras linguagens, não gera ao final de sua compilação um código binário que é interpretado pelo Sistema Operacional (SO). Ao invés disso, a linguagem Java produz os chamados *bytecodes* que serão interpretados pela *Java Virtual Machine* (JVM), uma máquina virtual disponibilizada pela plataforma Java (SCHILDT, 2006). Este processo pode ser visto na Figura 1.

Figura 1 - Execução de Programa Tradicional x Execução de programa Java



Fonte: (SANTOS, 2014, p. 7)

As plataformas são compostas, para uso como desenvolvedor, de um compilador - o javac, de um ambiente de execução - a JVM, e de uma biblioteca de

classes e interfaces Java. O que diferencia uma plataforma da outra, são as classes e interfaces que estão disponíveis.

O Java, como plataforma, disponibiliza uma grande quantidade de classes já implementadas para uso do programador, bastando a ele apenas referenciar essas classes, através da declaração *import*, e usar seus métodos. As classes são agrupadas no que é chamado no Java de pacote, ou *package* (DEITEL; DEITEL, 2010).

2.2 APPLICATION PROGRAMMING INTERFACE (API)

Basicamente, uma API nada mais é que um conjunto de classes e interfaces.

> A API do Java é composta por dois tipos de recursos distintos: classes e interfaces. Ela é como um bloco dividido em duas grandes partes[...]. São centenas de interfaces e milhares de classes que acompanham o kit de desenvolvimento do Java e que podem ser empregadas para a realização de diversos tipos de tarefas durante a construção do programa (SANTOS, 2014 p. 110).

Para Ciriaco (2009), API é um padrão de programação, o qual permite a construção de aplicações, nas quais a utilização da API não fica evidente para o usuário final. “API é a “matriz” dos aplicativos, ou seja, uma interface que roda por trás de tudo”.

Ainda, segundo Garcia (2015), “API é um grupo de programas de suporte destinados a cumprir funções específicas, que é dividido em diferentes partes funcionais chamadas pacotes”.

A compilação das diversas classes e interfaces, os chamados *packages*, é o que se pode denominar API (DEITEL; DEITEL, 2010). Essas APIs podem vir juntamente com a plataforma Java, ou pode ser feita a aquisição de forma separada. Existem algumas empresas que desenvolvem APIs para que possam ser usadas na construção de softwares, facilitando a vida de quem programa, pois os programadores precisam somente conhecer os métodos que são implementados por suas classes e interfaces. Um exemplo de desenvolvedor de API é o Google, que disponibiliza o código original de suas aplicações, através das APIs.

2.2.1 APIs Nativas

Ao fazer o download das ferramentas de desenvolvimento Java, o chamado *Java Development Kit* (JDK), que é “um conjunto de utilitários cuja a finalidade é a permissão para criação de jogos e programas para a plataforma Java” (DIONISIO, 2013), além de disponibilizar o compilador java, o javac, traz junto uma gama de APIs, as chamadas APIs nativas. Essas APIs na versão 14, a mais recente do Java SE, estão divididas em dois módulos. Esses módulos são chamados através da declaração *import*.jdk, que varia em cada versão da ferramenta, e *import.java*, que trazem as APIs disponíveis na versão 14.

2.2.2 API Swing

A linguagem Java possui, há vários anos, uma API muito importante quando se fala em construção de softwares baseados em *Graphic User Interface* ou, simplesmente, GUI. A API Swing surgiu em meados de 1997, como solução para problemas ocorridos na primeira API gráfica do Java, a *Abstract Window Toolkit* (AWT). A primeira vez que Swing foi liberada para uso, foi como parte da *Java Foundation classes* (JFC), na versão 1.2 do Java. Logo, na versão seguinte do Java, a versão J2SE, Swing foi disponibilizado integrando totalmente o pacote Java (SCHILDT, 2015).

“Os componentes Swing são um rico conjunto de controles gráficos da interface do usuário. Eles são escritos para parecer e se comportar para o usuário de forma idêntica em todos os sistemas, tanto quanto possível” (ARNOLD et al, 2006. p. 591). Ou seja, isso mantém a ideia inicial do Java, de que seria possível escrever o código uma vez, e rodá-lo qualquer plataforma.

Ainda, segundo Santos (2014) os componentes gráficos disponíveis em Swing estão dentro do *package* javax.swing e são, em sua maioria, codificados puramente em linguagem Java. Por serem assim desenvolvidos, são chamados de componentes leves. Diferentemente, os componentes da AWT, que possuem interação direta com o sistema de janelas do SO em que estão rodando, são chamados de componentes pesados.

Para Eckstein (1998) e Zukowski (2005), a API Swing, através de seus componentes, permite a criação de variadas aplicações. É possível criar sistemas

simples, com apenas uma caixa de diálogo com o usuário (JOptionPane), até sistemas complexos, com múltiplas interfaces contendo áreas de texto (JTextArea), menus (JMenu), gerenciadores de *layout*, além de ser possível capturar eventos que acontecem nessas janelas. Tudo isso para facilitar a interação do usuário com o sistema**.**

2.2.3 APIs Adicionais

Para quem é desenvolvedor, trabalhar com as opções mais avançadas da Swing pode se tornar uma tarefa um pouco mais onerosa. Além de que, às vezes, ficam impossibilitados de realizar todas as tarefas necessárias que o sistema exige. Nativamente, com a API Swing não é possível, por exemplo, gerar gráficos ou fazer desenhos na tela. Para que essa funcionalidade seja incluída no sistema é necessário o uso de bibliotecas opcionais ao Java, que necessitam serem adicionadas ao projeto em desenvolvimento.

Algumas dessas bibliotecas são disponibilizadas pela própria plataforma, como é o caso da Java2D, que, segundo Deitel e Deitel (2010 p. 503) “fornece capacidades gráficas bidimensionais avançadas para programadores que requerem manipulações gráficas complexas e detalhadas”. Já outras possibilidades, como a criação de relatórios, não são possíveis sem o uso de APIs desenvolvidas por outros programadores ou empresas.

Na Tabela 1, podem ser vistas algumas APIs encontradas na busca feita para cumprir os requisitos apontados para desenvolvimento do software que foi usado como estudo piloto (conforme será descrito no Capítulo 3). Para essa busca, foram analisadas várias fontes. Duas das principais foram o site Stack Overflow e o site GUJ, onde foram procurados tópicos relacionados com as APIs disponíveis para construção de software em linguagem Java.

O site Stack Overflow reúne usuários que trabalham com diferentes linguagens de programação. Ele funciona no sistema perguntas e respostas. O usuário cadastra uma pergunta que pode ser respondida por todos os demais. Cada pergunta/resposta recebe uma pontuação através de votos dos próprios usuários (RAMIRES, 2011; RAMOS, 2018).

Outra ferramenta, parecida com o Stack Overflow, é o site brasileiro GUJ, acrônimo para Grupo de Usuários Java, que reúne cerca de 218 mil usuários

(CAELUM, 2020). Esse site é como um fórum de discussão, no qual são feitas perguntas sobre determinado tema e os próprios usuários postam as respostas.

Tabela 1 - APIs para desenvolvimento em linguagem Java Desktop

Geração de Relatórios			
TIBICO JasperReports	BCL easyPDF SDK	iText 7 Core	Apache POI
Geração de Gráficos			
JFreeChart	Java VisualVM	Cewolf	
Envio de E-mail			
JavaMail	Mailgun	Commons e-mail	
Gerenciamento de Datas			
JCalendar	java.util.Calendar	java.util.Date	
Arquivos em Nuvem			
Google Drive API	Dropbox Core	Microsoft Graph	
Envio de SMS			
Twilio API	Bulk SMS API	Zenvia	
Outros			
Caelum Stella core			

Fonte: O autor (2020)

2.2.4 Visão geral das APIs encontradas

2.2.4.1 TIBICO JasperReports

Segundo Tibico Software Inc. (2020), a biblioteca JasperReports, que possui código aberto, é a mais popular do mundo. Permite a criação de relatórios renderizados a partir da coleta de dados brutos proveniente de qualquer tipo de fonte. Possui um software, escrito totalmente em Java, que possibilita a edição e visualização dos relatórios.

2.2.4.2 BCL easyPDF SDK

A easyPDF SDK é uma biblioteca que possibilita a conversão de qualquer documento para o formato PDF. Além disso, é possível a conversão de arquivos PDF para formatos conhecidos como .doc ou .docx (formatos disponíveis na suíte de aplicativos Microsoft Office). Isso se dá em apenas algumas linhas de código, aumentando o desempenho de aplicativos desenvolvidos com essa biblioteca, podendo ser tanto do lado servidor como em ambiente desktop (BLC TECHNOLOGIES, 2020).

2.2.4.3 iText 7 Core

“iText 7 Core é a versão mais recente de uma biblioteca para programar documentos PDF em Java ou .NET” (ITEXT GROUP, 2018). Com essa biblioteca é possível criar qualquer tipo de arquivo PDF, podendo ficar atento apenas no conteúdo do documento, tendo acesso a todas as estruturas internas que fazem parte do PDF (ITEXT GROUP, 2018).

2.2.4.4 Apache POI

Conforme Santana (2014), Apache POI é um *framework* que possibilita a manipulação de dados em um documento do pacote Office, da Microsoft. O controle dessa API é feito pela fundação Apache, que lançou em fevereiro de 2020 a sua ultima versão. Essa nova versão melhora o suporte a gráficos no Microsoft Excel, além de trazer novas atualizações de segurança relacionadas aos dados contidos nesses documentos (THE APACHE SOFTWARE FOUNDATION, 2020).

2.2.4.5 JFreeChart

“O JFreeChart é uma biblioteca de gráficos Java 100% gratuita que facilita aos desenvolvedores a exibição de gráficos de qualidade profissional em seus aplicativos” (GILBERT, 2020). Possui dentre suas características uma ampla variedade de gráficos, design flexível, suporte a componentes do Java Swing, suporte a arquivos de imagem, e é um software livre (GILBERT, 2020).

2.2.4.6 Java VisualVM

A Java VisualVM permite visualizar dados detalhados, em forma de gráficos, sobre os aplicativos que estão rodando na JVM, em tempo real. Essa API reúne as informações desses aplicativos e as apresenta para que possam ser usadas, por exemplo, para monitoramento ou solução de problemas (ORACLE, 2018a).

2.2.4.7 Cewolf

A API Cewolf serve para incorporar gráficos em uma página web.

> [...] pode ser usado dentro de um aplicativo web [...] para incorporar gráficos complexos de todos os tipos (por exemplo, linhas, pizza, barra, etc) em uma página web [...]. O Cewolf é baseado no JFreeChart e usa seu mecanismo de renderização para renderizar a imagem final do gráfico. [...] Nenhum arquivo é criado no lado do servidor (JSPIN.COM, 2015).

