CENTRO DE EDUAÇÃO TECNOLÓGIA PAULA SOUZA

FACULDADE DE TECNOLOGIA DE SÃO CAETANO DO SUL ANTONIO RUSSO

Cecília dos Santos Silva

Igor Veras da Silva

Victor Gomes Gazotti

SISTEMA WEB PARA BIBLIOTECA VIRTUAL DE TCC

Trabalho de Graduação apresentado por Cecília Silva, Igor Veras e Victor Gazotti, como pré-requisito para conclusão de Curso Superior de Tecnólogo em Análise e desenvolvimento de sistemas, da Faculdade de Tecnologia São Caetano do Sul Antonio Russo, elaborado sob a orientação do Prof. MSc. Márcio André Ferreira Pereira.

São Caetano do Sul

2016

AGRADECIMENTOS

Agradecemos a instituição de ensino FATEC São Caetano do Sul Antonio Russo, pelo suporte oferecido ao longo desses três anos; a todos os professores pela dedicação em contribuir para a nossa formação, em especial ao nosso orientador, Prof. MSc. Márcio André Ferreira Pereira, a todos os colegas de turma pelo companheirismo durante todo o curso; aos nossos amigos e familiares pelo incentivo na pesquisa e desenvolvimento deste trabalho acadêmico.

RESUMO

O Trabalho apresenta a proposta de desenvolvimento e implantação do Sistema Acadêmico Virtual TCC direcionado aos usuários da FATEC, tendo como objetivos a integração dos recursos informacionais de pesquisa para o aluno, o acesso aos materiais em formato digital, a otimização dos recursos acadêmicos na rede disponibilizando um serviço mais amplo de acesso à informação a comunidade acadêmica e demais interessados desse grande meio de pesquisa virtual. Para o desenvolvimento desse projeto foi utilizado o *framework Laravel* junto com a linguagem de programação PHP, na qual é flexível e muito utilizada atualmente. Foi aplicado como *front-end* o *framework* Bootstrap. Esse projeto foi criado com o objetivo de facilitar o processo de busca de uma pesquisa para alunos que estão elaborando um TCC.

Palavras-chave: biblioteca virtual, sistema acadêmico, PHP.

ABSTRACT

This monograph presents the proposed development and implementation of TCC's Academic Virtual System directed to FATEC's users, aiming the integration of information resources to search for students , access to material in digital for , the optimization of academic resources in providing network a broader service access to information for the academic community and other interested parties this great medium virtual. For the development of this project we used the Laravel framework along with the PHP programming language, which is flexible and currently used . It was used as front-end Bootstrap framework. This project was created in order to facilitate the search process of a search for students who are preparing a TCC.

Keywords: *virtual library, academic system, PHP.*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVEATURAS E SIGLAS

API- *Application Programming Interface*

AIX-*Advanced Interactive executive*

ASP- *Active Server Pages*

BBX-*BroadBand explorer*

C# - Linguagem de Programação Orientada à Objetos

C/C++- linguagem de programação compilada multi-paradigma

CGI- *Common Gateway Interface*

CSS - *Cascading Style Sheets*

CSS3- *Cascading Style Sheets 3*

CVS- *Concurrent verson system*

CGF- *Generic Connection Framework*

DBA-*Database administrator*

DBM- *decibel miliwatt*

DSO- *Dynamic Shared Objects*

FATEC – Faculdade de Tecnologia

FTP- *File Transfer Protocol*

HTML – *Hyper Text Markup Language*

HTML5 – *Hyper Text Markup Language* 5

HTTP- *Hypertext Transfer Protoco*

IOC – Inversão de controle

IMAP-*Internet Message Access Protocol*

IRIX- Sistema operacional baseado no Unix

JS – *Java Script*

JQuery – Biblioteca *Java Script*

LDAP-*Lightweight Directory Access Protocol*

MD5- *Message-Digest algorithm 5*

MVC –Model View Controller

MVVM - Model View - View Model

NNTP- *Network News Transfer Protocol*

OMT- *Object Modeling Technique*

OMG- *Object management group*

OOSE- *Object oriented software engineering*

ORM- *Object-relational mapping*

OS X- *Operating systems X*

PERL- linguagem de programação multiplataforma usada em aplicações de CGI

POO- Programação orientada a objetos

POP3- *Post Office Protocol*

PDF- *Portable Document Format*

PHP - *Personal Home Page*

PHP4 - *Personal Home Page 4*

PNG- *Portable Network Graphics*

RCS - *Revision Control System*

RPC- *Remote Procedure Call*

SCCS- *Source Code Control System*

SGBD- Sistema de gerenciamento de Banco de Dados

SOAP- *Simple Object Access Protocol*

SNMP- *Simple Network Management Protocol*

SPA – *Single Page Application*

SQL- *Structured Query Language*

SSL- *Secure Socket Layer*

SVG- *Scalable Vector Graphics*

TCC – Trabalho de Conclusão de Curso.

UML- *Unified Modeling Language*

URL -*Uniform Resource Locator*

W3C- *World Wide Web Consortium*

XMK- *Extensible Markup Language*

XML- *Extensible Markup Language*

Sumário

Introdução

O avanço da tecnologia da informação possibilitou a implantação de processos de controle e manipulação de dados viabilizando assim o controle de um volume cada vez maior dessas informações. Com o propósito de se tornar factível, foram criados mecanismos que possibilitam que esses dados sejam manipulados por sistemas integrados, os quais são chamados de *softwares.*

Com a utilização constante da internet como fonte de informação e comunicação, se fez necessário o desenvolvimento de páginas com conteúdos mais dinâmicos. A linguagem de script PHP (*Personal Home Pages*), escolhida para o desenvolvimento desse projeto é voltada para essa conjuntura, sendo hoje uma das mais utilizadas no mercado de desenvolvimento *web*.

Nesse contexto ainda, para cada projeto *web* desenvolvido é requisitado, cada vez mais, há a preocupação com a segurança da informação dos sistemas onde as informações são trafegadas, reforçando assim a escolha do PHP como linguagem base para esse projeto, pois o mesmo possui algumas soluções prontas que garantem grande parte dos princípios de segurança da informação, ferramentas estas que serão apresentadas neste projeto.

Este trabalho propõe o desenvolvimento de um Sistema Acadêmico Virtual de TCC, para a FATEC São Caetano do Sul Antonio Russo, que disponibilizara os trabalhos de conclusão de curso, para alunos, docentes e colaboradores. Sendo um projeto desenvolvido para facilitar a consulta, inclusão e acesso aos Trabalhos de Conclusão do Curso (TCC), através de uma plataforma digital.

Constatou-se a deficiência desse tipo de sistemas nas instituições educacionais do grupo FATEC, assim o objetivo desse trabalho é suprir essa deficiência fornecendo uma melhor integração dos alunos com o sistema, divulgar os trabalhos já desenvolvidos, e servir como fonte de pesquisa para projetos futuros.

O presente trabalho está dividido em três capítulos, apresentando a seguinte estrutura:

O capítulo um, apresenta todo o contexto e parâmetros dos softwares e ferramentas utilizadas para o desenvolvimento desse trabalho, passando pela linguagem de programação escolhida, pelos frameworks utilizados, pela estruturação do banco de dados e definindo biblioteca virtual.

O capítulo dois é dedicado á apresentação da estruturação do sistema, exibindo os diagramas e os requisitos necessários para finalização do trabalho.

No capítulo três, é abordada a contextualização do sistema através das imagens obtidas da tela dos mesmos.

CAPÍTULO I

Conceitos e ferramentas

Neste capitulo se desenvolverá os conceitos e ferramentas utilizados para o desenvolvimento desse projeto.

1.1 HTML

HTML é a sigla de *HyperText Markup Language*, (Linguagem de Marcação de Hipertexto). Consiste em uma linguagem utilizada para produção de páginas na web, que permite a criação de documentos que podem ser lidos em praticamente qualquer tipo de computador e transmitidos pela internet. (DEBONI, 2003)

O HTML é a linguagem base da internet, criada por Tim Berners-Lee na década de 1990. As especificações da linguagem são controladas pela W3C (World Wide Web Consortium), ficando conhecido quando começou a ser utilizada para formar a rede pública daquela época, o que se tornaria mais tarde a internet que conhecemos hoje. (DEBONI, 2003)

Segundo Deboni (2003) Não é possível programar em HTML, pois a linguagem não possui recursos através dos quais se possam construir procedimentos. Mas como HTML é texto simples pode ser editada em qualquer editor de texto. Toda a formatação de um arquivo HTML é feita exclusivamente através dos descritores (*Tag*). Ao ler um documento HTML, um *browser* tenta interpretar todas as sequências de caracteres que ficam entre os símbolos "<" e ">". O *browser* entende que qualquer coisa que estiver entre esses caracteres é um descritor HTML (*tag*) e não deve ser mostrado na tela. Se o descritor for desconhecido ele simplesmente não mostra o conteúdo na tela, mas se realmente for um elemento HTML (definido na especificação suportada pelo navegador), ele usará as informações contidas entre os símbolos para estruturar a página, formatando-a de acordo com alguma regra de estilo previamente definida.

1.2 PHP

No ano de 1994, Rasmus Lerdorf, inicia a utilização do PHP em sua pagina pessoal na internet. Já em 1995 o PHP passou a ser utilizado por diversos usuários, e assim foi reescrito com novos recursos, sendo nomeado para "*Personal Home Page Tools",* que era composto por um sistema simples que interpretava macros e utilitários que rodavam "por trás" das *homes- pages.* No mesmo ano o interpretador foi reescrito, e foi chamado de PHP/FI, o FI surgiu de um pacote escrito por Rasmus que interpretava dados de formulários HTML (*Form Interpreter*). Combinando os scripts do pacote *Personal Home Page Tools* com o FI, adicionando novos recursos como o suporte Mysql, entre outros. (DALL'OGLIO, 2007).

Estima-se que em 1996 o PHP/FI estava sendo utilizado por cerca de 15.000 sites pelo mundo, já em meados de 1997 esse número subiu para mais de 50.000. Assim ele deixou de ser um projeto de Rasmus, para se tornar de uma equipe mais organizada de desenvolvimento. Com uma equipe de desenvolvimento criou-se um novo interpretador que foi a base para a versão 3. (MILANI, 2010)

Nos anos 2000 há o lançamento do PHP4, trazendo muitas novidades aos programadores de PHP trouxe mudanças como: (MILANI, 2010)

- Suporte a sessão
- Referente à sintaxe;
- Novos recursos de programas;
- Otimizador Zend, capaz de executar *scrips* mais rapidamente.

PHP é a abreviação de *Hypertext PreProcessor,* que é uma linguagem de programação, em código aberto, interpretada, muito utilizado para a criação de conteúdo dinâmico na *Web*, possibilitando a manipulação de páginas HTML, e a interação com o usuário através de formulários, parâmetros da URL e *links*. (MILANI, 2010)

O PHP é uma linguagem de programação de domínio especifico, delimitado por *tags* inicias e finais, que permite ao desenvolvedor pular para dentro e fora do "modo PHP".

Algumas características do PHP segundo Milani (2010) são: Velocidade e robustez; Estruturado e orientação a objetos; Portabilidade e independência de plataforma; Sintaxe similar à linguagem C/C++ e PERL; Suporte a diversas bases de dados como: *Oracle, Sybase,*

PostgreSQL, *Interbase*, *MySQL*, *SQLite*, *MSSQL*, *Firebird*; Oferece suporte a diversos protocolos: IMAP; SNMP; NNTP, POP3, HTTP, LDAP, XML-RPC, SOAP; Disponível para múltiplos sistemas operacionais como: Windows, Linux, FreeBSD, Mac OS, OS/2, AS/400, Novell Netware, RISC OS, AIX, IRIX, Solaris; Compressão de bzip2; Conversão de calendário; ClibPDF; Crack; Ctype; CURL; Pagamento *Cybercash*; DBM; Abstração de banco de dados DBA, dBase, dbx, e outras.

A diferença do PHP com relação a linguagens semelhantes como o Javascript é que o código PHP é executado no servidor, sendo enviado para o cliente apenas HTML puro. Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente. Isso pode ser útil quando o programa está lidando com senhas ou qualquer tipo de informação confidencial. O que diferencia PHP de um script CGI escrito em linguagem C ou Perl é que o código PHP fica embutido no próprio HTML, enquanto no outro caso é necessário que o *script* CGI gere todo o código HTML, ou leia de um outro arquivo. (MILANI, 2010)

1.3 Programação orientada a objetos.

A programação Orientada a objetos (POO) é um padrão de desenvolvimento que é seguido por muitas linguagens, pois o desenvolvimento de software é extremamente amplo. Assim as diversas linguagens de programação seguem diferentes paradigmas. Um desses paradigmas é a orientação a objetos, que vem se evoluindo em questões de segurança e reaproveitamento de código, nas quais atualmente são muito importantes para o desenvolvimento de aplicações. (DEVMEDIA, 2016)

A POO se baseia em quatro pilares principais:

1. Abstração

A abstração é o ponto mais importante na orientação de objetos, uma vez que como se trata de um paradigma, a abstração lida com a representação de um objeto real e como ele será representado dentro do sistema. A abstração passa por 3 etapas. (DEVMEDIA, 2016)

A primeira etapa é fazer a identificação do objeto que será criado, criando uma identidade única dentro do sistema para que não haja conflito. A segunda etapa consiste em caracterizar o objeto, isto é definir elementos que serão chamados de propriedades. Na última etapa

definem-se as ações que o objeto irá executar, chamando as mesmas de métodos, que podem ser diversos. (DEVMEDIA, 2016)

2. Encapsulamento

O encapsulamento é um elemento que adiciona segurança em uma programação orientada a objetos, por meio da ocultação das propriedades, um exemplo de encapsulamento de uma rede telefônica é descrito na Tabela 1 (DEVMEDIA, 2016).

Tabela 1: Exemplo de encapsulamento

Telefone	
Interface publica	Botões utilizados para interagir com o objeto.
Implementação	As operações internas, o propósito do objeto.

3. Herança

Herança é um mecanismo que permite que características comuns a diversas classes sejam fatoradas em uma classe base, ou superclasse. A herança possibilita o reuso do código, otimizando a produção em tempo de linhas de código. Um exemplo dessa herança pode ser observada na Figura 1**Erro! Fonte de referência não encontrada.**.(DEVMEDIA, 2016).