2.2.4.10 JavaMail

Segundo Oracle (2018b), a API JavaMail possui uma estrutura que funciona independente de qualquer plataforma ou protocolo, permitindo assim, criar aplicativos para envio de e-mail. Essa biblioteca está disponível de forma opcional na plataforma Java SE. Carvalho (2010) nos diz que é possível, através da API JavaMail, “enviar, receber e manipular correio eletrônico [...] e messaging como clientes”.

2.2.4.11 Mailgun

A API Mailgun permite enviar e rastrear e-mails, sejam eles de marketing ou de negócios. Possui fácil integração com o *Simple Mail Transfer Protocol*, também chamado de protocolo SMTP e abstrai alguns detalhes do envio de e-mails em massa (MAILGUN TECHNOLOGIES, 2020).

2.2.4.12 Commons e-mail

Esta API foi construída com base na API JavaMail, porém, a diferença está na simplificação de algumas classes, como a *SimpleEmail* e a *MultiPartEmail*. No

primeiro caso fica permitido o envio de e-mail básico, contendo somente texto. A segunda permite o envio de e-mail de texto, mas com a possibilidade enviar anexos (THE APACHE SOFTWARE FOUNDATION, 2018).

2.2.4.13 JCalendar

JCalendar é uma API que permite a seleção de datas, de forma gráfica e intuitiva. Através dessa biblioteca é possível adicionar componentes que possibilitam a escolha não somente de datas completas, no formato dia/mês/ano, mas, somente ano, somente mês ou somente dia. O uso dessa API traz a alternativa de, também, selecionar uma localidade, ou seja, qual o fuso horário em que se encontra aquela aplicação/escolha de data (TOEDTER.COM, 2016).

2.2.4.14 java.util.Calendar

Calendar é uma classe que compõe o *package* util do Java, que fica na API base. Essa classe é abstrata e “fornece métodos para converter um instante específico no tempo e um conjunto de campos como, YEAR, MONTH, DAY_OF_MONTH e assim por diante.” (ORACLE, 2020a). Ainda, conforme Palmeira (2013), “essa classe pode produzir os valores de todos os campos de calendário necessários para implementar a formatação de data e hora, para uma determinada língua e estilo de calendário”.

2.2.4.15 Java.util.Date

Também pertencente ao *package* util, da API base do Java, essa classe Date não permite a internacionalização, sendo necessário o uso de métodos da classe Calendar. Dependendo do local (geográfico) em que a JVM esteja rodando, as informações podem não coincidir com as de uma JVM rodando em outro local, justamente por não permitir que a data seja formatada em um padrão internacional (ORACLE, 2020b). Palmeira (2013) ainda diz que a maioria dos métodos da classe Date estão classificados como *deprecated*, ou seja, estão depreciados, sendo necessário utilizar os métodos da classe Calendar.

2.2.4.16 Google Drive API

O Google disponibiliza uma API para manipular arquivos armazenados no Google Drive. A API do Google Drive permite, entre outras coisas, gerenciar o armazenamento desses arquivos, além de criar funcionalidades poderosas para a aplicação em desenvolvimento. A comunicação com Google Drive se dá através de tokens acesso (GOOGLE, 2020).

2.2.4.17 Dropbox Core

A API do Dropbox está na sua segunda versão. Essa API da aos desenvolvedores a possibilidade de trabalhar com arquivos armazenados no Dropbox, além de incluir muitas funcionalidades, como pesquisa de texto nos arquivos e compartilhamento avançado. Sua conexão baseia-se no protocolo *OAuth 2.0*, utilizando o cabeçalho de autorização e parâmetros *Uniform Resource Locator* (URL) (DROPBOX, 2020).

2.2.4.18 Microsoft Graph

Através dos pacotes de APIs Microsoft Graph é possível manipular arquivos armazenados na nuvem da Microsoft, independentemente de onde estão salvos, seja no *Microsoft Teams*, *SharePoint* ou outro local. É com essa API que se deve trabalhar para ter acesso a eles. Além do mais, com o Microsoft Graph é possível ter acesso aos metadados dos arquivos sem precisar fazer o download do arquivo binário (MICROSOFT, 2020).

2.2.4.19 Twilio API

As APIs da Twilio Inc. são APIs desenvolvidas para possibilitar a comunicação de todo o mundo. Para usufruir dos benefícios, é necessário desembolsar algum valor monetário, dependendo do serviço que for usado (TWILIO INC, 2020).

> As APIs do Twilio alimentam sua plataforma de comunicação. Por trás dessas APIs, há uma camada de software que conecta e otimiza as redes de

> comunicações em todo o mundo para permitir que seus usuários liguem e enviem mensagens para qualquer pessoa, globalmente (TWILIO INC, 2020).

2.2.4.20 Bulk SMS API

Da mesma forma que as APIs da Twilio Inc., a API Bulk SMS necessita de investimento financeiro para funcionar perfeitamente, visto que é necessário pagar para poder enviar uma mensagem, *Short Message Service* (SMS), para um celular. O uso dessa API permite mensagens Unicode e o uso de vários *web hooks* para criar e customizar os fluxos de trabalho (CELERITY SYSTEMS, 2020).

2.2.4.21 Zenvia

As APIs desenvolvidas pela Zenvia permitem, de maneira paga, o envio de SMS para qualquer operadora de telefonia no Brasil. Essa API dá a possibilidade de receber o status e as respostas para as mensagens enviadas. Além do mais, é possível enviar mensagens via aplicativo *WhatsApp* (ZENVIA, 2020).

2.2.4.22 Caelum Stella Core

Quando se requer algum documento que tenha formatação padrão e alguma forma de validação, se faz necessário, de alguma forma, validar esses dados.

> Alguns processos são comuns e recorrentes em sistemas voltados para o público brasileiro. O Stella vai te auxiliar nesse caso, oferecendo uma API simples para validação e formatação de CPF, CNPJ e outros cadastros comuns no Brasil (CAELUM, 2011).

Em síntese, quando o assunto é APIs para desenvolvimento Java, encontra-se uma gama de possibilidades. Existem APIs com as mais variadas funções. Assim o que deve ser levado em conta nessa escolha é a familiaridade que a equipe de desenvolvimento já tem com determinadas APIs, ou ainda, a facilidade de uso que determinada API apresenta. Além disso, a seleção deve se basear, também, pensando no cumprimento dos requisitos definidos pelo cliente.

3 APLICAÇÃO DE APIs NO DESENVOLVIMENTO DE UM SISTEMA DE GESTÃO DE INFORMAÇÕES

Com o objetivo de exemplificar a aplicação de algumas APIs Java encontradas na literatura, é realizado um estudo piloto para o desenvolvimento de um sistema de gestão de informações em um programa de extensão da Universidade Federal de Santa Maria (UFSM).

3.1 DEFINIÇÃO DO CONTEXTO

O Departamento de Tradições Gaúchas (DTG) Noel Guarany é um programa de extensão institucional da Universidade Federal de Santa Maria (UFSM), vinculado à pró-reitoria de extensão. Fundado em 2005, o DTG completa, no ano de 2020, 15 anos (UFSM, 2020).

Desde a sua fundação, o DTG possui registro das pessoas que participam ou participaram da entidade. Esses participantes são chamados, internamente, de sócios. Para fazer esse registro, é usada uma ficha impressa, a qual o sócio preenche com seus dados e, após, é armazenada em um arquivo físico, no próprio DTG.

Para facilitar o gerenciamento e manipulação dos dados dos associados, foi solicitado, pelos componentes do conselho diretor, o desenvolvimento de um software que colaborasse com essa demanda.

3.2 REQUISITOS DO SISTEMA

Segundo Guedes (2011, p. 22),

> as etapas de levantamento e análise de requisitos trabalham com o domínio do problema e tentam determinar “o quê” o software deve fazer e se é realmente possível desenvolver o software solicitado. Na etapa de levantamento de requisitos, o engenheiro de software busca compreender as necessidades do usuário e o que ele deseja que o sistema a ser desenvolvido realize. Isso é feito sobretudo por meio de entrevistas, nas quais o engenheiro tenta compreender como funciona atualmente o processo a ser informatizado e quais os serviços o cliente precisa que o software forneça.

Para o levantamento de requisitos, foram realizadas entrevistas com o grupo que compõe o conselho diretor do DTG Noel Guarany, buscando a coleta colaborativa.

Além do mais, foram feitas observações, *in loco*, de como é feito o registro dos sócios. Essas técnicas são descritas por Sommerville e Sawyer (1999).

Segundo Wazlawick (2011), é possível dividir os requisitos em funcionais, não funcionais e requisitos suplementares. Após essa análise, é possível elaborar um documento, chamado de documento de requisitos. “O documento de requisitos registra todos os tópicos relativos ao que o sistema deve fazer e sob quais condições. [...] e assume-se que não será completo em profundidade [...]” (WAZLAWICK, 2011, p. 26). Na Figura 2, é possível analisar os requisitos funcionais do Sistema de Informações Gerenciais (SIG) – DTG Noel Guarany. O documento completo está no Apêndice A.

Figura 2 - Documento de Requisitos - SIG DTG Noel Guarany

Sistema de Gestão de Informações – DTG Noel Guarany – Documento de Requisitos

Requisitos Funcionais

1. Registrar novos sócios;
2. Registrar novos centros de ensino;
3. Registrar novos cursos;
4. Registrar novos departamentos;
5. Registrar novas invernadas;
6. Registrar novas viagens;
7. Emitir relatórios referente a sócios;
8. Emitir relatórios referente a viagens;
9. Enviar e-mails para sócios ou grupo de sócios;
10. Salvar arquivos em nuvem;
11. Gerar gráficos estatísticos referentes a sócios;

Fonte: O autor (2020)

3.2.1 Escolha das APIs

Com base nos dados coletados na fase de análise e levantamento de requisitos foi possível determinar quais APIs serão usadas no desenvolvimento do sistema de gestão. Para se trabalhar com datas, foi feita a escolha pela API JCalendar. Para a parte que trata de validação de dados, foi escolhida a API Caelum Stella Core. No que diz respeito ao envio de e-mail e armazenamento de arquivos em nuvem, foram

escolhidas as APIs Commons E-mail e Dropbox Core, respectivamente. Quanto a emissão de relatórios optou-se pela utilização da API TIBICO JasperReports. A API utilizada para geração de gráficos foi a JFreeChart. A opção pelo não uso de envios de SMS se dá em virtude da necessidade de investimento financeiro mensal, o que foi descartado pelo DTG Noel Guarany.

3.3 MODELAGEM DO SISTEMA

A modelagem de software tem o intuito de obter uma visão abstrata de um sistema físico. Essa visão abstrata tem o objetivo de descrever aspectos estruturais e comportamentais do software (GUEDES, 2011).

Booch et al. (2006, p. 35) diz que

> Os modelos fornecem uma planta do projeto de um sistema. Os modelos poderão abranger planos detalhados, assim como planos mais gerais com uma visão panorâmica do sistema considerado. Um bom modelo inclui aqueles componentes que têm ampla repercussão e omite os componentes menores que não são relevantes em determinado nível de abstração. Todos os sistemas podem ser descritos sob diferentes aspectos, com a utilização de modelos distintos, e cada modelo será, portanto, uma abstração semanticamente específica do sistema.

Ainda segundo Booch et al. (2006), existem algumas linguagens-padrão para a elaboração de estruturas de projetos de software, ou modelo. Uma delas é a *Unified Modeling* Language (UML). A modelagem de um sistema pode ser vista sob diferentes perspectivas através de modelos distintos, sendo que essas visões são expressas por meio de diagramas. Esses diagramas, conforme a Tabela 2, estão divididos em duas partes: estática ou estrutural e dinâmica ou comportamental.

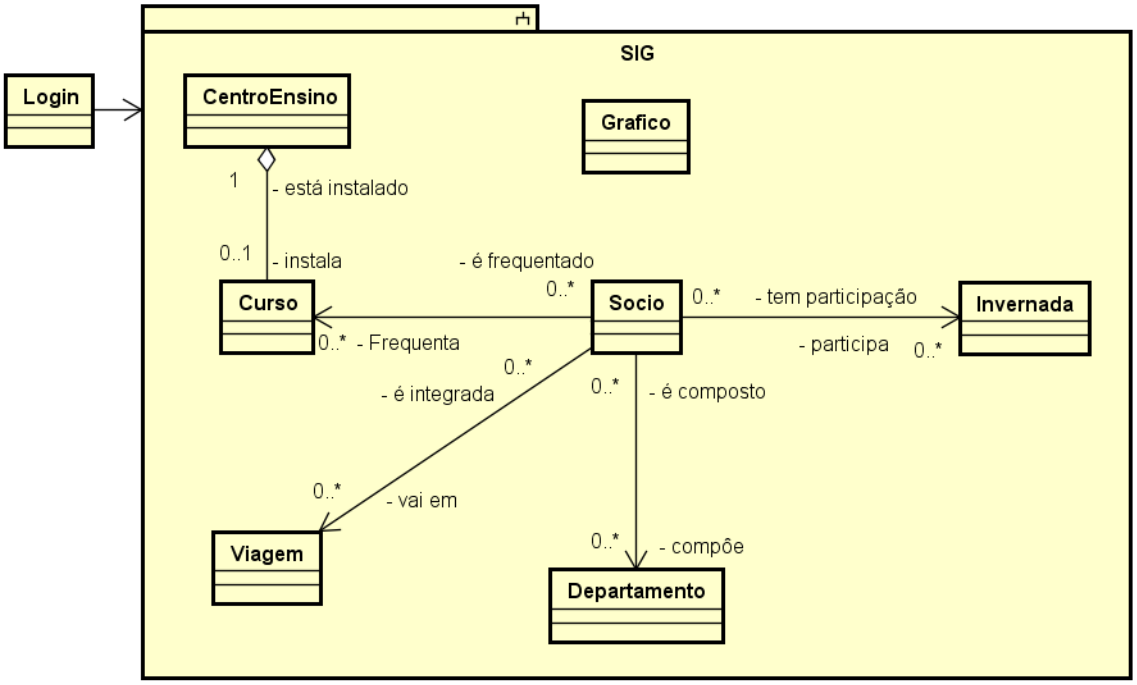
Para a modelagem do sistema usado no estudo piloto, foi escolhido o diagrama de classes para representar a parte estática e o diagrama de casos de uso para a parte dinâmica. Sommerville (2011, p. 90) nos diz que “os diagramas de classe em UML podem ser expressos em diferentes níveis de detalhamento”. Na Figura 3 podemos ver de forma mais simples o diagrama de classes e no Apêndice B podemos ver, de forma completa, o diagrama de classes desse sistema.

Tabela 2 - Diagramas estruturais x diagramas comportamentais

DIAGRAMAS ESTRUTURAIS	DIAGRAMAS COMPORTAMENTAIS
Diagrama de classes	Diagrama de casos de uso
Diagrama de componentes	Diagrama de sequência
Diagrama de estrutura composta	Diagrama de comunicação
Diagrama de objetos	Diagrama de estados
Diagrama de artefatos	Diagrama de atividades
Diagrama de implementação	

Fonte: O autor (2020)

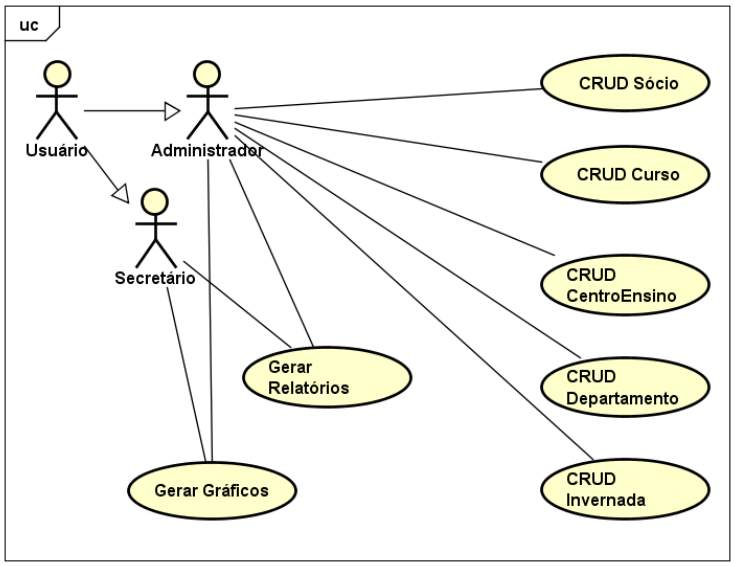
Figura 3 - Diagrama de classes simples com associações



Fonte: O autor (2020)

Booch et al. (2006, p. 347) define caso de uso como “um conjunto de sequências de ações, inclusive variantes, que um sistema executa para produzir um resultado de valor observável por um ator”. Um caso de uso geral do sistema piloto pode ser visto, de maneira simplificada, na Figura 4 e, de maneira um pouco mais detalhada, no Apêndice C.

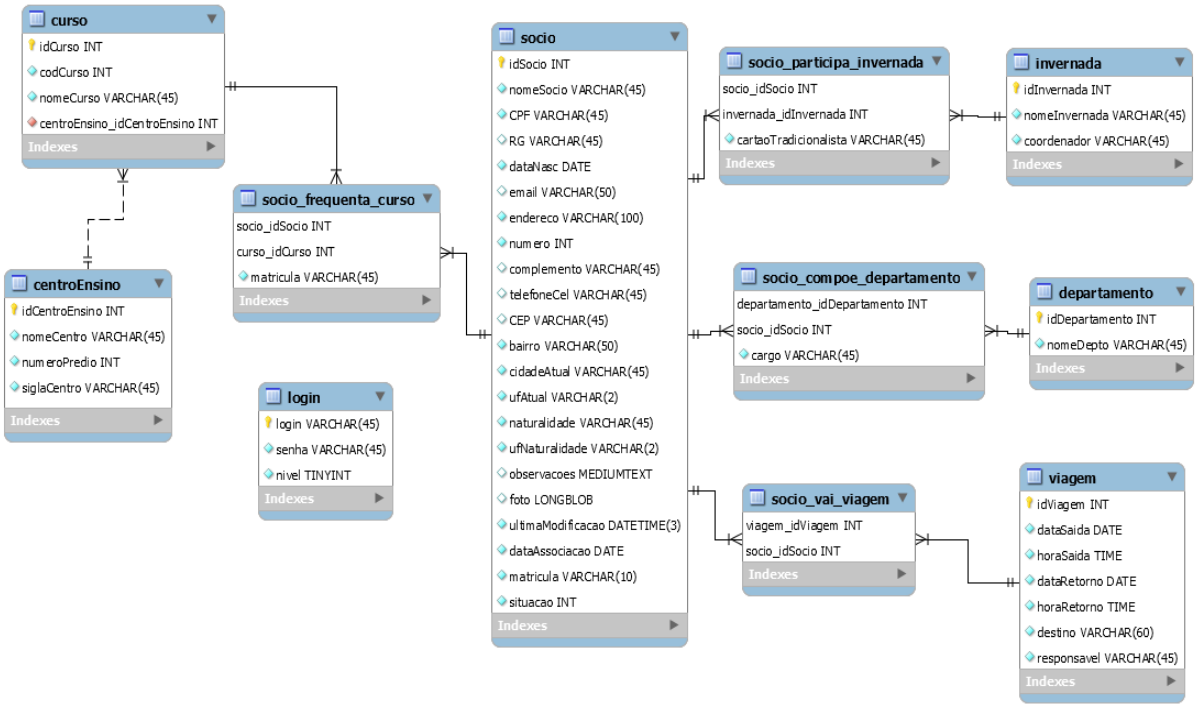
Figura 4 - Diagrama de caso de uso simples



Fonte: O autor (2020)

O Sistema de Gerenciamento de Banco de Dados (SGBD) que será usado nesse projeto é o MySQL. O modelo do banco de dados usado pode ser visto no diagrama Entidade-Relacionamento Estendido (EER) da Figura 5.

Figura 5 - Diagrama entidade relacionamento estendido - EER



Fonte: O autor (2020)

3.4 DESENVOLVIMENTO

Nesta seção serão apresentadas as principais interfaces do sistema desenvolvido, nas quais foram aplicadas cada uma das API escolhidas para o desenvolvimento. O sistema possui diferenciação de níveis de usuários que, dependendo do nível, restringe o uso de algumas funções.

Existem dois níveis de perfis que foram criados. O perfil “secretário”, que tem funções limitadas dentro do sistema, não sendo possível, por exemplo, criar novos usuários ou realizar alguns cadastros. Já o perfil de nível “administrador” tem acesso a todas as funcionalidades do sistema. Foi com o perfil de administrador logado que foram feitas as imagens apresentadas nessa seção.