Figura 1: Exemplo de herança entre objetos

Fonte: (DEVMEDIA, 2016)

Verifica-se na figura 1 o objeto abaixo na hierarquia herdará características de todos os objetos acima dele. A herança a partir das características do objeto mais acima é considerada herança direta, enquanto as demais são consideradas heranças indiretas. (DEVMEDIA, 2016)

4. Polimorfismo

Polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse. O polimorfismo está ligado à herança do objeto, alterando o funcionamento interno de um método herdado de um objeto. (DEVMEDIA, 2016)

As principais vantagens da programação orientada a objetos são: (DEVMEDIA, 2016)

- A reutilização de código: a orientação a objetos permite que haja uma reutilização do código criado, diminuindo o tempo de desenvolvimento, bem como o número de linhas de código.

- A leitura e manutenção de código: como a representação do sistema se aproxima muito do que se vê na vida real, tornando o entendimento do sistema como um todo e de cada parte individualmente mais simples.
- A criação de bibliotecas: Outro ponto que é muito mais simples com a orientação a objetos. Na POO as bibliotecas trazem representações de classes, tornando-as mais claras para a reutilização.

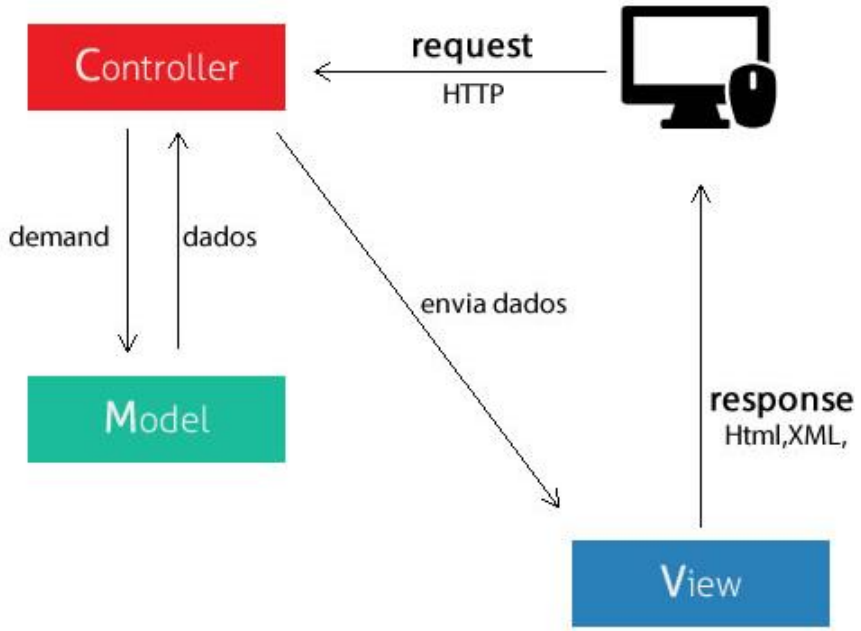
1.4 MVC

O MVC (*Model view controller)* é um padrão de arquitetura de software, que separa a aplicação em três camadas: (TABLELESS, 2016).

1. *VIEW:* camada de interação do usuário, responsável pela leitura, validação e escrita de dados.
2. *MODEL:* camada de manipulação dos dados, responsável pela exibição dos dados por meio de um HTML ou XML.
3. *CONTROLLER:* camada de controle, responsável por receber todas as requisições do usuário, usando métodos *actions* que são responsáveis por uma página, controlando qual *model* usar e qual *view* será mostrado ao usuário.

A Figura 2 representa um ciclo de requisição padrão sobre sistemas *WEB* que implementam o padrão MVC. O ciclo se inicia no cliente que busca por alguma ação específica no sistema. Essa ação é enviada para ser tratada em sua devida *Controller* e esta *Controller* faz as validações sobre os dados requisitados. Encerrando esse processo a *Controller* invoca a camada *Model* que deve fazer as devidas operações ao banco de dados (busca, escrita, atualização e exclusão) retornando o resultado novamente para a *Controller* que então enviará o resultado para ser apresentado na camada *View*.

Figura 2: Visualização gráfica do padrão MVC.



Fonte: (RAMOS, 2016)

1.5 *ORM*

O mapeamento objeto relacional (*ORM)* é uma ferramenta que tem como objetivo suprir as disparidades entre o paradigma orientado a objetos e o modelo entidade-relacional, assim promovendo uma ponte (mapeamento) entre o modelo racional e o modelo orientado a objeto. (PRINCIWEB, 2016).

O *ORM* gerencia detalhes de mapeamento de um conjunto de objetos para um banco de dados, além de possuir ferramentas que realizam a interação entre a aplicação e o banco de dados. (PRINCIWEB, 2016).

1.5.1 *Eloquent*

O *Eloquent*, é um *ORM* incluído no software *Laravel*, que fornece uma simples implementação *Active Record* para trabalhar com o banco de dados. Cada tabela do banco de

dados tem um *Model* e uma entidade correspondente que é usado para persistir os dados.(DEVMEDIA, 2016)

1.6 *FrameWork*

Framework é definido como uma abstração que une códigos entre vários projetos de software provendo uma funcionalidade genérica, podendo atingir uma funcionalidade especifica, por configuração, durante a programação de uma aplicação. (OFICINA DA NET, 2016).

O *framework* para Johnson (1997) descreve a arquitetura de um sistema orientado a objetos, além dos tipos e as interações entre os mesmos. Esta tecnologia pode ser compreendida como esqueleto (*template*) de uma aplicação podendo ser customizado pelo programador e empenhado a um conjunto de aplicações de um mesmo domínio.

A utilização de *frameworks* refere-se à busca pela a reutilização de subsistemas, elevando assim o grau de reutilização e contribuindo para uma melhor qualidade do software.

Os trabalhos de Fayad et. al (1999), apresenta os seguintes benefícios da utilização de *frameworks*:

1. Melhora a modularização: encapsulamento dos detalhes voláteis de implementação através de interfaces estáveis.
2. Aumenta a reutilização: definição de componentes genéricos que podem ser replicados para criar novos sistemas.
3. Extensibilidade: favorecida pelo uso de métodos *hooks* que permitem que as aplicações estendam interfaces estáveis.
4. Inversão de controle (IOC): o código do desenvolvedor é chamado pelo código do *framework*. Dessa forma, o *framework* controla a estrutura e o fluxo de execução dos programas.

Ainda segundo o autor Fayad et. al (1999), para a obtenção dos benefícios prometidos pelo *framework* é fundamental investir na qualidade do projeto do framework. Assim *framework* escolhido para esse projeto foi o *Lavarel* 5.2.

1.6.1 Lavarel 5.2

Laravel é um *framework* PHP para desenvolvimento *web* que utiliza a arquitetura MVC. Sua principal característica é ajudar a desenvolver aplicações seguras e de alto desempenho, com código limpo e simples. (DEVMEDIA, 2016)

Para a criação de interface gráfica, o *Laravel* utiliza uma *template Engine* chamada *Blade*, que traz um rol de ferramentas que ajudam a criar interfaces gráficas e funcionais de forma rápida e evitando a duplicação de código. (DEVMEDIA, 2016)

Para se comunicar com um Banco de Dados o *Laravel* utiliza uma implementação simples do *ActiveRecord* chamada de *Eloquent ORM*, que é uma ferramenta que traz várias funcionalidades para facilitar a inserção, atualização, busca e exclusão de registros. (DEVMEDIA, 2016)

Nesse projeto foi escolhido o *framework Lavarel* por causa de seu *composer* que facilita a utilização de componentes e também pelo *Artisan*, que permite a realização de migração do banco de dados, pelo comando *Codeingiter* que permite a realização de testes unitários, e o maior deles é para agilizar o processo de desenvolvimento, pois o framework já traz uma série de módulos pré-configurados para fazer as mais variadas e comuns, na sua versão mais atual e estável 5.2.

1.7 JavaScript

È uma linguagem de programação na maquina do usuário utilizada para controlar o HTML e o CSS, manipulado comportamentos na página, criado por Brendan Eich, primeiramente se chamando *LiveScript*, mas logo seu nome foi mudado para JavaScript. Foi desenvolvido para funcionar originalmente no *Netscape Navigator*, um navegador *web* com o propósito de oferecer aos desenvolvedores, formas de tornar determinados processos de páginas *web* mais dinâmicos. Posteriormente a Microsoft portou a linguagem para o seu navegador, consolidando-a. (CANALTECH, 2016)

O JavaScript tem como característica executar programas localmente. Assim o JavaScript fornece às páginas *web* a possibilidade de programação, transformação e processamento de dados enviados e recebidos, interagindo com a marcação e exibição de conteúdo da linguagem HTML e com a estilização desse conteúdo proporcionada pelo CSS nessas páginas. (CANALTECH, 2016).

1.8 *Media Query*

Segundo Zemel (2012), *Media Queries* nada mais é que a utilização de *Media Types* com uma ou mais expressões envolvendo características de uma media para definir formatações para diversos dispositivos.

As *Media Types* foram criadas pela W3C (*World Wide Web Consortium*), para o CSS (*Cascading Style Sheets*) com o objetivo de transformar um site feito em CSS mais acessível a diversos dispositivos. O CSS é uma linguagem de folhas de estilo utilizada para definir a apresentação de documentos escritos em uma linguagem de marcação, como HTML ou XML. O seu principal benefício é a separação entre o formato e o conteúdo de um documento. (W3C, 2016).

O conceito de *Media Types*, engloba uma página *web* é acessada ela seja renderizada conforme o estilo mais apropriado ao *device* que está sendo utilizado para realizar o acesso. (ZEMEL, 2012).

Existem 10 *Medias Types*, e são: (ZEMEL, 2012)

- *All*: Para todos os dispositivos
- *Braille*: Para dispositivos táteis
- *Embossed:* Para dispositivos que imprimem em braile
- *Hadheld:* Para dispositivos de mão, com telas pequenas e banda limitada.
- *Print*: Impressão em papel.
- *Projection:* Para apresentação como *PowerPoint.*
- *Screen:* Para monitores ou outros dispositivos com telas coloridas e com resolução adequada.
- *Speech:* Para sintetizadores de voz.
- *Tty*: Para dispositivos que utilizam uma grade fixa para exibição de caracteres.
- *Tv:* Para dispositivos como televisores.

Todos os navegadores modernos suportam o *Media Type*. Alguns celulares antigos suportam o *Media Types* do tipo *handheld* para estilos específicos para sites mobile. Os sistemas operacionais modernos como *iPhone* e *Android*, porém, ignoram o *Media Type handheld* pois são capazes de fazer a compilação de *sites* completos e não apenas as versões simples feitas

para os celulares antigos. Para a solução desse problema foi desenvolvido os *Media Queries.* (LOPES, 2016)

Uma *Media Query* consiste de um *Media Type* e zero ou mais expressões que apontam para condições específicas de um determinado tipo de mídia. Nada mais é que a expansão das *Medias Types* permitindo novas expressões, que permite controlar quando é necessário adicionar estilos específicos para visualização em determinados tipos de dispositivos. (LOPES, 2016)

Para o desenvolvimento desse trabalho foi escolhido o programa Bootstrap.

1.8.1 Bootstrap

Bootstrap é um *framework front-end* que fornece suporte para a criação de sites com tecnologia *mobile* (responsivo). Desenvolvido pela equipe do *Twitter*. Compatível com HTML5 e CSS3, o *framework* possibilita a criação de layouts responsivos e o uso de *grids*, permitindo que o conteúdo seja organizado em até 12 colunas e que se comporte de maneira diferente para cada resolução. (NASCIMENTO, 2016)

Algumas características segundo Costa (2016) são:

- Possui uma interface amigável e moderna;
- Atualmente possui uma grande diversidade de temas;
- Grande quantidade de *plugins* adaptados ou desenvolvidos para o *framework*;
- Integração com qualquer linguagem de programação;
- Sistema responsivo;
- Um dos *frameworks* mais utilizados no desenvolvimento de portais e sistemas do mundo.

As vantagens e desvantagens dessa ferramenta segundo Nascimento (2016) são:

Vantagens

- Possui documentação detalhada e de fácil entendimento;
- É otimizado para o desenvolvimento de *layouts* responsivos;

- Possui componentes suficientes para o desenvolvimento de qualquer site ou sistema *web* com interface simples;
- Facilita a criação e edição de *layouts* por manter padrões;
- Funciona em todos os navegadores atuais (Chrome, Safari, Firefox, IE, Opera).

Desvantagens

- Seu código terá de seguir os “padrões de desenvolvimento Bootstrap”;
- Tema padrão e comum do Bootstrap.

1.9 Banco de Dados

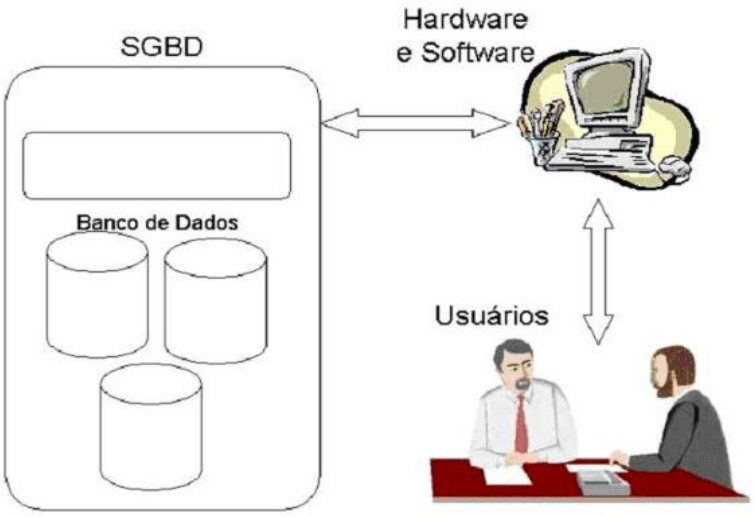
Rezende (2016) define banco de dados como uma coleção de dados inter-relacionados, representando informações sobre um domínio específico, ou seja, é a compilação de informações que se relacionam e tratam de um mesmo assunto.

Já um Sistema de Gerenciamento de Banco de Dados (SGBD) é um software que possui recursos capazes de manipular as informações do banco de dados e interagir com o usuário. (REZENDE, 2016).

Rezende (2016) define ainda que um Sistema de Banco de Dados é o conjunto de quatro componentes básicos: dados, *hardware*, *software* e usuários. Conforme se observa na

Figura 3 apresenta usuários utilizando um sistema que está rodando em um computador pessoal. Este sistema necessita fazer interações com o banco de dados (busca, escrita, atualização e exclusão) que são realizadas através de comandos específicos enviados ao sistema gerenciador de banco de dados (SGBD). O SGBD por sua vez é responsável por traduzir e realizar estas operações nos bancos de dados retornando o resultado ao sistema que por sua vez irá tratar esse resultado e apresentar ao usuário.