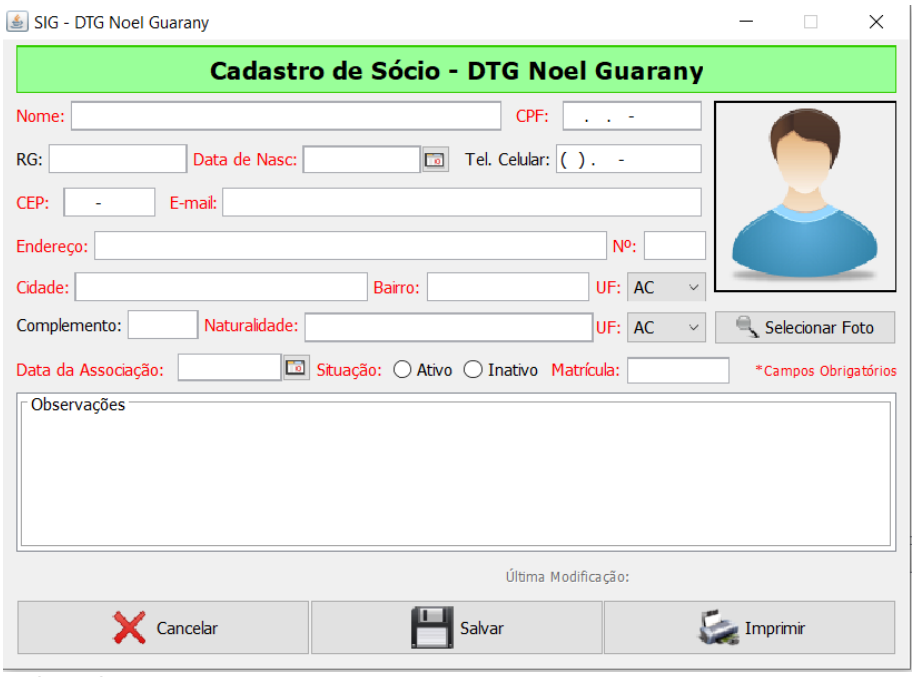
Após fazer o login, é apresentada ao usuário a interface principal do sistema, a qual dá acesso as demais funcionalidades. Essa interface está dividida em atalhos (botões) para as funções mais utilizadas do menu e em menus propriamente ditos, que apresentam o que é possível fazer no sistema. A interface principal pode ser visa na Figura 6.

Figura 6 - Interface principal SIG DTG Noel Guarany

Fonte: O autor (2020)

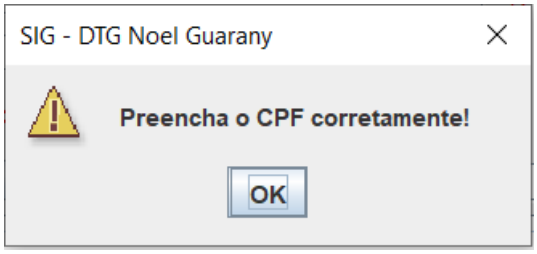
Uma das principais interfaces do sistema é a de cadastramento dos sócios da entidade. Esse cadastro teve por base a ficha de sócio que é preenchida de forma manual e pode ser vista no Anexo A. A interface traz um formulário um pouco mais elaborado, que solicita dados do sócio, sendo um desses dados o Cadastro de Pessoa Física (CPF). O CPF possui algumas formas de verificação, que são feitas através de cálculos matemáticos. Para facilitar essas checagens, foi usada a API Caelum Stella Core, que faz todas essas verificações retornando se o CPF é valido ou não. No momento em que o usuário clica no botão salvar, caso o CPF não seja válido, é informado ao usuário que o CPF precisa ser preenchido de forma correta. Nas Figuras 7 e 8, podem ser vistas a interface de cadastro e a mensagem de erro, respectivamente. Na figura 9, é possível ver parte do código onde se faz o uso de métodos dessa API.

Figura 7 - Interface de cadastro de sócios com uso da API Caelum Stella Core



Fonte: O autor (2020)

Figura 8 - Mesagem de erro de CPF com uso da API Caelum Stella Core



Fonte: O autor (2020)

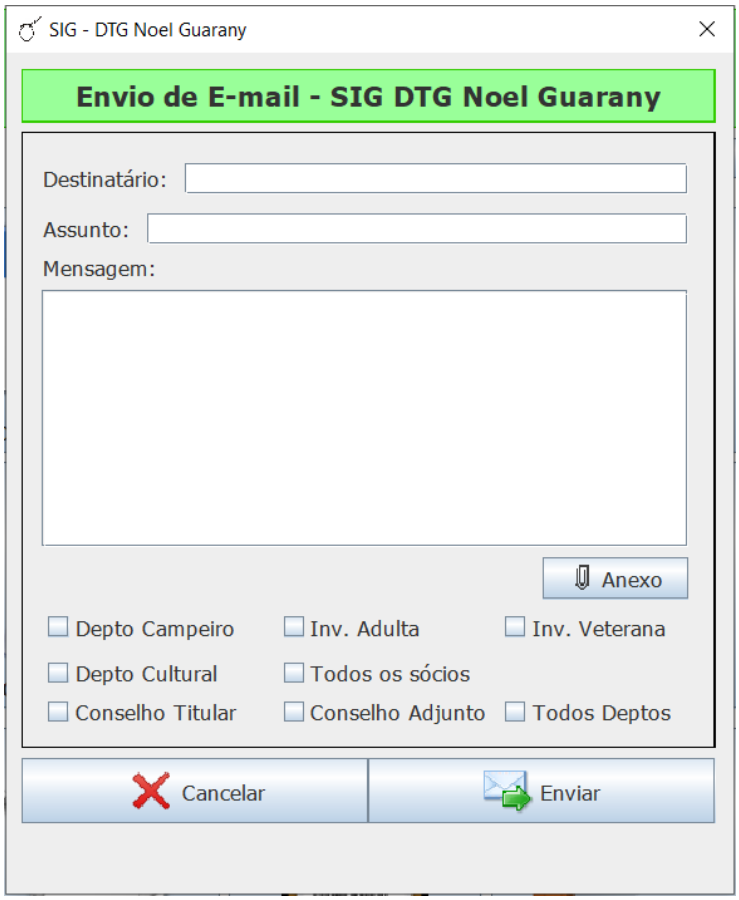
Figura 9 - Uso de métodos da API Caelum Stella Core

```
public static boolean validaCPF(String CPF){
    CPFValidator cpfValidator = new CPFValidator();
    List<ValidationMessage> erros = cpfValidator.invalidMessagesFor(CPF);
    return erros.size() <= 0;
}
```

Fonte: O autor (2020)

Uma das funcionalidades exigidas no levantamento de requisitos foi a possibilidade de envio de e-mails. Para o desenvolvimento dessa função foi feita a escolha da API Commons E-mail, que se baseia na API JavaMail. A Commons E-mail facilita o desenvolvimento, pois possui uma codificação mais simples, sem ser necessário configurar muitos parâmetros, como é o caso da JavaMail. Com a API Commons E-mail, fica fácil enviar tanto um e-mail com apenas uma mensagem de texto, quanto uma mensagem contendo um anexo. A interface de envio de e-mail pode ser vista na Figura 10. Já na figura 11 é possível ver os métodos necessários para fazer o envio de um e-mail sem anexo.

Figura 10 - Interface de envio de e-mail com uso da API Commons Email



Fonte: O autor (2020)

Figura 11 - Uso de métodos da API Commons Email

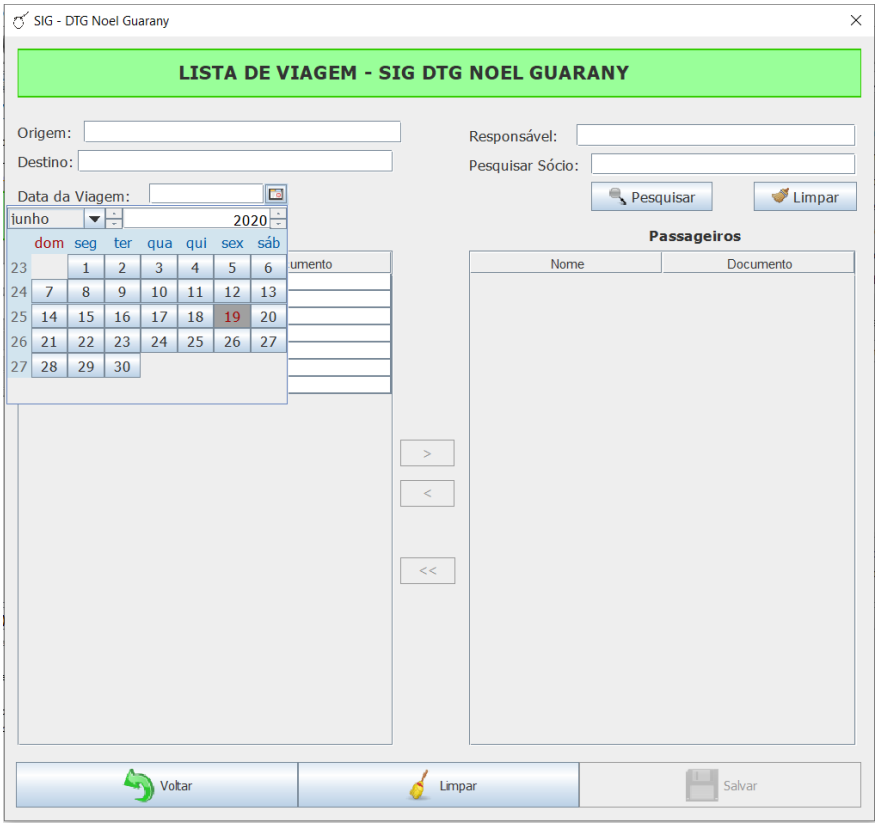
```
public static boolean enviarEmail(String[] destinatario, String assunto, String mensagem){

  SimpleEmail email = new SimpleEmail();
  try {
    email.setDebug(true);
    email.setHostName("[redacted]"); // o servidor SMTP para envio do e-mail
    for(int i=0;i<destinatario.length;i++){
        email.addTo(destinatario[i]); //destinatários
    }
    email.setFrom("[redacted]", "Souza"); // remetente
    email.setSubject(assunto); // assunto do e-mail
    email.setMsg(mensagem); //conteudo do e-mail
    email.setAuthentication("[redacted]");
    email.setSmtpPort(587);
    email.setSSL(true);
    email.setTLS(true);
    email.send();
    return true;
  } catch (EmailException ex) {
      System.err.println("Erro ao enviar email simples: "+ ex);
      return false;
  }
}
```

Fonte: O autor (2020)

Para o gerenciamento de datas, foi usada a API JCalendar. Essa API disponibiliza alguns elementos gráficos que facilitam ao usuário o preenchimento de datas. O elemento gráfico JDateChooser disponibilizado por essa biblioteca gera um calendário gráfico na tela, no qual é possível selecionar o dia, o mês e o ano que se quer. Também é possível simplesmente escrever os dados dentro desse campo, separados por uma barra “/”. A aplicação dessa funcionalidade pode ser vista em algumas interfaces, como no cadastro de sócios ou no cadastro de viagens. O exemplo da Figura 12 é a interface de criação de uma lista de viagem. O método usado para obter a data selecionada no elemento gráfico pode ser visto em destaque na figura 13.

Uma das formas existentes para guardar arquivos digitais é o armazenamento em nuvem. Essa funcionalidade requisitada está presente no sistema piloto através da biblioteca Dropbox Core, que permite o gerenciamento dos arquivos salvos nos servidores da empresa Dropbox. Na Figura 14, é possível visualizar a opção de download dos arquivos que estão armazenados, divididos por categorias. A lista é carregada automaticamente dos servidores do Dropbox, assim que clicado na categoria escolhida. Na Figura 15 é possível visualizar alguns dos métodos usados para cumprir esse requisito do sistema.

Figura 12 - Interface de cadastro de viagem com uso da API JCalendar



Fonte: O autor (2020)

Figura 13 - Uso de métodos da API JCalendar

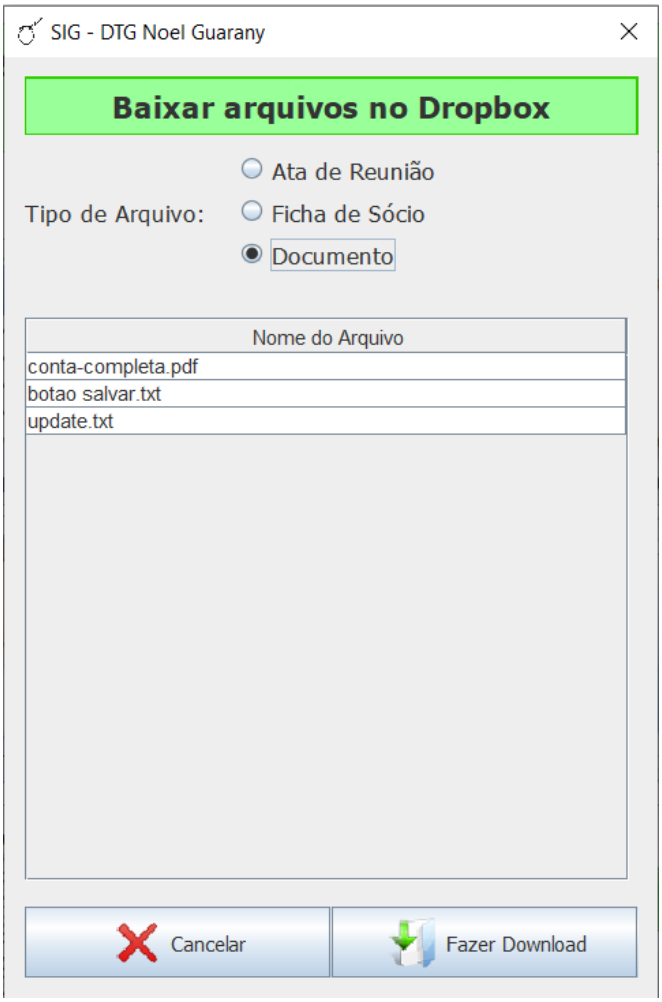
```
if (editar == false) {
    int idViagem = viagemCTRL.create(
            ConversaoDatas.convUtilDateToSQLDate(jDtChSaida.getDate()),
            ConversaoDatas.convUtilDateToSQLDate(jDtChRetorno.getDate()),
            ConversaoDatas.convStringToSQLTime(jTxtHoraSaida.getText()),
```

Fonte: O autor (2020)

Esses trabalhos são publicados em congressos, fóruns, eventos em geral e se faz necessária uma forma de quantificar as informações desses trabalhos. Pensando nisso, a funcionalidade de geração de gráficos facilita a obtenção desses dados.

A API JFreeChart foi escolhida para poder gerar, de forma simples, os mais diversos gráficos referentes aos dados inseridos no sistema. Um deles é o gráfico em setores, apresentado na Figura 16, que representa a idade dos sócios do DTG Noel Guarany. A Figura 17 mostra métodos usados para a construção dos gráficos.

Figura 14 - Download de arquivos com o uso da API Dorpbox Core



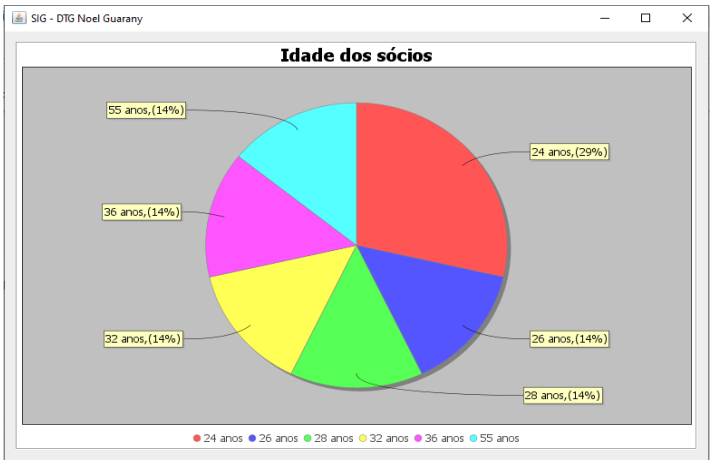
Fonte: O autor (2020)

Figura 15 - Uso de métodos da API Dropbox Core

```
public boolean baixarArquivo(String caminhoDestino, String caminhoBuscar){
    boolean situacao=false;
    try{
          OutputStream downloadFile = new FileOutputStream(caminhoDestino);
          try{
              FileMetadata metadata = cliente.files().downloadBuilder(caminhoBuscar).download(downloadFile);
              situacao = true;
          }
          catch (DbxException ex) {
           Logger.getLogger(Dropbox.class.getName()).log(Level.SEVERE, null, ex);
          }
          finally{
              downloadFile.close();
          }
       }
       //exception handled
    catch (IOException e){
          JOptionPane.showMessageDialog(null, "Unable to download file to local system\n Error: " + e);
          situacao =  false;
       }
    return situacao;
```

Fonte: O autor (2020)

Figura 16 - Geração de gráficos com o uso da API JFreeChart



Fonte: O autor (2020)

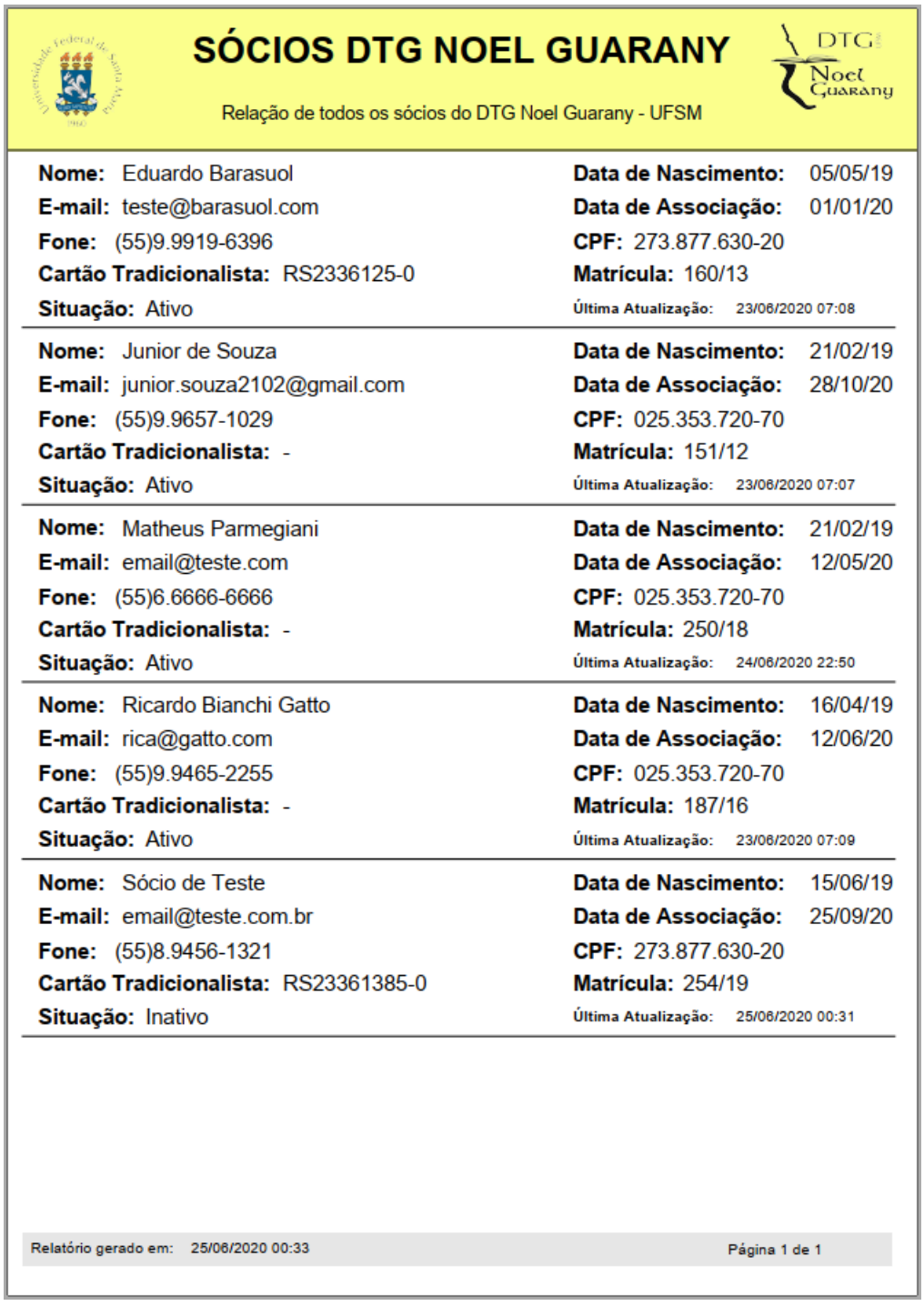
Figura 17 - Uso de métodos da API JFreeChart

```
JFrmGraficos jfrmGraficos = new JFrmGraficos();
DefaultPieDataset dpd= new DefaultPieDataset();
GraficosDAO gd = new GraficosDAO();
ArrayList<Grafico> dadosGrafico = gd.readIdadeSocios();
for (Grafico graficoMOD : dadosGrafico) {
    dpd.setValue(graficoMOD.getLegenda()+" anos", graficoMOD.getQuantidade());
}
JFreeChart grafico = ChartFactory.createPieChart("Idade dos sócios", dpd, true, true, true);
PiePlot plotagem = (PiePlot) grafico.getPlot();
plotagem.setLabelGenerator(new StandardPieSectionLabelGenerator("{0},({2})"));
ChartPanel chartPanel = new ChartPanel(grafico);
jfrmGraficos.jPnlGrafico.add(chartPanel);
jfrmGraficos.jPnlGrafico.validate();
jfrmGraficos.setLocationRelativeTo(this);
jfrmGraficos.setVisible(true);
```

Fonte: O autor (2020)

Uma das funcionalidades mais importantes definida pelos integrantes do conselho diretor do DTG Noel Guarany, na fase de levantamento de requisitos, foi a geração de relatórios. Um deles é composto por algumas informações dos associados e outro com informações referentes as viagens feitas pelo DTG, especificamente, uma lista de passageiros. Para gera-los com os requisitos levantados, foi usada a API JasperReports, da TIBICO. Essa biblioteca dá a opção de formatação do *layout*, bem como possibilita a escolha de informações diretamente do banco de dados. Os relatórios (com dados fictícios), de sócios e lista de passageiros, podem ser vistos nas figuras 18 e 19 (trocar a imagem 19). Já a Figura 20 mostra parte dos métodos usados para a geração desses relatórios.