Figura 3: Elementos de um sistema de banco de dados



Fonte: (DEVMEDIA, 2016)

Os objetivos de um sistema de banco de dados são o de isolar o usuário dos detalhes internos do banco de dados (promover a abstração de dados) e promover a independência dos dados em relação às aplicações, ou seja, tornar independente da aplicação, a estratégia de acesso e a forma de armazenamento. (REZENDE, 2016). Para o desenvolvimento desse projeto foi escolhido o programa MySQL *workbench*.

1.10 MySQL Workbench

Galbiatti (2016), define o MySQL Workbench como um SGBD relacional de código aberto usado na maioria das aplicações gratuitas para gerir suas bases de dados. O serviço utiliza a linguagem SQL (*Structure Query Language*).

O sistema foi desenvolvido pela empresa sueca MySQL AB e publicado, originalmente, em maio de 1995. Após, a empresa foi comprada pela Sun Microsystems e, em janeiro de 2010, integrou a transação bilionária da compra da Sun pela Oracle Corporation. (GALBIATTI, 2016).

As vantagens da utilização desse software são: (LOPES, 2016)

- Gratuita para uso sem restrições;
- Disponível para Windows, Linux e OS X;
- Ótima documentação, bem organizada e com uma linguagem simples para os iniciantes;
- Pouco consumo de memória;
- Permite fazer engenharia reversa;
- Conexão direta com o banco de dados;
- Exportar em vários formatos do mercado: PNG, PDF e SVG;
- Baixa curva de aprendizado

1.11 Sistema de controle de versões

O Sistema de Controle de versão CVS (*Concurrent Version System*) foi originalmente desenvolvido por Dick Grune em 1984 como o primeiro CVS a suportar o gerenciamento de múltiplos arquivos e tornado público em 1986 e é dele que os sistemas de controle de versão atuais derivam. (DEVMEDIA, 2016)

De acordo com Bolinger e Bronson (1995), em 1972, um dos primeiros sistemas de controle de versões o *Source Code Control System* (SCCS), foi elaborado por Marc. J. Rochkind. Ainda que seja considerado obsoleto, ele foi o principal sistema de controle de versão até a

aparição do RCS (*Revision Control System*). O RCS foi desenvolvido em 1982 por Walter F. Tichy, utilizando técnicas mais avançadas.

Spain (2016), afirma que o CVS, foi desenvolvido com base no RCS, mas com a possibilidade de gerenciar projetos inteiros e não só um arquivo individualmente, como no RCS.

No CVS, os arquivos do projeto ficam armazenados em um repositório, onde o histórico das versões é salvos. Na Figura 4, a seguir podem ser observado ao funcionamento do CVS, onde há a troca de informações sobre as versões.

Figura 4: Esquema de funcionamento dos controles de versão

Fonte: (DEVMEDIA, 2016)

O programa de controle de versões escolhido para esse trabalho foi o GIT.

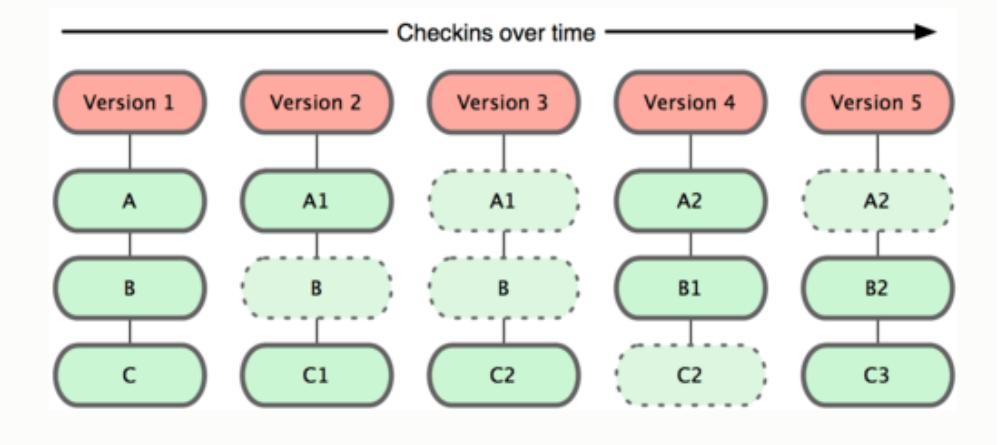
1.11.1 GIT

GIT é um sistema de controle de versão de arquivos. No qual se desenvolve projetos compartilhados, ou seja, uma gama de desenvolvedores podem contribuir simultaneamente, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas. (TABLELESS, 2016)

Outro fator importante do GIT é a possibilidade de criar, a qualquer momento, vários snapshots do seu projeto.

GIT não pensa ou armazena sua informação dessa forma. Ao invés disso, o Git considera que os dados são como um conjunto de s*napshots* (captura de algo em um determinado instante) de um minissistema de arquivos. Permitindo assim a consolidação do estado do projeto, através do armazenamento de uma referencia do *snapshots*. A Figura 5 apresenta o funcionamento do GIT em relação aos dados. (TABLELESS, 2016)

Figura 5: Armazenamento de dados como *snapshots* ao longo do tempo.



Fonte: (TABLELESS, 2016)

O Git considera que os arquivos sempre estejam em um dos três estados fundamentais que são: Consolidado (*committed*); Modificado (*modified*); Preparado (*staged*).

Assim quando os dados são classificados como consolidados estes estão armazenados na base local. Já quando classificados como modificados, os arquivos sofreram mudanças e não foram consolidados na base de dados. Já os arquivos classificados como preparados são arquivos modificados e marcados na versão corrente para que ele participe do snapshot do *commit* (consolidação). (TABLELESS, 2016)

1.12 Servidor APACHE

O servidor web escolhido para o desenvolvimento do projeto foi o servidor APACHE, criado por Rob McCool, em 1995. O servidor APACHE é responsável por possibilitar que páginas e todos os recursos possam ser acessados pelo internauta. Desde envio de e-mails, mensagens, compras online e diversas outras funções. (APACHE SOFTWARE FOUNDATION, 2016)

Apache é um servidor gratuito, sendo a Apache Software Foundation, responsável pelo projeto, e é compatível com o protocolo HTTP versão 1.13. Suas funcionalidades são mantidas através de uma estrutura de módulos, o que permite que os usuários escrevam seus próprios módulos por meio da API do software. Ele está disponível para Windows, Novell Netware, OS/2 e outros sistemas do padrão POSIX, como o Unix e o Linux, onde é amplamente utilizado. (CANALTECH, 2016)

As principais características segundo Alecrim (2016):

- Possui suporte a *scripts* CGI usando linguagens como *Perl*, PHP, *Shell Script*, ASP, etc;
- Suporte a autorização de acesso podendo ser especificadas restrições de acesso separadamente para cada endereço/arquivo/diretório acessado no servidor;
- Autenticação requerendo um nome de usuário e senha válidos para acesso a alguma página/sub-diretório/arquivo (suportando criptografia via Crypto e MD5);
- Negociação de conteúdo, permitindo a exibição da página *web* no idioma requisitado pelo Cliente Navegador;
- Suporte a tipos mime;
- Personalização de logs;
- Mensagens de erro;
- Suporte a virtual *hosting* (é possível servir duas ou mais páginas com endereços/ portas diferentes através do mesmo processo ou usar mais de um processo para controlar mais de um endereço);
- Suporte a *IP virtual hosting*;
- Suporte a *name virtual hosting*;
- Suporte a servidor *Proxy* FTP e HTTP, com limite de acesso, *caching* (todas flexivelmente configuráveis);

- Suporte a *proxy* e redirecionamentos baseados em URL's para endereços Internos;
- Suporte a criptografia via SSL, Certificados digitais;
- Módulos DSO (*Dynamic Shared Objects*) permitem adicionar/remover funcionalidades e recursos sem necessidade de recompilação do programa.

1.13 *Sublime Text* 3

Outro *software* utilizado para o desenvolvimento desse projeto é o *Sublime Text* 3, segundo Amaral (2016),o *Sublime Text* é um editor de texto rápido, simples, flexível e fácil de usar, usado para escrever o código do projeto.

As características do *Sublime Text* 3 são: (AMARAL, 2016)

- Goto *Anything*: Comando que faz com que a transição entre os arquivos pareça instantânea.
- Seleção Múltipla: A capacidade de selecionar trechos de texto de forma flexível é um dos principais recursos do *Sublime Text*. Permitindo fazer a manipulação do texto usando mais de um cursor e mais de uma região selecionada.
- *Plugins*: Oferece a capacidade de estender as funcionalidades do editor com um vasto ecossistema de *plugins* e pacotes de que podem ser acessado por meio de comandos do próprio editor.
- *Minimap*: facilita localizar um trecho do arquivo.
- Opções de *layout*: permite modificar o *layout* da janela de edição para exibir até quatro arquivos simultâneos, dispostos em linhas, colunas ou em grade.
- Personalização: Teclas de atalho, menus, *plugins*, auto complementos de texto, macros, temas visuais, esquemas de cores, realce de sintaxe, por meio da edição de arquivos de configuração em texto puro.
- *Command Palette*: permite pesquisar os próprios comandos do *Sublime Text*.
- *Snippets*: são trechos de código ou texto prontos para inserção em qualquer ponto de um arquivo por meio de um trigger acionado pela tecla Tab.
- Desempenho: a resposta a qualquer comando executado acontece instantaneamente, em transições suaves.
- Multiplataforma: com versões para Linux, OS X e Windows.

1.14 UML

Ribeiro (2016) conceitua a UML (*Unified Modeling Language)* como uma linguagem que define uma série de artefatos de modelar e documentar os sistemas orientados a objetos está sendo desenvolvido.

Objetivos da UML segundo Targettrust (2016) são: especificação, documentação, estruturação para sub-visualização e maior visualização lógica do desenvolvimento completo de um sistema de informação. A UML é um modo de padronizar as formas de modelagem.

A UML tem origem na compilação dos conceitos de *Booch*, OMT (*Rumbaugh*) e *OOSE* (*Jacobson*) fundindo-os numa única linguagem de modelagem comum e largamente utilizada. Os esforços para a criação da UML tiveram início em outubro de 1994, quando Rumbaugh se juntou a Booch na Rational. Com o objetivo de unificar os métodos *Booch* e OMT, decorrido um ano de trabalho, foi lançado, em outubro de 1995, o esboço da versão 0.8 do *Unified Process*. Nesta mesma época, Jacobson se associou à Rational e o escopo do projeto da UML foi expandido para incorporar o método OOSE. Nasceu então, em junho de 1996, a versão 0.9 da UML.(TARGETTRUST, 2016)

Finalmente em 1997, a UML foi aprovada como padrão pelo OMG (Object Management Group), um consórcio internacional de empresas que define e ratifica padrões na área de Orientação a Objetos. (TARGETTRUST, 2016)

A grande vantagem da utilização da UML é que ela engloba as atividades de análise, “design”, implementação e teste. (COMPUTAÇÃO UFCG, 2016)

O modelo subjacente à linguagem é composto de: (COMPUTAÇÃO UFCG, 2016)

- Itens;
- Relacionamentos;
- Diagramas.

Assim em itens pode-se verificar: (COMPUTAÇÃO UFCG, 2016)

- Estruturais
    - classes, interfaces, casos de uso, componentes ...
- Comportamentais
    - Interações, máquinas de estado.

- Grupos de elementos
    - Pacotes, frameworks e subsistemas.
- Anotacionais
    - Notas

Em relacionamentos encontra-se: (COMPUTAÇÃO UFCG, 2016)

- Dependência
- Associação
- Generalização
- Realização

Os relacionamentos são entre itens como classes, casos de uso e etc.

A UML possui nove tipos de diagramas que são usados para documentar e modelar diversos aspectos dos sistemas. São eles: (RIBEIRO, 2016)

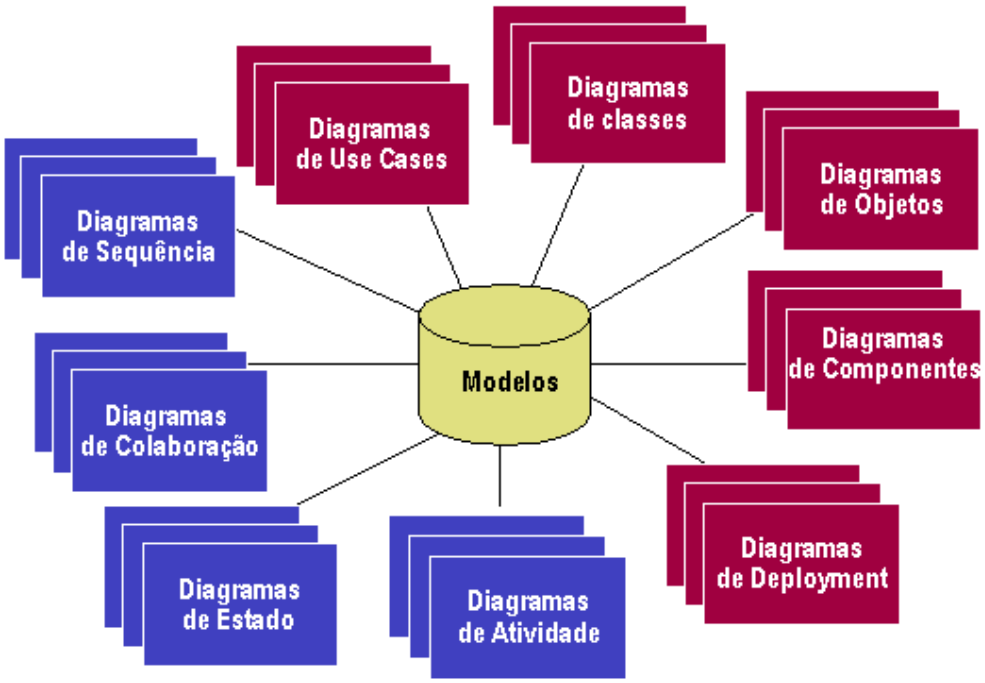
- Diagrama de classes: é a espinha dorsal da maioria dos métodos orientados a objeto, inclusive UML. Descreve a estrutura estática do sistema (entidades e relacionamentos);
- Diagrama de objetos: Descreve a estrutura estática de um sistema em um determinado momento, podem ser usados para testar a precisão dos diagramas de classe.
- Diagrama de caso de uso: Modela a funcionalidade do sistema através de atores e casos de uso. Os Casos de uso são serviços ou funções fornecidas pelo sistema aos seus usuários.
- Diagrama de sequência: Descreve as interações entre as classes através das trocas de mensagens ao logo do tempo.
- Diagrama de colaboraçõe: Representa as interações entre objetos em termos de mensagens em sequência.
- Diagramas de gráficos de estado: Descrevem o comportamento dinâmico do sistema em resposta a estímulos externos São especialmente úteis para modelar objetos reativos cujos estados são disparados por eventos específicos.
- Diagrama de atividade: Ilustra a natureza dinâmica de um sistema modelando o fluxo de controle de uma atividade para outra. Uma atividade representa uma operação em uma classe do sistema que resulta na mudança do estado do sistema. Tipicamente, são

usados para modelar fluxo de trabalho ou processos de negócio e funcionamento interno.