Figura 18 - Lista de sócios gerada com a API JasperReport



SÓCIOS DTG NOEL GUARANY

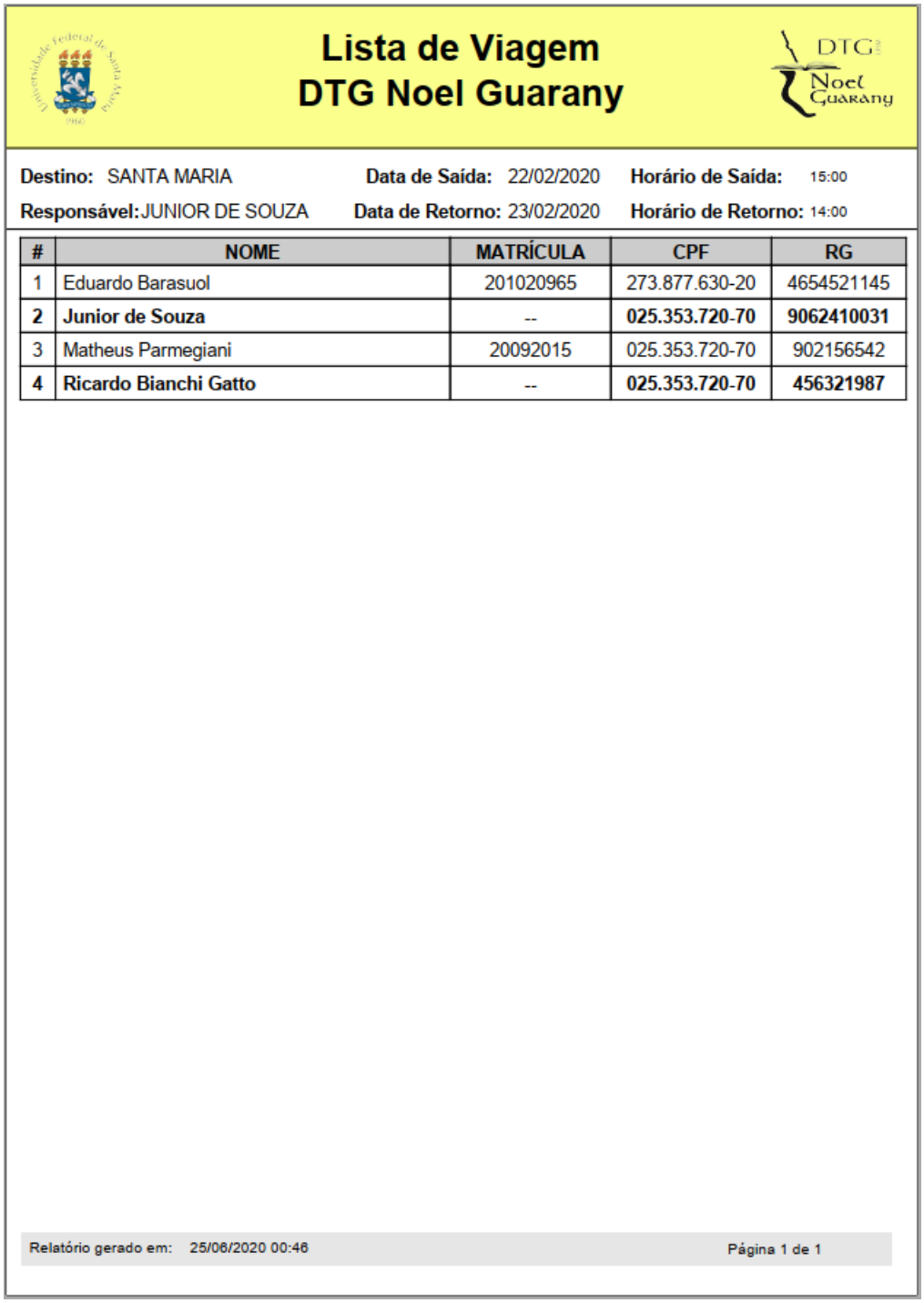
Relação de todos os sócios do DTG Noel Guarany - UFSM

Nome: Eduardo Barasuol	**Data de Nascimento:** 05/05/19
E-mail: teste@barasuol.com	**Data de Associação:** 01/01/20
Fone: (55)9.9919-6396	**CPF:** 273.877.630-20
Cartão Tradicionalista: RS2336125-0	**Matrícula:** 160/13
Situação: Ativo	**Última Atualização:** 23/06/2020 07:08
Nome: Junior de Souza	**Data de Nascimento:** 21/02/19
E-mail: junior.souza2102@gmail.com	**Data de Associação:** 28/10/20
Fone: (55)9.9657-1029	**CPF:** 025.353.720-70
Cartão Tradicionalista: -	**Matrícula:** 151/12
Situação: Ativo	**Última Atualização:** 23/06/2020 07:07
Nome: Matheus Parmegiani	**Data de Nascimento:** 21/02/19
E-mail: email@teste.com	**Data de Associação:** 12/05/20
Fone: (55)6.6666-6666	**CPF:** 025.353.720-70
Cartão Tradicionalista: -	**Matrícula:** 250/18
Situação: Ativo	**Última Atualização:** 24/06/2020 22:50
Nome: Ricardo Bianchi Gatto	**Data de Nascimento:** 16/04/19
E-mail: rica@gatto.com	**Data de Associação:** 12/06/20
Fone: (55)9.9465-2255	**CPF:** 025.353.720-70
Cartão Tradicionalista: -	**Matrícula:** 187/16
Situação: Ativo	**Última Atualização:** 23/06/2020 07:09
Nome: Sócio de Teste	**Data de Nascimento:** 15/06/19
E-mail: email@teste.com.br	**Data de Associação:** 25/09/20
Fone: (55)8.9456-1321	**CPF:** 273.877.630-20
Cartão Tradicionalista: RS23361385-0	**Matrícula:** 254/19
Situação: Inativo	**Última Atualização:** 25/06/2020 00:31

Relatório gerado em: 25/06/2020 00:33 Página 1 de 1

Fonte: O autor (2020)

Figura 19 - Lista de passageiros gerada com a API JasperReport



Lista de Viagem
DTG Noel Guarany

Destino: SANTA MARIA **Data de Saída:** 22/02/2020 **Horário de Saída:** 15:00
Responsável: JUNIOR DE SOUZA **Data de Retorno:** 23/02/2020 **Horário de Retorno:** 14:00

#	NOME	MATRÍCULA	CPF	RG
1	Eduardo Barasuol	201020965	273.877.630-20	4654521145
2	**Junior de Souza**	--	**025.353.720-70**	**9062410031**
3	Matheus Parmegiani	20092015	025.353.720-70	902156542
4	**Ricardo Bianchi Gatto**	--	**025.353.720-70**	**456321987**

Relatório gerado em: 25/06/2020 00:46 Página 1 de 1

Fonte: O autor (2020)

Figura 20 - Uso de métodos da API JasperReport

```
Connection con = ConnectionFactory.getConnection();
String src = "[illegible]/relatorioDadosSocios.jasper";
JasperPrint jasperPrint = null;
 try {
     jasperPrint = JasperFillManager.fillReport(src, null,con);
 } catch (JRException ex) {
     System.err.println("Erro gerar relatorios: "+ ex);
 }

 jFlChRelatorio.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
 String save;
if(jFlChRelatorio.showSaveDialog(this) == JFileChooser.APPROVE_OPTION){
    save = jFlChRelatorio.getSelectedFile().getAbsolutePath()+"\\"+"Relatorio"+ new Date().getTime()+".pdf";
    try {
        JasperExportManager.exportReportToPdfFile(jasperPrint, save);
        Desktop.getDesktop().open(new File(save));
    } catch (JRException ex) {
        Logger.getLogger(JFrmPrincipal.class.getName()).log(Level.SEVERE, null, ex);
    } catch (IOException ex) {
        Logger.getLogger(JFrmPrincipal.class.getName()).log(Level.SEVERE, null, ex);
    }
```

Fonte: O autor (2020)

Conforme visto nesta seção, é possível analisar os trechos de códigos e perceber que, em alguns casos, uma ou duas linhas de instruções bastam para que a funcionalidade seja implementada. Como disse Deitel e Deitel (2010), com o uso de APIs não é preciso “reinventar a roda”, basta, apenas, utilizá-la.

3.5 TESTES

Conforme nos diz Wazlawick (2011), os teste de software podem ser divididos em seis classificações, sendo elas: teste de unidade, teste de integração, teste de sistema, teste de aceitação, teste de operação e teste de regressão. O teste de unidade tem por objetivo verificar se o funcionamento dos métodos mais básicos, individualmente, está correto. Já o teste de integração possibilita saber se a integração dos objetos se dá de forma adequada. O teste de sistema analisa a visão do usuário final, sobre a execução do software. O teste de aceitação é conduzido pelos próprios usuários finais, em interação com o sistema. O teste de operação é feito pelos responsáveis por administrar o sistema, no ambiente final de execução do software. Por fim, o teste de regressão pode ser aplicado quando forem desenvolvidas novas versões do software analisado.

No projeto do sistema para o DTG Noel Guarany os testes foram sendo realizados conforme se dava o desenvolvimento de módulos. Ou seja, foram usados os testes de unidade e de integração para garantir que cada parte do sistema funcionasse de forma independente e, depois, de forma integrada.

Quando o software chegou a um ponto em que várias funcionalidades já estavam testadas, foi lançada, para alguns membros do conselho diretor, uma versão de testes. Nessa versão foram analisados pontos relacionados à interface gráfica e a erros que possam acontecer, caso haja desvio do fluxo principal de execução. Os testes usados para tal foram o de sistema e o de aceitação.

4 CONCLUSÕES

Com base no que foi apresentado ao longo do trabalho, é possível perceber que o uso de APIs acaba facilitando muito o desenvolvimento de um sistema, pois auxilia o programador a não precisar desenvolver códigos muito complexos e nem muito extensos para chegar ao objetivo final, ou seja, cumprir determinado requisito do sistema.

O uso de bibliotecas fornece soluções simples para problemas considerados difíceis, cabendo ao programador apenas utilizá-las da maneira mais adequada no seu projeto. Com um pouco de pesquisa em fóruns sobre programação, ou grupos de usuários referentes à linguagem que se usa, é possível obter inúmeras bibliotecas para determinada função. Cabe ao programador analisá-las e definir qual se encaixa melhor no seu projeto.

Para futuras pesquisas sobre o tema, sugerem-se abordagens que foquem o desenvolvimento para a web, visto que muitas das APIs pesquisadas possuem módulos que são voltados para esse campo da programação.

REFERÊNCIAS

ARNOLD, K.; GOSLING, J.; HOLMES, D. **The Java programming language**. 4th ed ed. Upper Saddle River, NJ: Addison-Wesley, 2006.

BLC TECHNOLOGIES. **BCL easyPDF SDK: a PDF Programming Toolkit for PDF enterprise server and PDF desktop applications.** Disponível em: <http://www.bcltechnologies.com/easypdf/sdk/>. Acesso em: 10 jun. 2020.

BOOCH, G. et al. **UML guia do usuário**. 2. ed. Rio de Janeiro: Elsevier Campus, 2006.

CAELUM. **Sobre**. Disponível em: <https://www.guj.com.br/about>. Acesso em: 5 jun. 2020.

CAELUM, E. E I. **Stella - Simplificando o desenvolvimento no Brasil**. Disponível em: <http://stella.caelum.com.br/>. Acesso em: 11 jun. 2020.

CARVALHO, M. C. **Enviando e-mail com JavaMail utilizando Gmail**. Disponível em: <https://www.devmedia.com.br/enviando-email-com-javamail-utilizando-gmail/18034>. Acesso em: 11 jun. 2020.

CELERITY SYSTEMS. **BulkSMS.com | SMS Gateway to 213 Countries**. Disponível em: <https://www.bulksms.com>. Acesso em: 11 jun. 2020.

CIRIACO, D. **O que é API?** Disponível em: <https://www.tecmundo.com.br/programacao/1807-o-que-e-api-.htm>. Acesso em: 21 maio. 2020.

DEITEL, H. M.; DEITEL, P. J. **Java como programar**. São Paulo (SP): Pearson Prentice Hall, 2010.

DIONISIO, E. J. **Java JDK: Introdução ao Java Development Kit**. Disponível em: <https://www.devmedia.com.br/introducao-ao-java-jdk/28896>. Acesso em: 21 maio. 2020.

DROPBOX. **HTTP - Developers**. Disponível em: <https://www.dropbox.com/developers/documentation/http/documentation>. Acesso em: 11 jun. 2020.

ECKSTEIN, R.; LOY, M.; WOOD, D. **Java Swing**. 1st ed ed. Sebastopol, Calif: O'Reilly, 1998.

GARCIA, V. G. DE A. **Fábrica de Software » API JavaFábrica de Software**, 13 mar. 2015. Disponível em: <http://fabrica.ms.senac.br/2015/03/api-java/>. Acesso em: 9 abr. 2020

GIL, A. C. **Como elaborar projetos de pesquisa**. São Paulo: Atlas, 2009.

GILBERT, D. **JFreeChart**. Disponível em: <http://www.jfree.org/jfreechart/>. Acesso em: 10 jun. 2020.

GODOY, A. S. Pesquisa Qualitativa: Tipos Fundamentais. **Revista de Administração de Empresas**, v. 35, maio 1995.

GOOGLE. **Introduction to Google Drive API | Google Developers**. Disponível em: <https://developers.google.com/drive/api/v3/about-sdk?hl=pt-br>. Acesso em: 11 jun. 2020.

GOSLING, J. et al. The Java® Language Specification. **The Java® Language Specification**, p. 796, 20 fev. 2020.

GUEDES, G. T. A. **UML 2 : Uma abordagem prática**. 2. ed. São Paulo (SP): Novaterra Editora, 2011.

ITEXT GROUP. **iText 7 Core**. Product. Disponível em: <https://itextpdf.com/en/products/itext-7/itext-7-core>. Acesso em: 10 jun. 2020.

JSPIN.COM. **Cewolf - Chart Enabling Web Object Framework**. Disponível em: <http://cewolf.sourceforge.net/new/index.html>. Acesso em: 11 jun. 2020.

MAILGUN TECHNOLOGIES. **Mailgun Technologies**. Disponível em: <https://www.mailgun.com>. Acesso em: 11 jun. 2020.

MICROSOFT. **Visão geral da API de armazenamento de arquivos do OneDrive - Microsoft Graph**. Disponível em: <https://docs.microsoft.com/pt-br/graph/onedrive-concept-overview>. Acesso em: 11 jun. 2020.

ORACLE. **Differences between Java EE and Java SE - Your First Cup: An Introduction to the Java EE Platform**. Disponível em: <https://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html>. Acesso em: 6 abr. 2020.

ORACLE. **VisualVM**. Disponível em: <https://docs.oracle.com/javase/7/docs/technotes/guides/visualvm/index.html>. Acesso em: 11 jun. 2020a.

ORACLE. **JavaMail**. Disponível em: <https://javaee.github.io/javamail/>. Acesso em: 11 jun. 2020b.

ORACLE. **Java Platform, Standard Edition Java API Reference**. Disponível em: <https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Calendar.html>. Acesso em: 11 jun. 2020a.

ORACLE. **Java Platform, Standard Edition Java API Reference**. Disponível em: <https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Date.html>. Acesso em: 11 jun. 2020b.

PALMEIRA, T. V. V. **Classes em Java: trabalhando com Date, Calendar e SimpleDateFormat**. Disponível em: <https://www.devmedia.com.br/trabalhando-com-as-classes-date-calendar-e-simpledateformat-em-java/27401>. Acesso em: 11 jun. 2020.

RAMIRES, M. **Stack Overflow – Conheça o melhor site de perguntas e respostas de programação**. Disponível em: <http://www.techtudo.com.br/artigos/noticia/2011/02/stack-overflow-conheca-o-melhor-site-de-perguntas-e-respostas-de-programacao.html>. Acesso em: 5 jun. 2020.

RAMOS, A. **Não sabe? No Stack Overflow você encontra a resposta**. Disponível em: <https://medium.com/devzera/n%C3%A3o-sabe-no-stack-overflow-voc%C3%AA-encontra-a-resposta-9a2d6c066f52>. Acesso em: 5 jun. 2020.

SANTANA, E. F. Z. **Apache POI: Manipulando Documentos em Java**. Disponível em: <https://www.devmedia.com.br/apache-poi-manipulando-documentos-em-java/31778>. Acesso em: 10 jun. 2020.

SANTOS, R. R. DOS. **Programação de Computadores em Java**. 2ª ed. Rio de Janeiro: Novaterra Editora, 2014.

SCHILDT, H. **Java: The Complete Reference**. 7. ed. New York, USA: McGraw-Hill Professional Publishing, 2006.

SCHILDT, H. **Java para iniciantes**. 6. ed. Porto Alegre: Bookman, 2015.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Prentice Hall, 2011.

SOMMERVILLE, I.; SAWYER, P. **Requirements engineering: a good practice guide**. Chichester, Eng. ; New York: Wiley, 1999.

THE APACHE SOFTWARE FOUNDATION. **Commons Email - Página inicial**. Disponível em: <http://commons.apache.org/proper/commons-email/index.html>. Acesso em: 11 jun. 2020.

THE APACHE SOFTWARE FOUNDATION. **Apache POI - the Java API for Microsoft Documents**. Disponível em: <https://poi.apache.org/index.html>. Acesso em: 10 jun. 2020.

TIBICO SOFTWARE INC. **JasperReports® Library**. Text. Disponível em: <https://community.jaspersoft.com/project/jasperreports-library>. Acesso em: 10 jun. 2020.

TOEDTER.COM. **JCalendar**, 2016. Disponível em: <https://toedter.com/jcalendar/>. Acesso em: 11 jun. 2020

TWILIO INC. **Twilio's REST APIs**. Disponível em: <https://www.twilio.com/docs/usage/api?utm_source=docs&utm_medium=social&utm_campaign=guides_tags>. Acesso em: 11 jun. 2020.

UFSM. **Portal de Projetos - Visualizar projeto**. Disponível em: <https://portal.ufsm.br/projetos/publico/projetos/view.html?idProjeto=64128>. Acesso em: 12 jun. 2020.

W3SCHOOLS. **What Is “Write Once and Run Anywhere” Feature of Java?** Disponível em: </java-questions-answers/write-once-run-anywhere-wora/>. Acesso em: 21 maio. 2020.

WAZLAWICK, R. S. **Análise e projeto de sisteas de de informações orientados a objetos**. 2. ed. Rio de Janeiro (RJ): Elsevier Editora Ltda, 2011.

ZENVIA. **Zenvia - API SMS · Apiary**. Disponível em: <https://zenviasms.docs.apiary.io/#reference/servicos-da-api>. Acesso em: 11 jun. 2020.

ZUKOWSKI, J. **The definitive guide to Java Swing: creating Java-based graphical user interfaces using the J2SE 5 Swing component set**. 3. ed ed. Berkeley, Calif: Apress, 2005.

APÊNDICE A – DOCUMENTO DE REQUISITOS

Sistema de Gestão de Informações – DTG Noel Guarany – Documento de Requisitos

Requisitos Funcionais

1. Registrar novos sócios;
2. Registrar novos centros de ensino;
3. Registrar novos cursos;
4. Registrar novos departamentos;
5. Registrar novas invernadas;
6. Registrar novas viagens;
7. Emitir relatórios referente a sócios;
8. Emitir relatórios referente a viagens;
9. Enviar e-mails ou SMS para sócios ou grupo de sócios;
10. Salvar arquivos em nuvem;
11. Gerar gráficos estatísticos referentes a sócios;

Requisitos Suplementares

1. Apenas pessoas com perfil cadastrado no sistema podem acessar;
2. Executar em diferentes sistemas operacionais;
3. Diferentes níveis de perfis de usuários;
4. Somente o perfil “Administrador” poderá criar novos perfis;

1. Registrar novos sócios

Descrição:

O usuário registra o novo sócio baseado na coleta dos dados referentes a esse sócio (ver modelo de ficha no Anexo A). Após isso, é possível adicionar os vínculos que esse sócio tem, seja com curso, com invernada ou departamento.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de Entrada:

Os dados pessoais do sócio a ser cadastrado.

Informações de saída:

Ficha digital do sócio cadastrado.

Restrições lógicas:

Somente poderá ser feito o cadastro se as validações dos dados estiverem corretas.

Os vínculos do novo sócio só poderão ser cadastrados após o sócio ser cadastrado e o vínculo (curso, departamento, invernada) já existir no sistema.

Restrições tecnológicas:

2. Registrar novos centros de ensino

Descrição:

O usuário registra novos centros de ensino, conforme for a necessidade (centros de ensino referentes a UFSM).

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

O usuário informa o nome, a sigla e o número do prédio referente ao centro de ensino.

Informações de saída:

Lista atualizada com todos os centros de ensino cadastrados.

Restrições lógicas:

Não há.

Restrições tecnológicas:

Não há.

3. Registrar novos cursos

Descrição:

O usuário registra novos cursos conforme necessidade (cursos referentes a UFSM).

Fontes:

Conselho Diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

O usuário informa o código do curso, o nome e o centro de ensino a qual o curso está vinculado.

Informações de saída:

Lista atualizada com todos os cursos cadastrados.

Restrições lógicas:

O centro de ensino deve ser cadastrado previamente.

Restrições tecnológicas:

Não há.

4. Registrar novos departamentos

Descrição:

O usuário registra novos departamentos conforme for a necessidade do DTG.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

O usuário informa o nome do departamento.

Informações de saída:

Lista atualizada de departamentos do DTG Noel Guarany.