- Diagrama de componente: Descreve a organização dos componentes físicos de software.
- Diagrama de implantação: Descreve os recursos físicos em um sistema, incluindo nós, componentes e conexões.

Pode-se encontrar na Figura 6 a representação do modelo dos dois conjuntos de diagramas, sendo vermelho para estático ou estrutural e azul para dinâmico ou comportamental.

Figura 6: Modelo do sistema de diagramas



Fonte: (COMPUTAÇÃO UFCG, 2016)

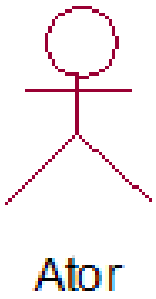
1.14.1 Diagrama de casos de uso

Segundo Computação UFCG (2016) o diagrama de Casos de Uso tem como objetivo auxiliar a comunicação entre os analistas e o cliente. E ainda descreve um cenário que mostra as funcionalidades do sistema do ponto de vista do usuário. Assim o cliente deve ver no diagrama de Casos de Uso as principais funcionalidades de seu sistema.

O diagrama de Caso de Uso é representado por: (COMPUTAÇÃO UFCG, 2016)

- Atores: Um ator é representado por um boneco e um rótulo com o nome do ator. Conforme figura 7. Um ator é um usuário do sistema, que pode ser um usuário humano ou um outro sistema computacional.

Figura 7: Exemplo de representação de Atores



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Casos de uso: Um caso de uso é representado por uma elipse e um rótulo com o nome do caso de uso. Conforme figura 8, um caso de uso define uma grande função do sistema. A implicação é que uma função pode ser estruturada em outras funções e, portanto, um caso de uso pode ser estruturado.

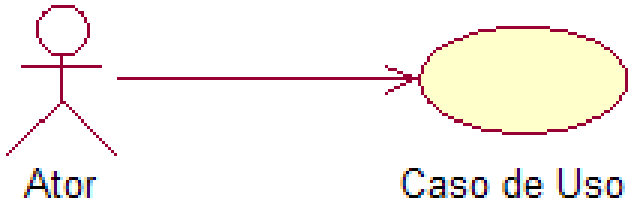
Figura 8: Exemplo de representação de casos de uso

Fonte: (COMPUTAÇÃO UFCG, 2016)

- Relacionamentos entre estes elementos: Os relacionamentos ajudam a descrever casos de uso e são divididos em:

- Associações entre atores e casos de uso: Que define uma funcionalidade do sistema do ponto de vista do usuário. É representado conforme a figura 9, onde um ator é associado a um caso de uso.

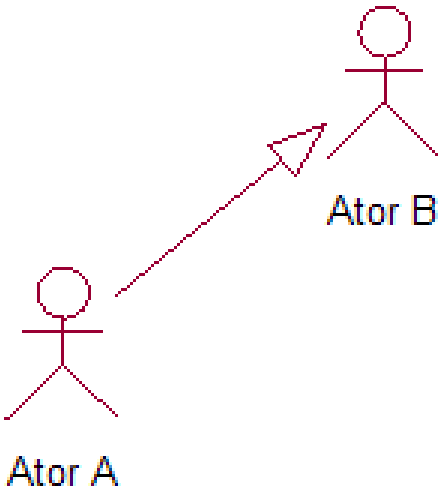
Figura 9: Exemplo de representação de relacionamento entre um ator e um caso de uso



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Generalizações entre os atores: Quando os casos de uso de B são também casos de uso de A, são representados conforme a figura 10.

Figura 10: Exemplo de representação de relacionamento entre atores



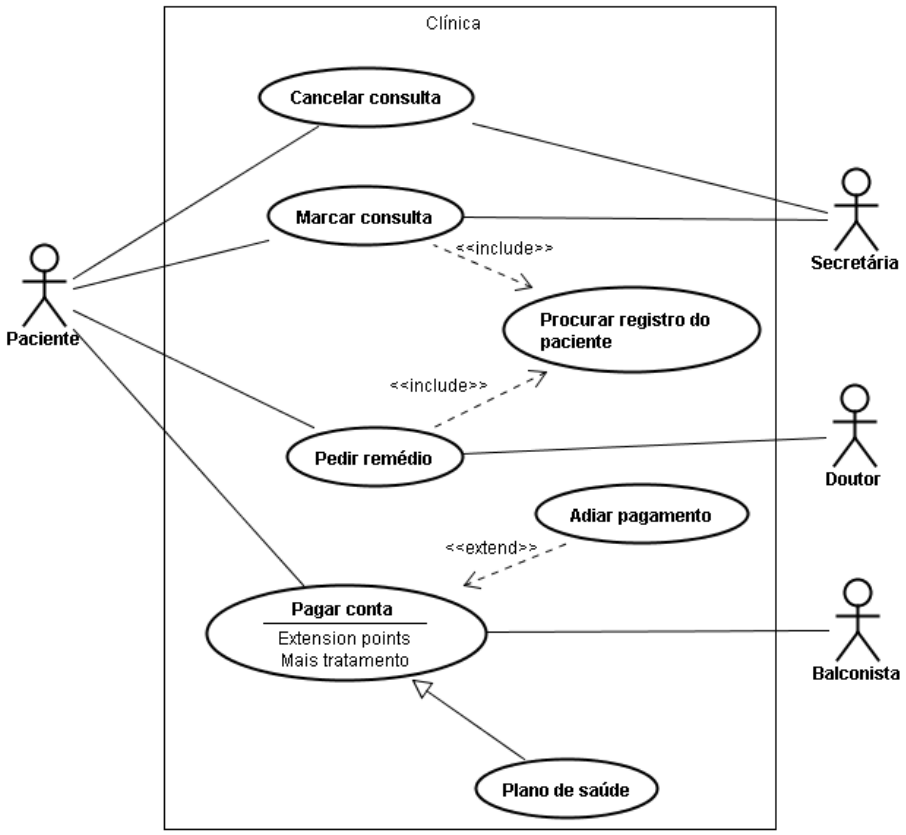
Fonte: (COMPUTAÇÃO UFCG, 2016)

- *Extends* entre os casos de uso: Um relacionamento *extend* em um caso de uso B para um caso de uso A indica que o caso de uso B pode ser acrescentado para descrever o comportamento de A, extensão é inserida em um ponto de extensão do caso de uso A.

- *Includes* entre os casos de uso. Um relacionamento *include* de um caso de uso A para um caso de uso B indica que B é essencial para o comportamento de A.

- Sistema
  - Limites do sistema: representado por um retângulo envolvendo os casos de uso que compõem o sistema. Conforme a Figura 11
  - Nome do sistema: Localizado dentro do retângulo. Conforme a Figura 11

Figura 11: Exemplo de representação do limite do sistema e do nome do sistema



Fonte: (COMPUTAÇÃO UFCG, 2016)

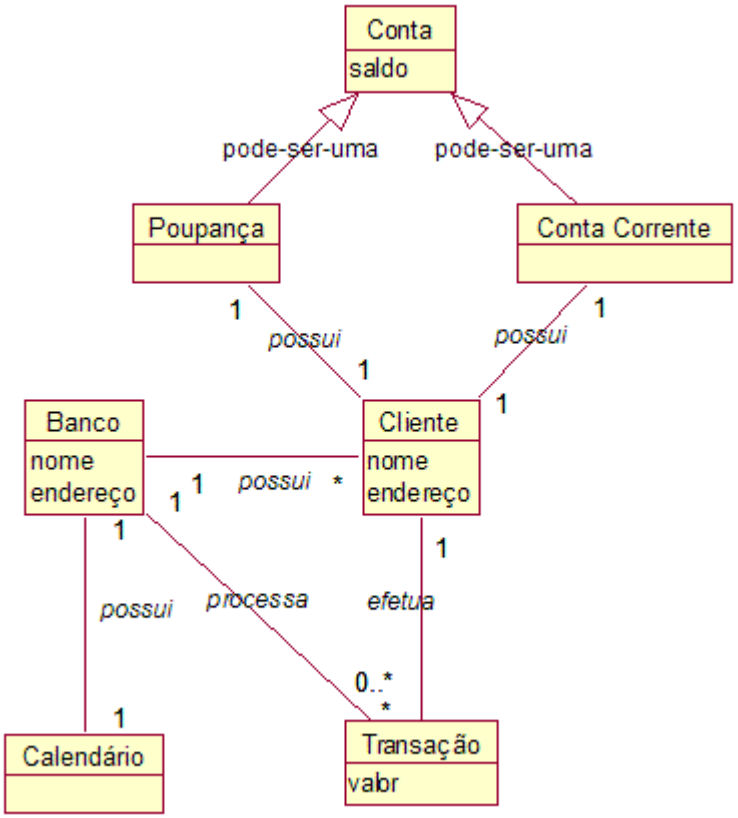
1.14.2 Diagrama de classes

O objetivo do diagrama de classes é descrever os vários tipos de objetos no sistema e o relacionamento entre eles. Assim o diagrama de classes pode oferecer três perspectivas, cada uma para um tipo de usuário diferente. São elas: (COMPUTAÇÃO UFCG, 2016)

- Conceitos ou Entidades

Que representa os conceitos do domínio em estudo, e tem uma perspectiva destinada ao cliente. Conforme o exemplo da figura 12.

Figura 12: Exemplo da representação dos conceitos ou entidades.

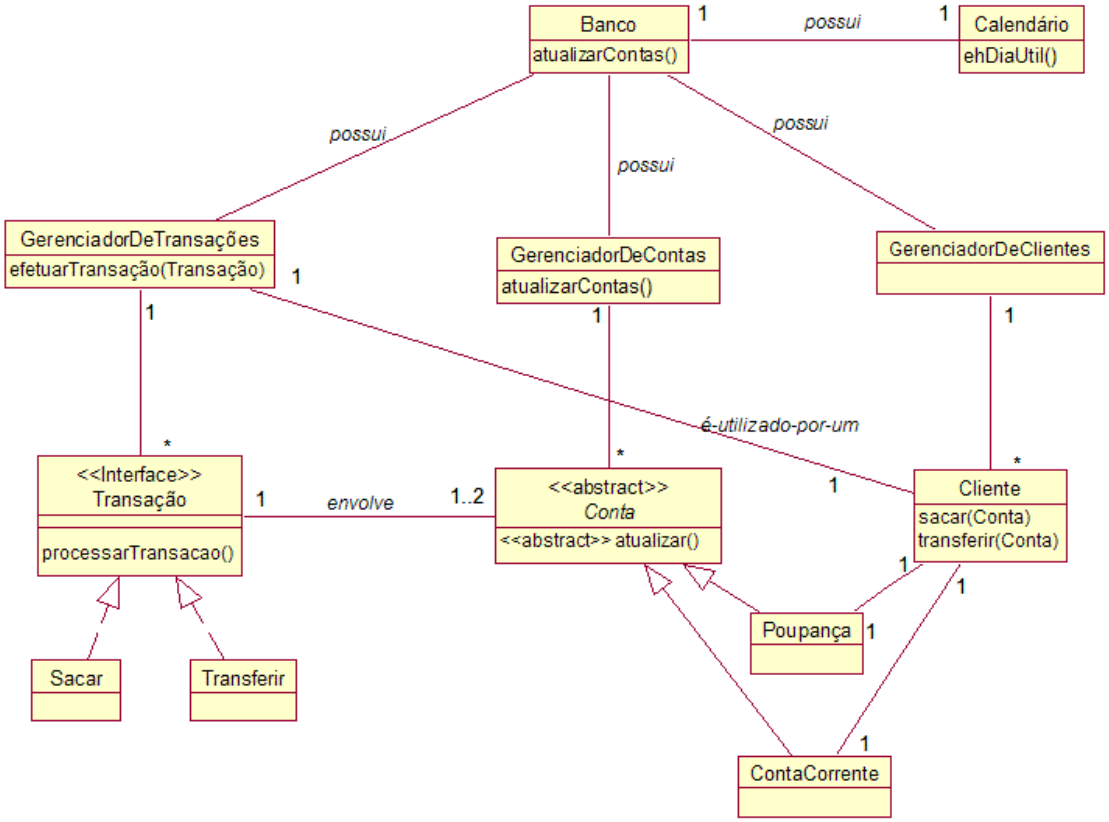


Fonte: (COMPUTAÇÃO UFCG, 2016)

- Classes

Tem foco nas principais interfaces da arquitetura, nos principais métodos, e não como eles irão ser implementados, e possui uma perspectiva destinada as pessoas que não precisam saber detalhes de desenvolvimento, tais como gerentes de projeto. Conforme o exemplo da figura 13.

Figura 13: Exemplo da representação das perspectivas de classes

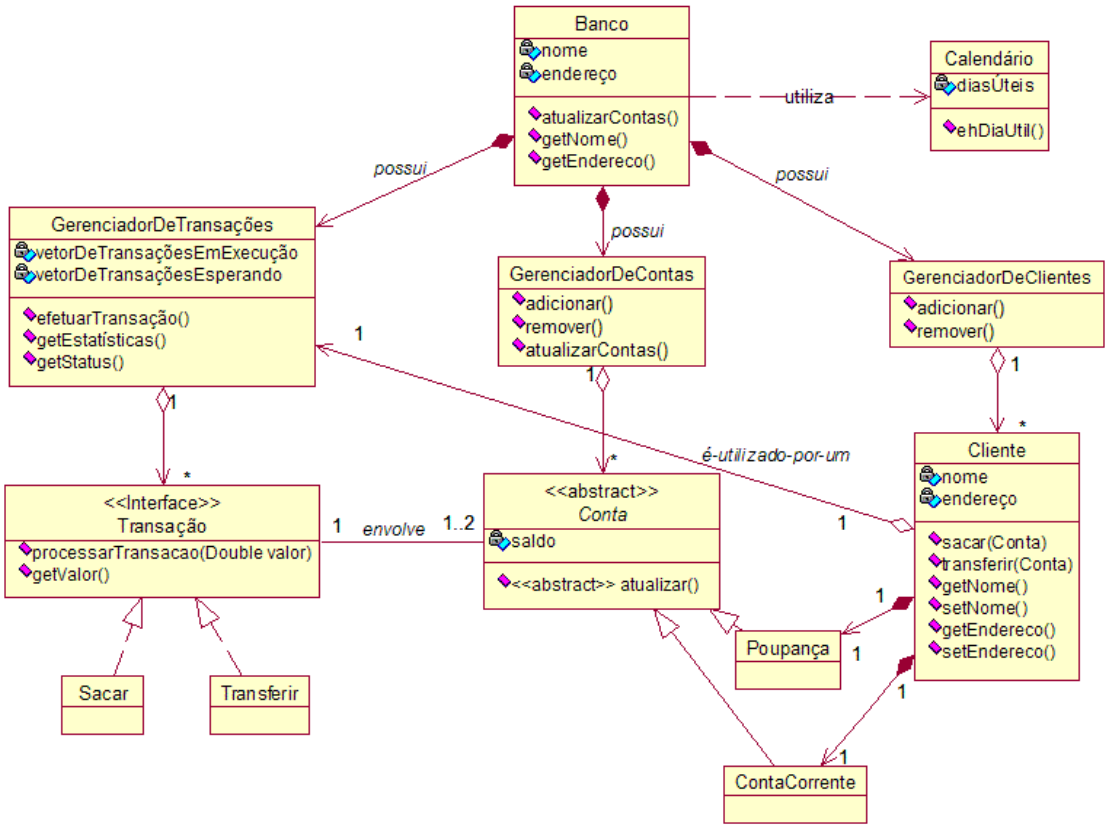


Fonte: (COMPUTAÇÃO UFCG, 2016)

- Classes de Software

Aborda vários detalhes de implementação, tais como navegabilidade, tipo dos atributos, entre outros, possui a perspectiva destinada ao time de desenvolvimento. Conforme Figura 14.