Restrições lógicas:

Não há

Restrições tecnológicas:

Não há.

5. Registrar novas invernadas

Descrição:

O usuário registra novas invernadas conforme elas forem criadas no DTG Noel Guarany.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

O usuário informa o nome da invernada e o(s) coordenador(es)

Informações de saída:

Lista atualizada de todas as invernadas cadastradas no sistema.

Restrições lógicas:

Não há.

Restrições tecnológicas:

Não há.

6. Registrar novas viagens

Descrição:

O usuário cria uma lista de viagem e registra essa viagem no sistema.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

As informações são a data da viagem, a cidade de origem, a cidade de destino, a pessoa responsável e a lista de passageiros.

Informações de saída:

Lista com todas as viagens cadastradas no sistema.

Restrições lógicas:

Somente sócios cadastrados poderão estar na viagem.

Restrições tecnológicas:

A seleção dos sócios é feita por uma lista de sócios organizada em ordem alfabética.

7. Emitir relatórios referentes a sócios

Descrição:

Emitir diferentes relatórios referentes aos dados dos sócios.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

Tipo de relatório (sócios ativos, sócios por invernada...).

Informações de saída:

Relatório em formato PDF.

Restrições lógicas:

Os sócios deverão estar cadastrados previamente no sistema.

Restrições tecnológicas:

8. Emitir relatórios referentes a viagens

Descrição:

Possibilidade de imprimir ou reimprimir a lista de uma viagem cadastrada no sistema.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (qualquer perfil).

Informações de entrada:

O usuário seleciona qual a viagem quer emitir o relatório.

Informações de saída:

A lista dos passageiros e seus dados, bem como as informações da viagem.

Restrições lógicas:

Não há.

Restrições tecnológicas:

A seleção da viagem é feita em uma lista gerada pelo sistema, a qual deve estar em ordem decrescente de data (viagem mais atual até a viagem mais antiga).

9. Enviar e-mails ou SMS para sócios ou grupo de sócios

Descrição:

Envio de e-mails ou SMS para um único sócio ou para um grupo pré-determinado de sócios. Possibilidade de anexo ou não.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

E-mail, telefone celular ou lista de sócios, mensagem e anexo (se necessário).

Informações de saída:

Confirmação do envio.

Restrições lógicas:

Somente um anexo por e-mail.

Restrições tecnológicas:

Não há.

10. Salvar arquivos em nuvem

Descrição:

Possibilidade de armazenamento em nuvem de arquivos importantes como atas e termos de ciência. Download e upload de arquivos.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (perfil de administrador).

Informações de entrada:

Arquivo a ser enviado ou baixado para/da nuvem.

Informações de saída:

Confirmação de upload/download.

Restrições lógicas:

Necessidade de informar o tipo de arquivo (ficha de sócio, ata, documento).

Restrições tecnológicas:

Download/upload de um arquivo por vez.

11. Gerar gráficos estatísticos referentes a sócios

Descrição:

Gerar gráficos referentes aos dados dos associados, bem como gráficos relacionados a invernadas e departamentos. Gráficos podem ser de vários tipos: barras, pizza, linha, etc.

Fontes:

Conselho diretor DTG Noel Guarany.

Usuário:

Usuário previamente cadastrado no sistema (qualquer perfil).

Informações de entrada:

Qual o tipo de gráfico e quais os dados de referência.

Informações de saída:

Gráfico relacionado a seleção.

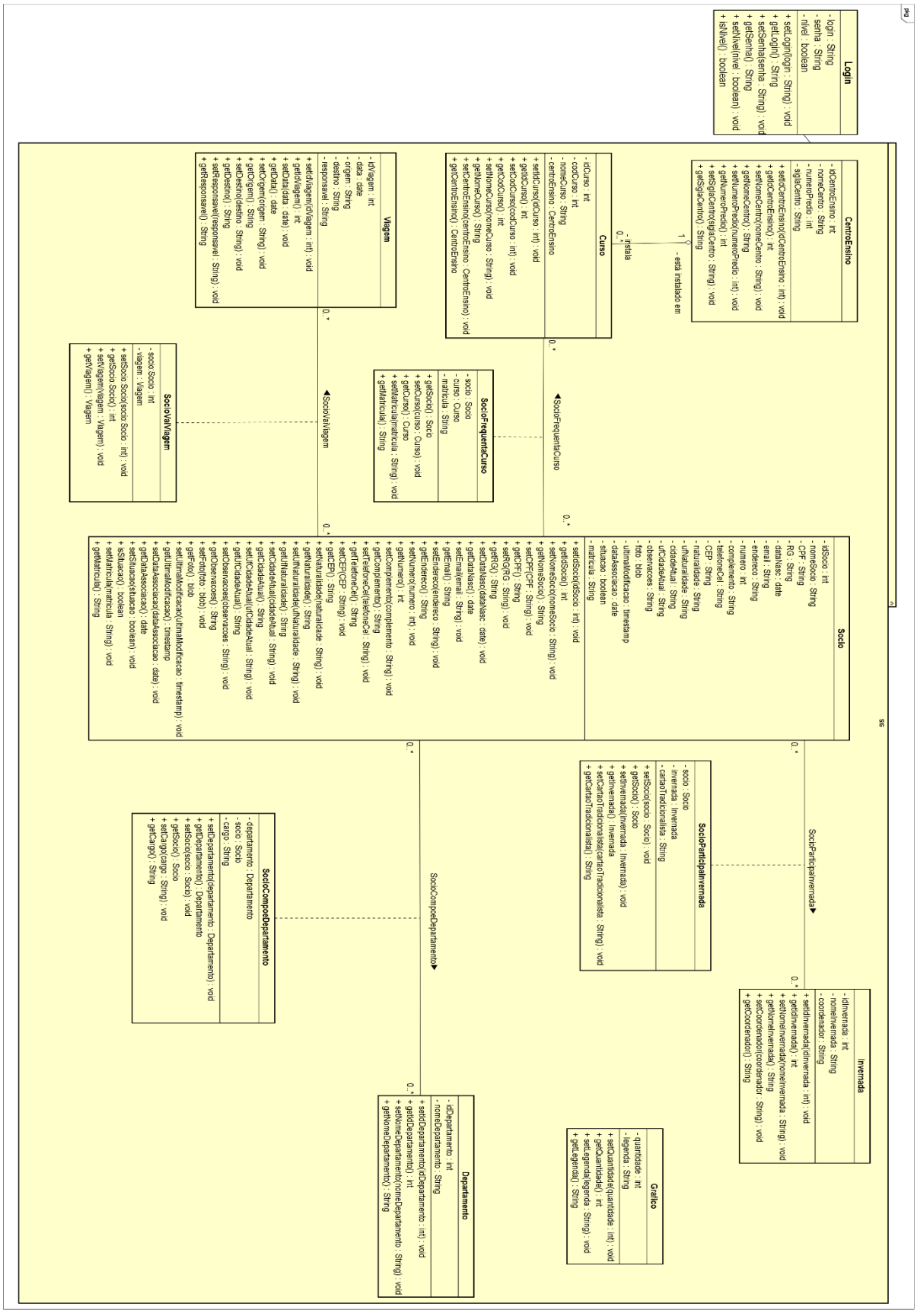
Restrições lógicas:

Os gráficos serão pré-definidos dentro do sistema.

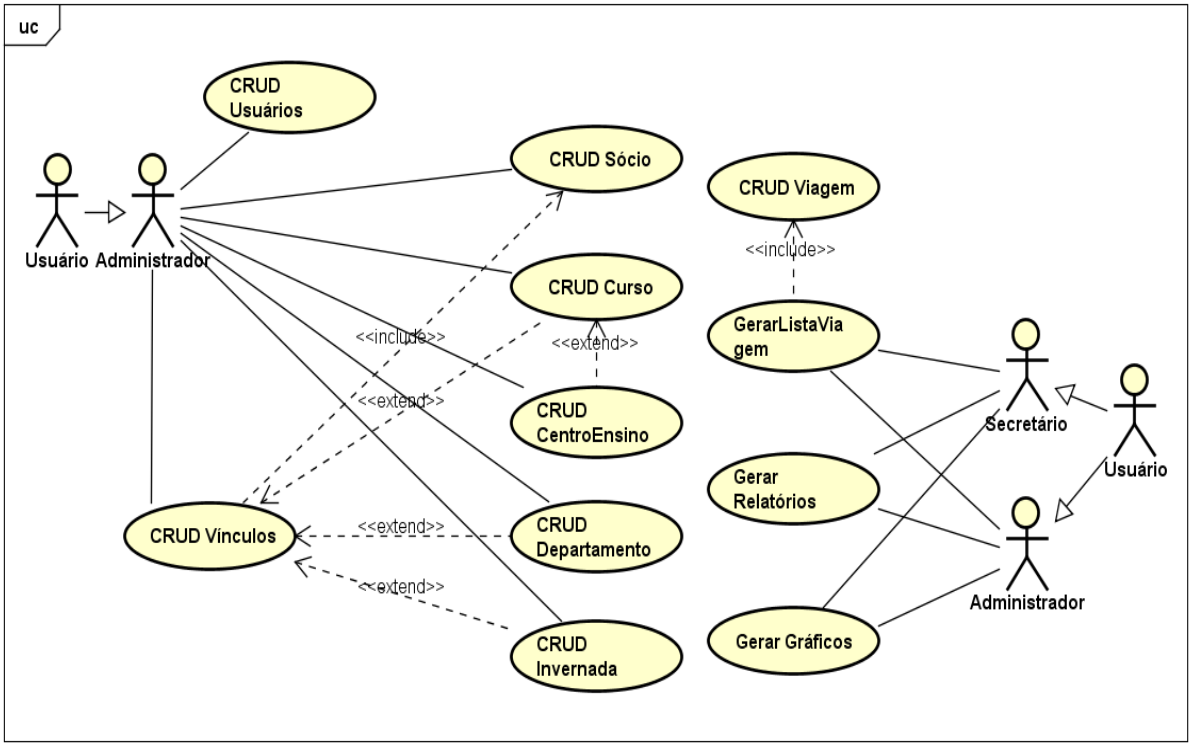
Restrições tecnológicas:

Somente um gráfico por vez.

APÊNDICE B – DIAGRAMA DE CLASSES COMPLETO



APÊNDICE C – DIAGRAMA DE CASOS DE USO COMPLETO



ANEXO A – FICHA DE SÓCIO

UNIVERSIDADE FEDERAL DE SANTA MARIA
PRÓ-REITORIA DE EXTENSÃO
DEPARTAMENTO DE TRADIÇÕES GAÚCHAS
NOEL GUARANY

FICHA DE CADASTRO DE ASSOCIADO

Foto

Nome: _____

Data de Nascimento: _____

Endereço: _____

Telefone: _____

E-mail: _____

Curso/Semestre: _____

Interesses/Habilidades Artísticas: _____

_____

_____

_____

Santa Maria, ____ de _____ de _____.

_____

Assinatura