Figura 14: Exemplo de representação de classes de *software*



Fonte: (COMPUTAÇÃO UFCG, 2016)

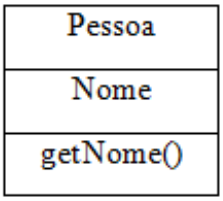
Um diagrama de classes contem: (RIBEIRO, 2016)

- Entidades;
- Relacionamentos.

Por sua vez as entidades são dividias em: (RIBEIRO, 2016)

- Classe:
    - Classe Concreta: uma classe é representada na forma de um retângulo, contendo duas linhas que separam três partes. A primeira contém no nome da classe, a segunda os atributos da classe e a última os métodos da mesma. Conforme a Figura 15, abaixo.

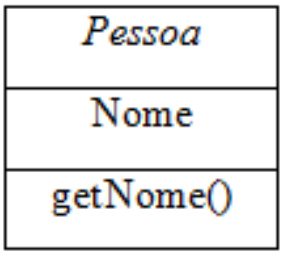
Figura 15: Exemplo de Representação de classes concretas



Fonte: Autores

- Classe Abstrata: a representação de uma classe abstrata em UML é quase igual à representação de uma classe concreta, a única diferença é o estilo da fonte do nome da classe, que, neste caso, está em itálico. Conforme a Figura 16.

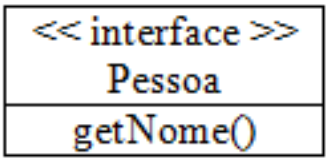
Figura 16: Exemplo de representação de classe abstrata



Fonte: Autores

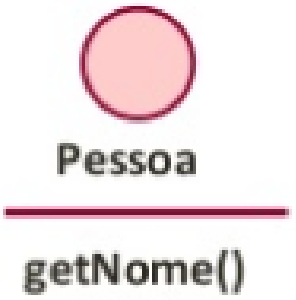
- Interface: As interfaces são representadas no diagrama em dois tipos a *Icon*, expressa na Figura 17, e a *Label* representada na Figura 18.

Figura 17: Exemplo de representação *Icon.*



Fonte: Autores

Figura 18: Exemplo de representação *Label.*



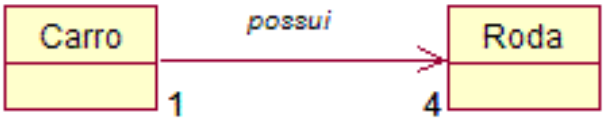
Fonte: (COMPUTAÇÃO UFCG, 2016)

- Perspectivas:
    - Conceitual: Onde apenas classes são utilizadas. Neste tipo de perspectiva, uma classe é interpretada como um conceito. Apenas atributos são utilizados.
    - Especificação: Tanto classes como interfaces são utilizados neste tipo de perspectiva. O foco consiste em mostrar as principais interfaces e classes juntamente com seus métodos.
    - Implementação: Nesta perspectiva, são abordados os detalhes de implementação.

Para Ribeiro (2016) os relacionamentos são divididos em:

- Papel: que descreve o relacionamento. Conforme a Figura 19.

Figura 19: Exemplo de representação de papel.



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Multiplicidade: que é utilizado em todas as perspectivas de forma uniforme. A Tabela 2 mostra as notações possíveis e a Figura 20 a representação gráfica.

Tabela 2: Exemplo de notações possíveis no diagrama de classes

Tipos	***Significa***
0..1	Zero ou uma instância. A notação n..m indica n para m instâncias.
0..* ou *	Não existe limite para o número de instâncias.
1	Exatamente uma instância.
1..*	Ao menos uma instância.

Fonte: (COMPUTAÇÃO UFCG, 2016)

Figura 20: Exemplo de representação gráfica de multiplicidade

Fonte: (COMPUTAÇÃO UFCG, 2016)

- Associação: que é aplicada em todas as perspectivas conforme a Figura 21.

Figura 21: Exemplo da representação gráfica da associação

Fonte: (COMPUTAÇÃO UFCG, 2016)

- Herança ou Generalização: utilizado em todas as perspectivas. Representada na Figura 22.

Figura 22: Representação gráfica da herança

Fonte: (COMPUTAÇÃO UFCG, 2016)

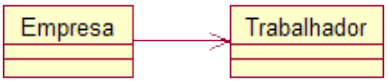
- Navegabilidade: um relacionamento sem navegabilidade implica que ele pode ser lido de duas formas, isto é, em suas duas direções. Conforme Figura 23.

Figura 23: Representação gráfica da navegabilidade

Fonte: (COMPUTAÇÃO UFCG, 2016)

Pode-se observar na Figura 23 que uma empresa possui um trabalhador, como também um trabalhador trabalha em uma empresa. Assim utilizando a propriedade de navegabilidade, pode–se restringir a forma de ler um relacionamento. Assim em vez de ter duas direções, aparecera apenas uma direção de acordo com a direção da navegação. Conforme a Figura 24.

Figura 24: Representação gráfica da navegabilidade.



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Agregação: Agregação é uma associação em que um objeto é parte de outro, de tal forma que a parte pode existir sem o todo. De forma geral, utiliza-se agregação para enfatizar detalhes de uma futura implementação (perspectiva de implementação). Conforme pode-se observar na Figura 25.

Figura 25: Representação gráfica da agregação

Fonte: (COMPUTAÇÃO UFCG, 2016)

- Composição: define-se como passagem por valor. O todo contém as partes (e não referências para as partes). Quando o todo desaparece, todas as partes também desaparecem. Conforme a Figura 26.

Figura 26: Representação gráfica da composição

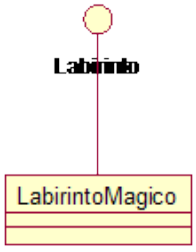
Fonte: (COMPUTAÇÃO UFCG, 2016)

- Implementação: Utilizado para indicar que uma classe implementa uma interface, como pode-se observar nas Figura 27, Figura 28 e Figura 29 abaixo.

Figura 27: Representação gráfica da implementação

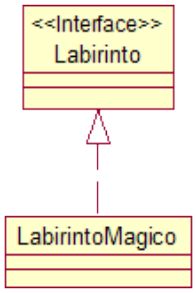
Fonte: (COMPUTAÇÃO UFCG, 2016)

Figura 28: Implementação de uma interface representada por um círculo



Fonte: (COMPUTAÇÃO UFCG, 2016)

Figura 29: Implementação de uma interface representada por um retângulo



Fonte: (COMPUTAÇÃO UFCG, 2016)

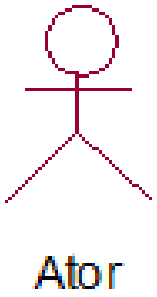
1.14.3 Diagrama de sequência

O diagrama de sequência tem como objetivo mostrar como as mensagens entre os objetos são trocadas no decorrer do tempo para a realização de uma operação. (COMPUTAÇÃO UFCG, 2016), ou seja, ele mostra uma interação, assim apara a sua criação ele necessita dos diagramas de classe e casos de uso, pois trata-se de interações de objetos de um determinado caso de uso, atribuindo assim as responsabilidades a cada um dos objetos no sistema.

O diagrama de sequência é composto por: (COMPUTAÇÃO UFCG, 2016)

- Atores: São responsáveis pelo inicio do processo, e interage com o sistema e solicita serviços, são representados conforme a, Figura 30, a seguir:

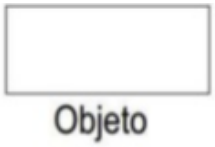
Figura 30: Exemplo de representação de Atores



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Objetos: São representados por retângulos, conforme a figura 31, no topo do diagrama e a sua nomenclatura é “nome_do_objeto:Sua_classe”.

Figura 31: Exemplo de representação de objetos



Fonte: (COMPUTAÇÃO UFCG, 2016)

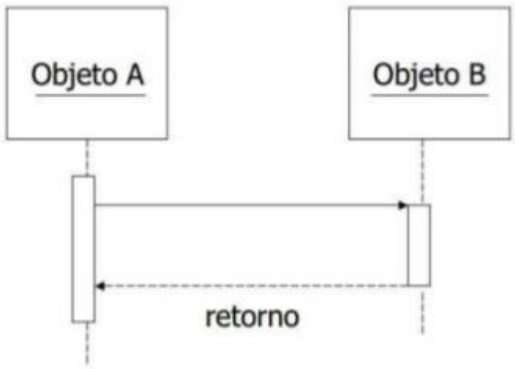
- Mensagens: Podem ser:
    - Simples, onde mostra como o controle é assado de um objeto para o outro, sem detalhes. É representado pela Figura 32

Figura 32: Representação gráfica de mensagem simples

Fonte: (COMPUTAÇÃO UFCG, 2016)

    - Síncrona, são mensagens que implicam um sincronismo rígido entre os estados do objeto que envia a mensagem e os objetos de destino das mensagens. Ou seja, o objeto que enviou a mensagem tem que aguardar a conclusão do processamento da mensagem feito pelo objeto de destino. Conforme a.Figura 33.

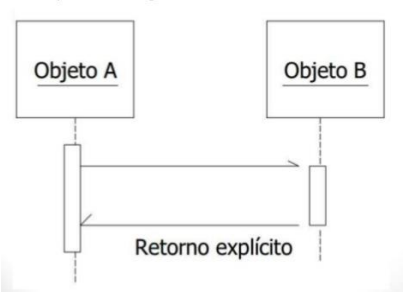
Figura 33: Exemplo de mensagem síncrona



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Assíncrona, são mensagens enviadas de um objeto a outro sem que haja uma dependência de estado entre os dois objetos. Ou seja, o objeto de origem envia a mensagem e prossegue seu processamento, independentemente do processamento feito pelo objeto de destino. Conforme a.Figura 34.

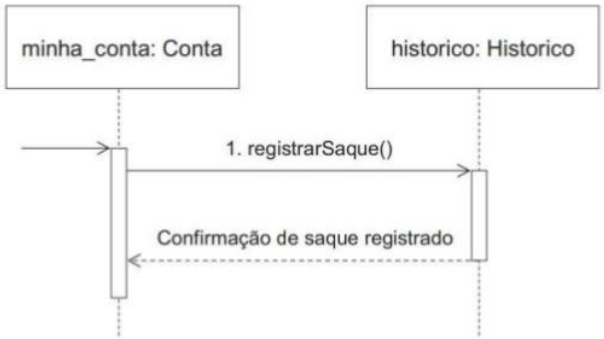
Figura 34: Exemplo de mensagem assíncrona



Fonte: (COMPUTAÇÃO UFCG, 2016)

- Retorno, indica respostas ao ator e respostas para objetos, conforma a Figura 35

Figura 35: Exemplo de mensagem de retorno



Fonte: (COMPUTAÇÃO UFCG, 2016)

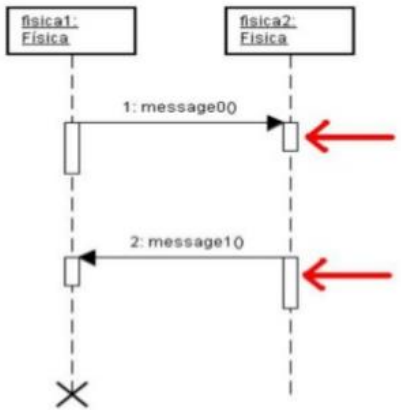
- Linha da vida: Representa o tempo em que um objeto existe durante o processo, é representada de acordo com a Figura 36, é localizada abaixo do objeto.

Figura 36: Exemplo de linha da vida

Fonte: (COMPUTAÇÃO UFCG, 2016)

- Foco no controle: Indica os períodos em que um determinado objeto está participando ativamente do processo, é representado pela Figura 37.

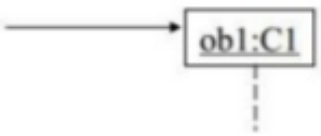
Figura 37: Exemplo de foco no controle



Fonte: (COMPUTAÇÃO UFCG, 2016)

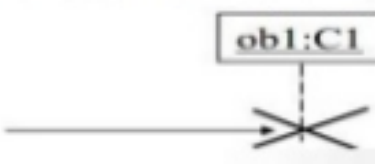
- Criação e destruição dos objetos: A criação do objeto é representada pela Figura 38, sendo uma mensagem dirigida à própria caixa que representa o objeto, já a destruição é representada pela Figura 39, onde há um "X" no fim da linha.

Figura 38: Representação gráfica de criação de objetos



Fonte: (COMPUTAÇÃO UFCG, 2016)

Figura 39: Representação gráfica de destruição de objetos



Fonte: (COMPUTAÇÃO UFCG, 2016)

1.15 Biblioteca Virtual

Marchiori (1997) define a biblioteca virtual como o "conceito de biblioteca virtual está relacionado ao conceito de acesso, por meio de redes, a recursos de informação disponíveis em sistemas de base computadorizada, normalmente remotos".

Rowley (2002) define ainda que a biblioteca virtual independe de local, é acessada e fornecida pelas redes de comunicações. O usuário pode acessar a informação a partir de qualquer ponto, e a informação pode estar em qualquer lugar; é irrelevante para o usuário saber onde a informação é mantida. No desenvolvimento de uma biblioteca virtual, a padronização é o processo essencial para caracterização, preservação dos documentos e interoperabilidade.

Um dos benefícios da utilização de uma biblioteca virtual é o acesso em qualquer tempo e lugar, prescindindo da ida ao ambiente físico de uma biblioteca. A rapidez do acesso à informação, facilitada pela consulta a bases de dados, substitui os antigos métodos de consulta na antiga biblioteca. (CUENCA et. al, 2009)

Ainda segundo Cuenca et. al (2009), a necessidade de informação com validade científica e de forma rápida é uma vantagem da biblioteca virtual. Outra vantagem é a melhoria da busca bibliográfica, com sistemas integrados de bases de dados possibilitando buscas simultâneas, interfaces personalizadas e serviços em rede que permitem navegação em inúmeras coleções. Abrangendo assim maior número de usuários sendo seu acesso universal e ilimitado, promovendo assim a inclusão digital.

Na implementação de uma biblioteca virtual, questões como direito autoral e propriedade intelectual, legitimidade e recuperação da informação devem ser plenamente discutidas para um bom desenvolvimento do projeto. (CARVALHO et.al, 2004).

É importante que as bibliotecas virtuais utilizem mecanismos de segurança para garantir a integridade de sua informação, e estudos também devem ser continuados com a intenção de garantir a recuperação do documento. (CARVALHO, et.al, 2004).

A plataforma Web é considerada um dos suportes mais utilizados e crescentes que dinamiza, sociabiliza, e inova informações, ao usuário onde quer que esteja, por isso ao mencionar o termo dinamizar e sociabilizar, nos abre precedente para inovar criando produtos e serviços de informação. (CARVALHO, et. al, 2004).

CAPÍTULO II

Estrutura do sistema

Neste capítulo apresenta-se o modelo de casos de uso utilizado, representando os fluxos e os usuários envolvidos em cada ação do sistema.

2.1 Estudo de Viabilidade

O estudo de viabilidade visa à tomada de decisão, verificando se um sistema é possível ou não de ser realizado. Nesse caso pode-se afirmar, que esse sistema *web* de TCC apresentará, uma proposta de solução para o problema atual, pois possuem objetivos e requisitos funcionais e não funcionais, trazendo vantagens e outras alternativas para o sistema atual.

1.16 Engenharia de Requisitos

Apresenta-se a seguir, as etapas da engenharia de requisitos, possibilitando assim o entendimento das atividades que contribuem para a produção e manutenção do *software*.

1.16.1 Propósito

Verificou-se a necessidade de desenvolver um sistema para ajudar a comunidade acadêmica a ter mais facilidade e agilidade na criação de TCC's. Deste modo apresenta-se o projeto de sistema web para TCC, que tem como objetivo suprir essa deficiência.

1.16.2 Administrador

É o responsável por gerenciar os trabalhos e definir quais serão aprovados para publicação ou reprovados.

1.16.3 Usuários

São os responsáveis por utilizarem o sistema para divulgar seus trabalhos ou consultar quando necessário. Podem ser alunos, discentes ou professores.

1.17 Restrições do sistema

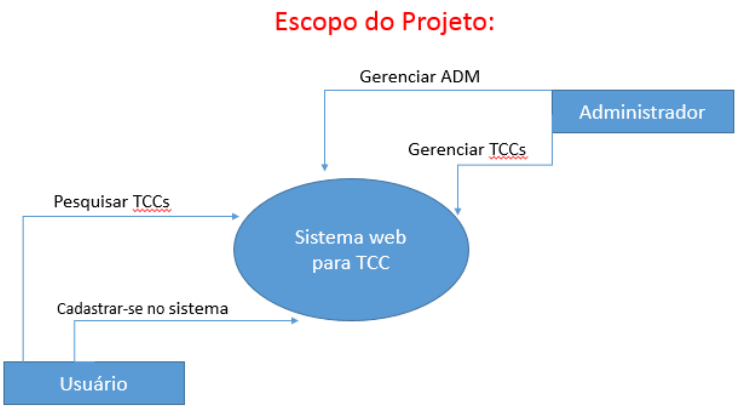
Apresenta-se abaixo, a lista de restrições do sistema:

- O sistema deverá realizar todo o gerenciamento dos trabalhos envolvidos, e registrar aqueles que serão aprovados pelos administradores.
- O sistema manterá o cadastro dos usuários, o qual será obrigatório para realizar a consulta ou para adicionar algum trabalho novo.
- O administrador consultará o sistema para verificar os trabalhos aprovados para inclusão deles no cadastro web.
- O usuário consultará o sistema para realizar as pesquisas dos trabalhos, ou cadastrar algum projeto novo.
- O sistema deverá interagir com o usuário e o administrador para registrar os trabalhos.

1.18 Escopo do projeto

O escopo do projeto é criar um sistema para arquivamento virtual de monografias realizadas na FATEC, onde hoje é apenas realizado por consulta aos arquivos físicos na biblioteca. Observado este aspecto, procurou-se desenvolver uma solução utilizando tecnologia da informação, possibilitando o acesso aos trabalhos por meio de computadores, celulares, *tablets*, e qualquer dispositivo com acesso à internet. Isso traz uma maior flexibilidade para o aluno e permite que o conteúdo esteja disponível 24 horas por dia.

Figura 40: Escopo do projeto



Fonte: Autores

1.19 Requisitos funcionais

Apresenta-se como requisitos funcionais por parte do administrador, ações essenciais para manter e administrar o conteúdo que está sendo publicado no site e também avaliar conteúdo das monografias enviadas com fim de identificar quaisquer erros que impossibilitem a publicação no sistema.

i. Requisito funcional 1- O sistema deverá manter o cadastro dos usuários.

O sistema deverá manter (criar, alterar, consultar, deletar) todos os dados referentes aos administradores, para que assim se possa obter uma base de dados confiável com diferentes níveis de acesso e permissões permitindo que os usuários façam o gerenciamento dos trabalhos devidamente.

ii. Requisito funcional 2 - O sistema deverá manter o cadastro dos trabalhos.

O sistema deverá manter (criar, alterar, consultar, deletar) todos os dados referentes aos trabalhos podendo assim categoriza-los organizadamente: os aprovados, que serão publicados no sistema para consulta do usuário e os recusados, que não serão publicados no site, sendo necessário reenviar um novo convite para o autor fazer as alterações necessárias para sua monografia ser aceita para publicação.

iii. Requisito funcional 3 - O sistema deverá permitir que o administrador insira novos trabalhos.

O sistema deverá permitir que o administrador possa inserir novos trabalhos como publicações no site, as monografias não devem conter nenhum erro de direitos autorais ou qualquer outro erro que impeça a publicação da monografia no site.

iv. Requisito funcional 4 - O sistema deverá permitir que o administrador receba todos os tipos de trabalhos para aprova-los ou não se necessário.

O sistema deverá fornecer uma estrutura onde o administrador possa analisar a monografia fazendo o download do arquivo, após ler a monografia e não constatar nenhum erro, o arquivo será aprovado pelo administrador e assim automaticamente será publicado no site para consulta dos alunos ou qualquer um que tenha acesso ao sistema. Caso encontre algum erro a monografia é recusava e não ficará disponível para consultas. O autor é notificado por email em caso de aprovação ou reprovação dos trabalhos.

v. Requisito funcional 5 - O sistema deverá permitir que o administrador e o usuário, consultem os trabalhos.

O sistema deverá permitir que o administrador e os usuários possam consultar as monografias. Com a diferença que o usuário só tem a liberdade de consultar as monografias por cursos ou na barra de pesquisa e fazer o download. Já o administrador tem uma pesquisa mais apurada para manter organizado as monografias que estão publicadas no sistema, os recusados e os pendentes.

vi. Requisito funcional 6 - O sistema deverá atualizar o banco de dados

O sistema deverá atualizar o banco de dados a cada alteração que o administrador realizar, seja ela na aprovação de uma monografia, recusa ou deletar uma monografia já publicada anteriormente. O administrador também pode alterar informações referentes aos cursos ou as próprias informações de cadastro.

1.20 Requisitos não funcionais

Apresenta-se como requisitos não funcionais a portabilidade e mobilidade que se tornam essenciais para todo o sistema web. O sistema por utilizar o método responsivo, diminui o tempo de resposta, e propõe uma pesquisa mais interativa aos usuários.

1.20.1 Requisitos de usabilidade

O sistema possui como característica ser bem simples, contendo as informações de maneira clara, para ser bem prático e rápido nas consultas dos trabalhos de TCC.

1.20.2 Requisitos de eficiência

Como todo sistema, esse foi criado para facilitar a consulta de TCC, para agilizar no processo de busca do usuário ao realizar uma pesquisa para o desenvolvimento do trabalho.

1.20.3 Requisitos de portabilidade

O sistema está disponível nas plataformas Windows e Linux, pode ser acessado por qualquer navegador como Internet Explorer, Google Chrome e Firefox, além de ser responsivo para as telas de computadores, tabletes e smartphones.

1.20.4 Requisitos de confiabilidade

Os trabalhos aprovados devem estar disponíveis para consulta dentro do sistema.

1.20.5 Requisitos de implementação

O sistema para web, deve ser desenvolvido na linguagem PHP, utilizando o framework do Bootstrap, e o Laravel para implementação.

1.20.6 Requisitos de interoperabilidade

O sistema de consulta de TCC deve se comunicar com o sistema de cadastro de usuário.

1.20.7 Requisitos Éticos

O sistema de consulta web para TCC deve ser aberto a todo usuário cadastrado. Possui um termo de direitos autorais sobre o autor da obra.

1.20.8 Requisitos de Segurança

Os dados cadastrais dos usuários e os trabalhos de pesquisa somente devem ser exibidos após a validação da senha.

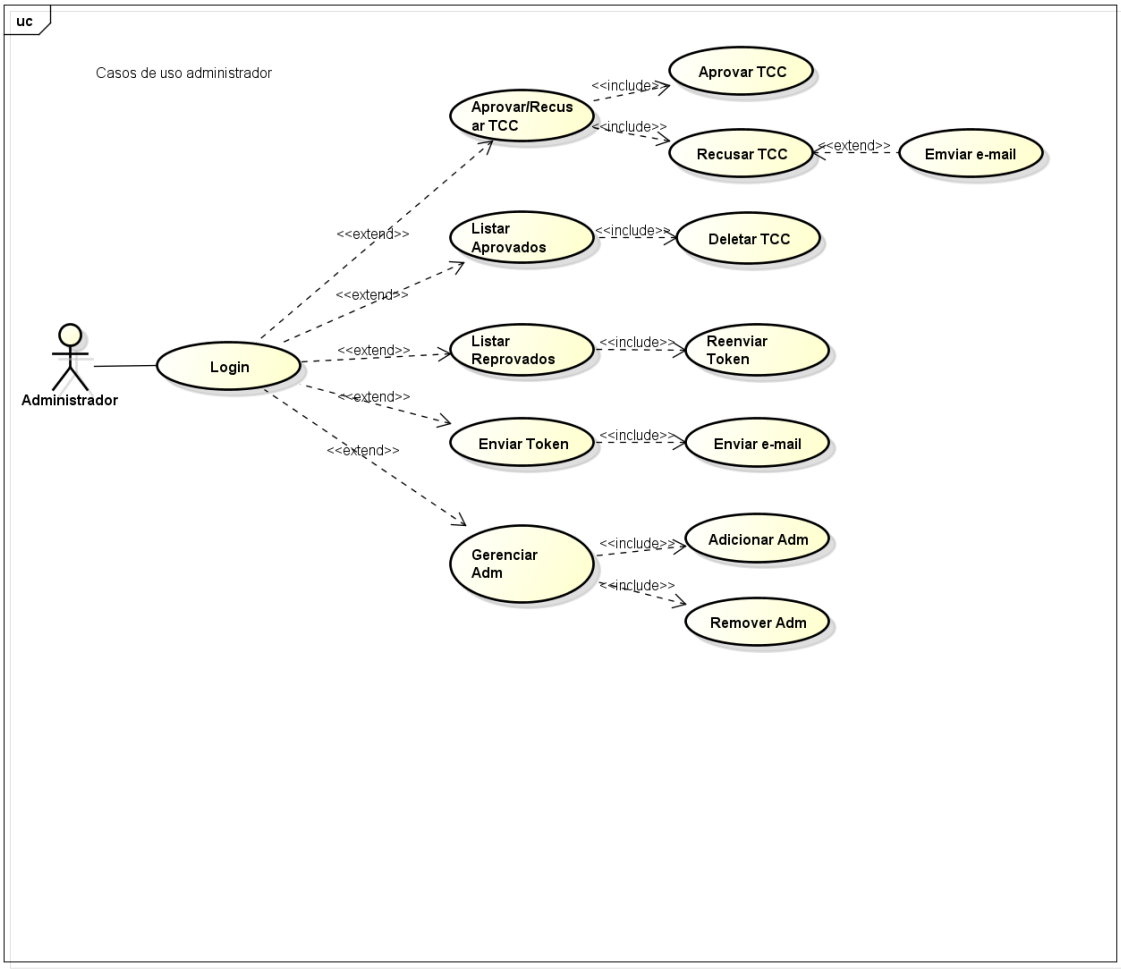
1.21 Descrição das Regras de Negócio

Somente usuários, alunos e discentes cadastrados no sistema poderão consultar os projetos de TCC da Fatec. Os trabalhos recebidos poderão ser de qualquer instituição, desde que, para cadastrar um projeto o TCC precisa ser aprovado, caso contrário, não será feito o cadastro.

1.22 Especificação e Diagrama de Casos de Uso

A seguir, se apresenta os diagramas dos Casos de Uso do Sistema representando as possíveis ações dos usuários e administradores do sistema.

Figura 41: Diagrama dos casos de uso: Administrador



Fonte: Autores

O diagrama representado pela Figura 41 descreve de modo gráfico todas as possíveis ações que os administradores podem ter no sistema. Após um administrador fazer o *login* na plataforma ele vai ter a sua disposição um menu lateral que irá listar todas as ações disponíveis para ser realizadas na plataforma. Cada ação será representada e descrita nas tabelas abaixo:

A Tabela 3 descreve o caso de uso Administrador, onde se resume no início da interação dos administradores com a plataforma. Essas interações são ações que permitem com que os administradores cadastrem outros administradores para gerenciar a plataforma, façam o convite de *upload* de trabalhos para os alunos, aprovem ou reprovem trabalhos e façam a gestão dos cursos que estão disponíveis na plataforma.

Tabela 3: Descrição do caso de uso Administrador

ATOR: Administrador
ID: A2
Resumo: O administrador interage com o usuário e o sistema para registrar os trabalhos.
Responsabilidades: Cadastrar novo projeto, atualizar banco de dados, registrar status do trabalho e enviar e-mail.
Ambiente Físico: O administrador permanece em um local que tenha acesso ao computador para registrar os projetos.
Número e tipo: Apenas um administrador para gerenciar os projetos
Frequência com que usa o sistema: O administrador utiliza o sistema constantemente para cadastrar e realizar consultas.

Fonte: Autores

Na

Tabela 4 o caso de uso representa o *login* dos usuários no sistema, para se realizar o fluxo com sucesso é necessário que o administrador possua um e-mail e uma senha previamente cadastradas, caso as credenciais sejam válidas o administrador poderá ter acesso à plataforma.

Tabela 4: Caso de uso efetuar *login*

Caso de uso: Efetuar *login*
Resumo: O Administrador vai se conectar ao sistema através da validação do *login.*
Atores: Administrador
Pré-condições: 1. Possuir um usuário e senha. 2. Ter acesso a um computador com o sistema instalado.
Fluxo de evento principal: O Administrador deve entrar no sistema por meio de *login*
Pós-condições: O Administrador deve permanecer logado no sistema

Fonte: Autores

A Tabela 5 representa o fluxo de aprovação e reprovação de trabalhos. Após um aluno fazer o upload de um trabalho no sistema o administrador poderá aprovar ou reprovar este trabalho. Caso aprovado o trabalho poderá aparecer na listagem principal para ser consultado pela comunidade acadêmica. Caso seja reprovado o aluno receberá um e-mail informando o ocorrido.

Tabela 5: Caso de uso aprovar/recusar TCC

Caso de uso: Caso de uso aprovar/recusar TCC
Resumo: O administrador vai analisar os trabalhos e classificá-los em aprovados ou reprovados
Atores: administrador
Pré-condições: 1. Estar logado no sistema 2. Ter acesso aos trabalhos cadastrados
Fluxo de evento principal: 1. O administrador deve fazer uma avaliação sobre o TCC apresentado
Pós-condições: O Administrador deve incluir os trabalhos na listagem

Fonte: Autores

A Tabela 6 o fluxo de aprovação de trabalhos. Após o *upload* ser realizado, os administradores poderão fazer o *download* do mesmo e analisarem, caso o trabalho esteja dentro das normas os administradores poderão aprovar para que este trabalho fique disponível para a sociedade acadêmica. Ao ser aprovado o usuário que fez o *upload* recebe um e-mail informando que o trabalho foi aprovado e o *link* que o mesmo pode ser consultado.

Tabela 6: Caso de uso aprovar TCC

Caso de uso: Caso de uso aprovar TCC
Resumo: O administrador irá analisar caso o TCC contenha alguma irregularidade para ser publicado no site.
Atores: administrador
Pré-condições: O administrador deverá ter feito o *login* no sistema para realizar a avaliação
Fluxo de evento principal: 1. O administrador irá analisar os TCC's fazendo o *download* do arquivo, procurando possíveis erros que impossibilitariam o trabalho de ser publicado.
Pós-condições: Caso o administrador aprove o TCC, um e-mail será enviado automaticamente ao autor informando que o TCC foi aprovado e será publicado.

Fonte: Autores

A Tabela 7 representa o fluxo de reprovações de trabalhos. Os administradores ao analisarem um trabalho e encontrarem alguma irregularidade podem optar por reprovar o trabalho e não permitir que o mesmo fique disponível para a comunidade acadêmica. Quando um trabalho é reprovado o aluno que fez o *upload* recebe um e-mail informando sobre o ocorrido.

Tabela 7: Caso de uso recusar TCC

Caso de uso: Caso de uso recusar TCC
Resumo: O administrador ira poder visualizar os TCC's reprovados e também se quiser poderá reenviar o *token* para o autor fazer um novo *upload.*
Atores: administrador
Pré-condições: O administrador deverá fazer o *login* no sistema
Fluxo de evento principal: 1. O administrador ira clicar no botão recusar TCC 2. Enviara novo *token* na tela de reprovados

Fonte: Autores

A Tabela 8 representa o fluxo de listagem de trabalhos aprovados. Os administradores podem listar todos os trabalhos que já foram aprovados na plataforma. Esta listagem contêm

informações como: tema do TCC, e-mail do usuário que fez o *upload*, data de envio, link para visualização e um botão para reprovar caso necessário. Essa listagem dispõe de filtros que permitem a filtragem por qualquer informação referente ao trabalho bem como a ordenação por qualquer uma das colunas citadas. Os dados são agrupados em paginações e podem ser dispostos em 10, 25, 50 ou 100 por página à escolha do administrador.

Tabela 8: Caso de uso listar aprovados

Caso de uso: Caso de uso listar aprovados
Resumo: O administrador após classificar o TCC em aprovado irá inseri-lo na lista.
Atores: administrador
Pré-condições: 1. Estar logado no sistema 2. Ter acesso aos trabalhos cadastrados
Fluxo de evento principal: 1. O administrador deve inserir o TCC apresentado na lista de aprovados.
Pós-condições: O administrador deve incluir os trabalhos na listagem de aprovados.
Fluxo de eventos alternativos: O administrador após incluir os trabalhos, verificará toda a listagem dos trabalhos aprovados. Em **{visualizar aprovados}** se o administrador escolher a opção aprovar: O administrador deverá visualizar os trabalhos aprovados no sistema.

Fonte: Autores

A Tabela 9 representa o fluxo de listagem de trabalhos reprovados. Os administradores podem listar todos os trabalhos que já foram reprovados na plataforma. Esta listagem contêm informações como: tema do TCC, e-mail do usuário que fez o upload, *link* para download, botão para reenviar o convite caso necessário, data de reprovação, e quem foi o usuário que realizou a reprovação. Essa listagem dispõe de filtros que permitem a filtragem por qualquer informação referente ao trabalho bem como a ordenação por qualquer uma das colunas citadas. Os dados são agrupados em paginações e podem ser dispostos em 10, 25, 50 ou 100 por página à escolha do administrador.

Tabela 9: Caso de uso listar reprovados

Caso de uso: Caso de uso listar reprovados
Resumo: O administrador após classificar o TCC em reprovado irá inseri-lo na lista.
Atores: administrador
Pré-condições: 1. Estar logado no sistema 2. Ter acesso aos trabalhos cadastrados
Fluxo de evento principal: 1. O administrador deve inserir o TCC apresentado na lista de reprovados.
Pós-condições: O administrador deve incluir os trabalhos na listagem de reprovados.
Fluxo de eventos alternativos: Em **{visualizar reprovados}** se o administrador selecionar a opção reprovados TCC: O administrador deverá visualizar os trabalhos reprovados no sistema.

Fonte: Autores

A Tabela 10 representa o fluxo envio de convites de *upload*. Os administradores podem enviar e-mails de convite para que alunos realizem o *upload* de trabalhos no sistema. O convite é composto de um *link* com um *token* único a cada envio. Quando o aluno clica nesse *link* o sistema verifica e valida este *token*, caso o mesmo seja válido o aluno pode acessar a página de *upload* de trabalhos, caso seja inválido o sistema exibe uma página de erro.

Tabela 10: Caso de uso enviar *token*

Caso de uso: Caso de uso enviar token
Resumo: O administrador irá enviar o *token* ao usuário após verificar o cadastro dele no sistema.
Atores: administrador
Pré-condições: 1. Estar logado no sistema 2. Usuário cadastrado no sistema
Fluxo de evento principal: O administrador deve enviar o *token* para o usuário entrar no sistema.
Pós-condições: O *token* irá possuir um prazo para expiração.
Fluxo de eventos alternativos: Em **{enviar e-mail}:** 1. Após a aprovação do cadastro o administrador irá enviar um e-mail contendo o *token* de acesso ao sistema. 2. Caso não seja aprovado, também será enviado um e-mail de notificação informando ao usuário.

Fonte: Autores

A Tabela 11 representa o fluxo de remoção dos TCC's. Os administradores podem a qualquer momento remover um TCC que já tenha sido previamente aprovado. Após clicar em remover, o aluno que realizou o *upload* recebe um e-mail informando sobre o ocorrido.

Tabela 11: Caso de uso remover TCC's

Caso de uso: Caso de uso deletar TCC's
Resumo: O administrador poderá remover um TCC que já esteja publicado.
Atores: administrador
Pré-condições: O administrador deverá ter feito *login* no sistema
Fluxo de evento principal: 1. Administrador pode remover um TCC clicando no botão excluir.
Fluxo de eventos alternativos: 1. O administrador escolhera editar TCC 2. Ira preencher os campos novamente se necessário e alterar apenas os campos que deseja, ou até mesmo substituir o arquivo.

Fonte: Autores

A Tabela 12 representa o fluxo de remoção dos TCC's. Os administradores podem a qualquer momento remover um TCC que já tenha sido previamente aprovado. Após clicar em remover, o aluno que realizou o *upload* recebe um e-mail informando sobre o ocorrido (alterar informações da Tabela 12)

Tabela 12: Caso de uso gerenciar ADM

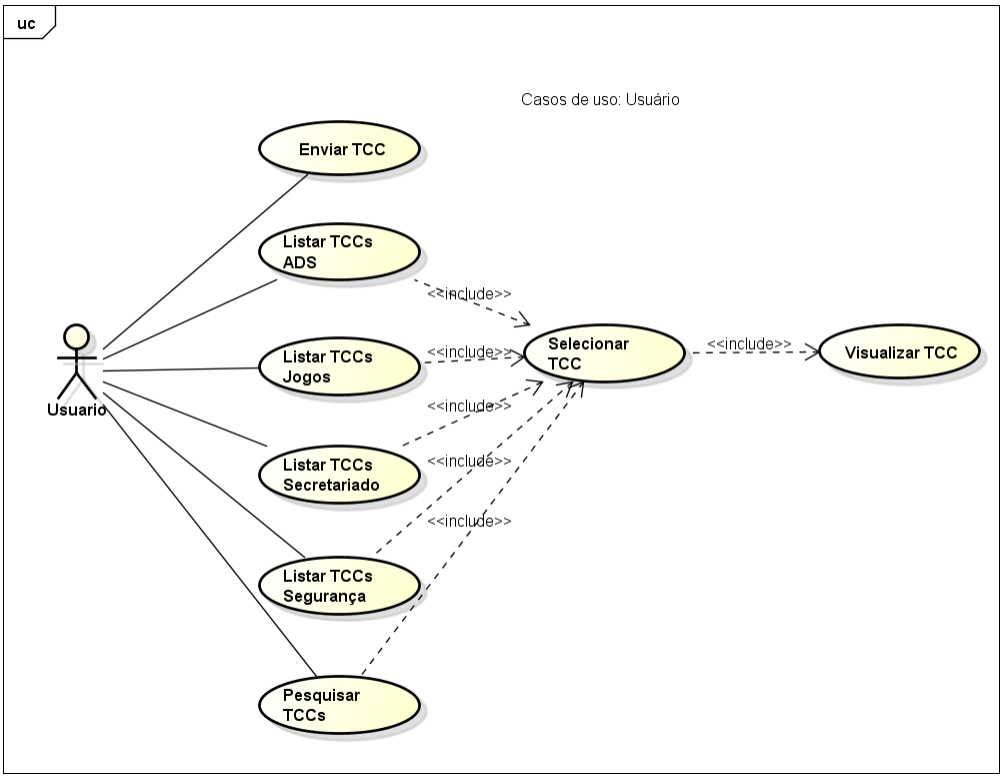
Caso de uso: Caso de uso gerenciar ADM
Resumo: O administrador deve analisar os dados cadastrados pelo usuário para ativar o cadastro ou excluir se necessário.
Atores: administrador e usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao cadastro do usuário.
Fluxo de evento principal: 1. O administrador deve verificar se o usuário está cadastrado ou não no sistema. 2. O sistema deve exibir os dados cadastrais do usuário
Pós-condições: O administrador deve atualizar os dados no sistema
Fluxo de eventos alternativos: Em **{adicionar ADM}** se o administrador selecionar a opção cadastrar: 1. O sistema deve preencher os campos com os dados do usuário. 3. O administrador deve clicar no botão atualizar. 4. O sistema deve verificar se os dados estão coerentes com os campos. 5. O sistema deve atualizar o cadastro no banco de dados. 6. O sistema deve informar ao administrador que os dados do usuário foram inseridos. 7. O sistema deve adicionar mais um cadastro. Em **{remover ADM}** se o administrador selecionar a opção excluir cadastro: 8. O sistema exibe uma tela contendo um campo para pesquisa 9. O administrador pode pesquisar digitando o nome do usuário. 10. O sistema deve consultar no banco de dados. 11. O administrador escolhe o cadastro incorreto e exclui. 12. O sistema enviará um e-mail ao usuário informando que não foi cadastrado.

Fonte: Autores

O diagrama de casos de uso representado na Figura 42 representa as ações que o aluno pode tomar diante ao sistema, fazer pesquisa relacionados a algum curso especifico ou até mesmo

fazer uma pesquisa mais detalhada pelo nome do TCC e por último ele tem a opção visualizar e imprimir um formulário de autorização para publicação do TCC no sistema.

Figura 42: Diagrama de caso de uso: usuário



Fonte: Autores

Abaixo, segue as tabelas que iram descrever a interação do usuário com o sistema. A Tabela 13 representa a interação do usuário com o sistema para consultar os trabalhos de TCC. Neste processo de busca o usuário já tem que estar cadastrado no sistema e pode se cadastrar quantos usuários for necessário.

Tabela 13: Descrição do caso de uso usuário

ATOR: Usuário
ID: A1
Resumo: O usuário interage com o sistema para consultar os trabalhos.
Responsabilidades: Cadastrar-se no sistema e consultar TCC.
Ambiente Físico: O usuário permanece em um local que tenha acesso ao computador para consultar o TCC
Número e tipo: A quantidade de usuários cadastrados varia de acordo com os tipos dos TCC
Frequência com que usa o sistema: O usuário utiliza o sistema constantemente para fazer os projetos.

Fonte: Autores

A Tabela 14 representa uma listagem dos trabalhos referentes ao curso de Análise e Desenvolvimento de Sistemas relacionados com o tema de busca. O usuário pode verificar os trabalhos cadastrados e mantê-los no sistema. Essa listagem irá apresentar os últimos trabalhos adicionados.

Tabela 14: Listar TCCs de Análise e Desenvolvimento de Sistemas

Caso de uso: Caso de uso listar TCCs de Análise e Desenvolvimento de Sistemas
Resumo: O usuário deve cadastrar todos os trabalhos com o tema do curso de Análise e Desenvolvimento de Sistemas para inserir na lista.
Atores: usuário
Pré-condições: 1. Estar logado no sistema. 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema.

Fonte: Autores

A Tabela 15 representa uma listagem dos trabalhos referentes ao curso de Jogos relacionados com o tema de busca. O usuário pode verificar os trabalhos cadastrados e mantê-los no sistema. Essa listagem irá apresentar os últimos trabalhos adicionados.

Tabela 15: Listar TCCs de Jogos

Caso de uso: Caso de uso listar TCCs ADS
Resumo: O usuário deve cadastrar todos os trabalhos com o tema do curso de Jogos para inserir na lista.
Atores: usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema.

Fonte: Autores

A Tabela 16 representa uma listagem dos trabalhos referentes ao curso de Secretariado relacionados com o tema de busca. O usuário pode verificar os trabalhos cadastrados e mantê-los no sistema. Essa listagem irá apresentar os últimos trabalhos adicionados.

Tabela 16: Listar TCCs de Secretáriado

Caso de uso: Caso de uso listar TCCs de Secretariado
Resumo: O usuário deve cadastrar todos os trabalhos com o tema do curso de Secretariado para inserir na lista.
Atores: usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema.

Fonte: Autores

A Tabela 17 representa uma listagem dos trabalhos referentes ao curso de Segurança relacionados com o tema de busca. O usuário pode verificar os trabalhos cadastrados e mantê-los no sistema. Essa listagem irá apresentar os últimos trabalhos adicionados.

Tabela 17: Listar TCCs de Segurança

Caso de uso: Caso de uso listar TCCs de Segurança
Resumo: O usuário deve cadastrar todos os trabalhos com o tema do curso de Segurança para inserir na lista.
Atores: usuário
Pré-condições: 1. Estar logado no sistema. 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema.

Fonte: Autores

A Tabela 18 representa o processo de pesquisa que o usuário irá fazer ao buscar pelo tema do TCC desejado. O usuário digita no campo de pesquisa o tema do TCC, que são enviados para o processo de busca da solicitação dentro do sistema.

Tabela 18: Pesquisar TCCs

Caso de uso: Caso de uso pesquisar TCCs
Resumo: O usuário deve analisar os trabalhos cadastrados no sistema para realizar a pesquisa.
Atores: usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema

Fonte: Autores

A Tabela 19 representa a verificação do trabalho selecionado. Após realizar a pesquisa sobre o tema de TCC buscado, o usuário irá selecionar o TCC em questão e serão exibidos na tela os dados referentes ao trabalho.

Tabela 19: Selecionar TCCs

Caso de uso: Caso de uso selecionar TCCs
Resumo: O usuário deve selecionar o trabalho que irá pesquisar.
Atores: usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve verificar se o trabalho já está com o tema e descrição definidos para listar. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve manter os trabalhos cadastrados no sistema

Fonte: Autores

A Tabela 20 representa a visualização do trabalho selecionado que já foi pesquisado previamente. Para ter acesso a esse trabalho de pesquisa o usuário já possui um tema definido ao qual se propõe.

Tabela 20: Visualizar TCCs

Caso de uso: Caso de uso visualizar TCCs
Resumo: O usuário deve possuir acesso para visualizar os trabalhos de acordo com o tema de pesquisa.
Atores: usuário
Pré-condições: 1. Estar logado no sistema 2. Ter acesso ao trabalho inserido.
Fluxo de evento principal: 1. O usuário deve acessar o trabalho para pesquisa. 2. O sistema deve exibir os trabalhos selecionados de acordo com o tema.
Pós-condições: O usuário deve realizar a pesquisa a qual se propõe.

Fonte: Autores

A Tabela 21 representa as ações que o usuário necessita para enviar o arquivo da monografia para o site, onde é necessário clicar no de termo de autorização que se encontra na *index* e posteriormente preencher os campos necessários juntamente com os anexos da monografia e do termo de autorização.

Tabela 21: Caso de uso Enviar TCC

Caso de uso: Enviar TCC
Resumo: O aluno poderá clicar no botão para poder imprimir o termo de autorização.
Atores: Aluno.
Pré-condições: 3. Ter acesso a um computador com o sistema instalado.
Fluxo de evento principal: 1. Clicar no botão de termo de autorização

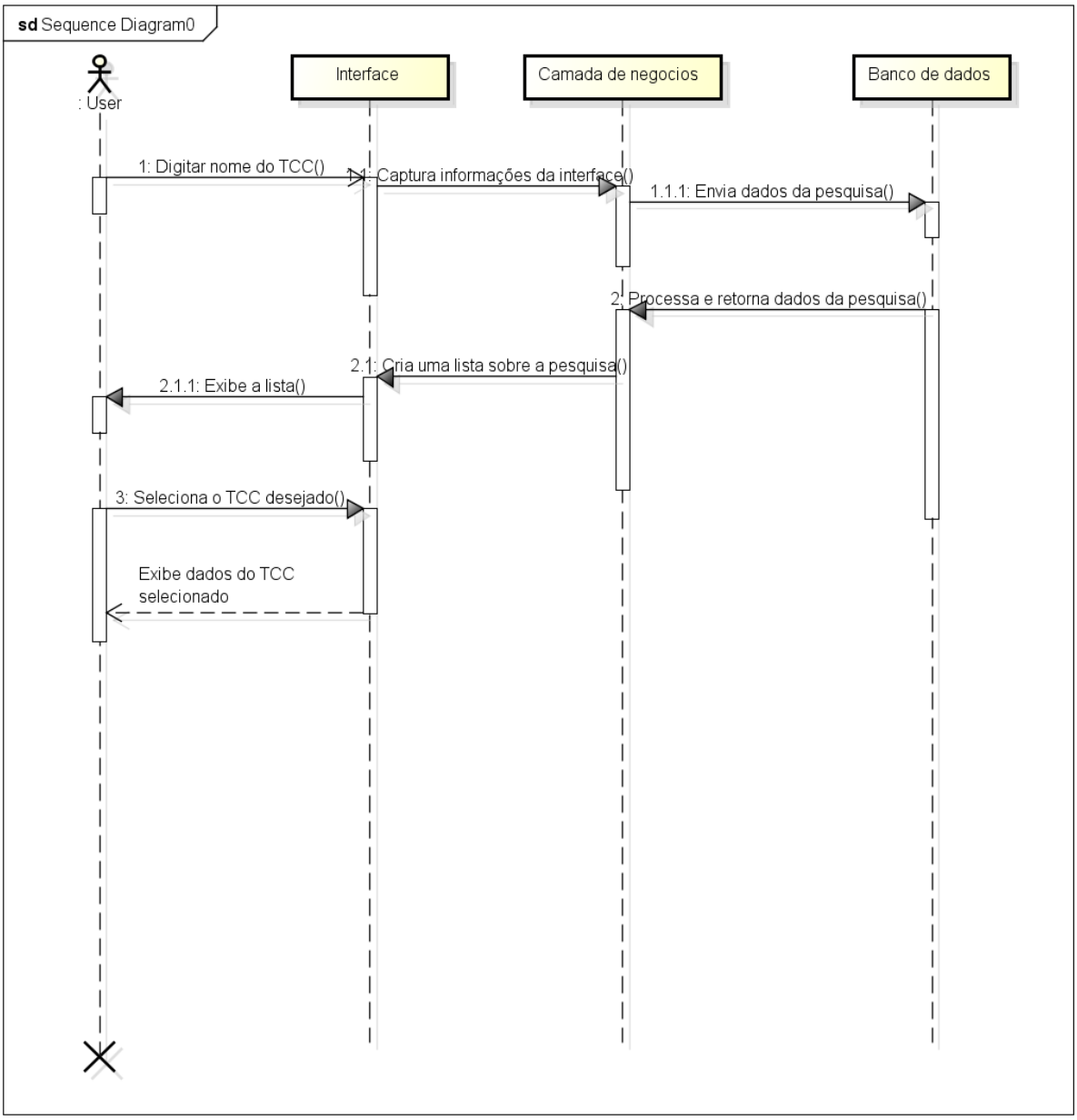
Fonte: Autores

1.22.1 Diagrama de sequência

Nas figuras abaixo são apresentados os fluxos de busca de TCC's demonstrando a interação do usuário com o sistema através do diagrama de sequencias.

No diagrama de sequência da Figura 43, o aluno pode realizar uma pesquisa de um determinado TCC clicando nos botões dos cursos ou também pode realizar uma pesquisa de forma geral digitando no campo de pesquisa o tema. Essa busca pelos dados é feita no banco de dados mysql, o banco envia os dados para a interface que retorna para o usuário. Os resultados serão listados em uma página onde o aluno pode selecionar o TCC que deseja.

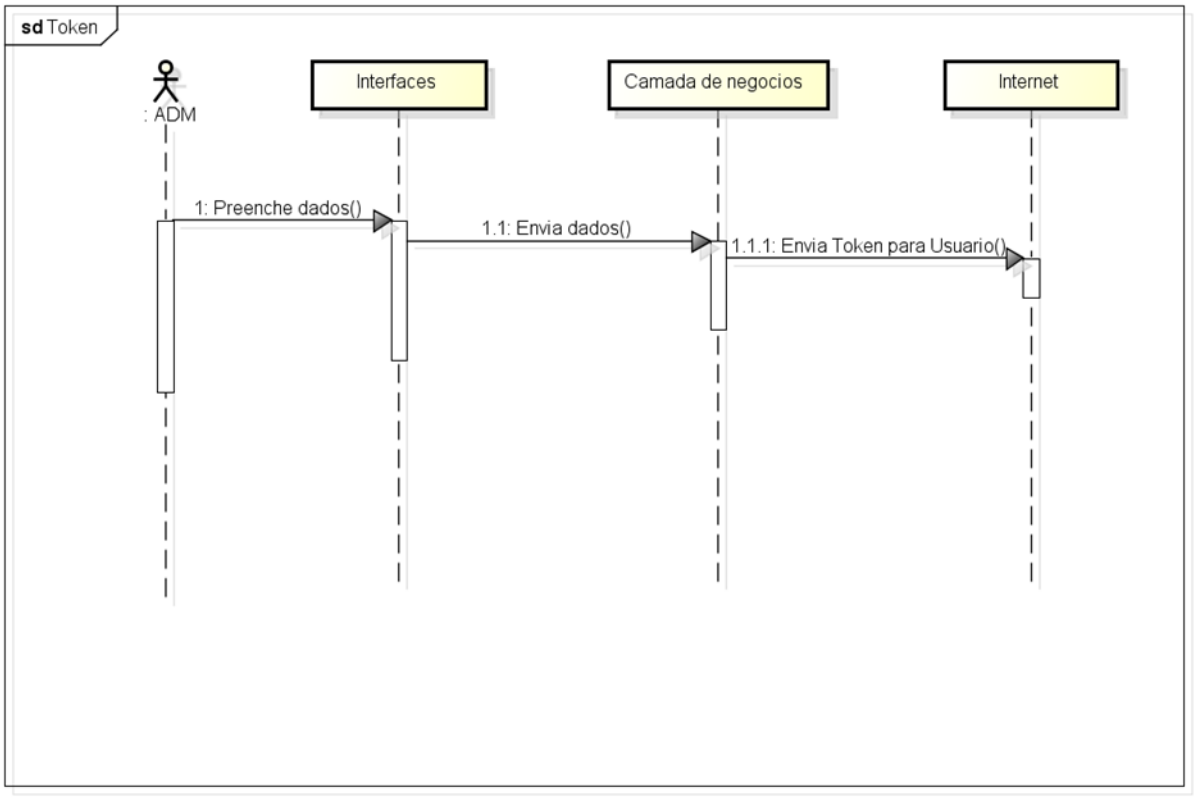
Figura 43: Diagrama de sequencia: usuário



Fonte: Autores

No diagrama a seguir, na Figura 44, o usuário/administrador preenche o campo de texto com o e-mail do usuário. O *token* é gerado para o usuário através de um link enviado por esse e-mail. Nesse processo encontra-se o formulário para ser preenchido pelo usuário com os dados correspondentes, que ficam armazenados na camada de negócios, onde é coletado as informações, para ser enviado novamente ao administrador.

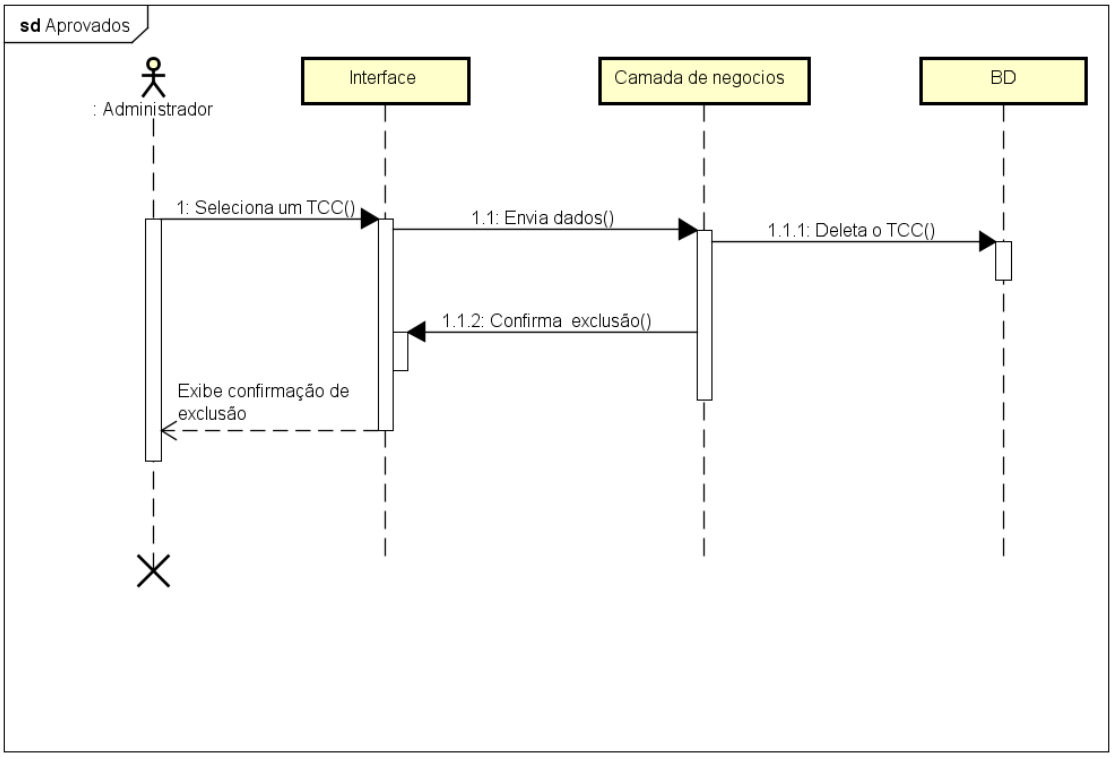
Figura 44: Diagrama de sequência *token*



Fonte: Autores

Na Figura 45, o diagrama de sequência representa as ações que o administrador é possibilitado em seu ambiente na tela de aprovados, o administrador tem o poder de deletar um TCC já aprovado anteriormente, caso note que aprovou por algum engano ou notou um erro tardio no TCC.

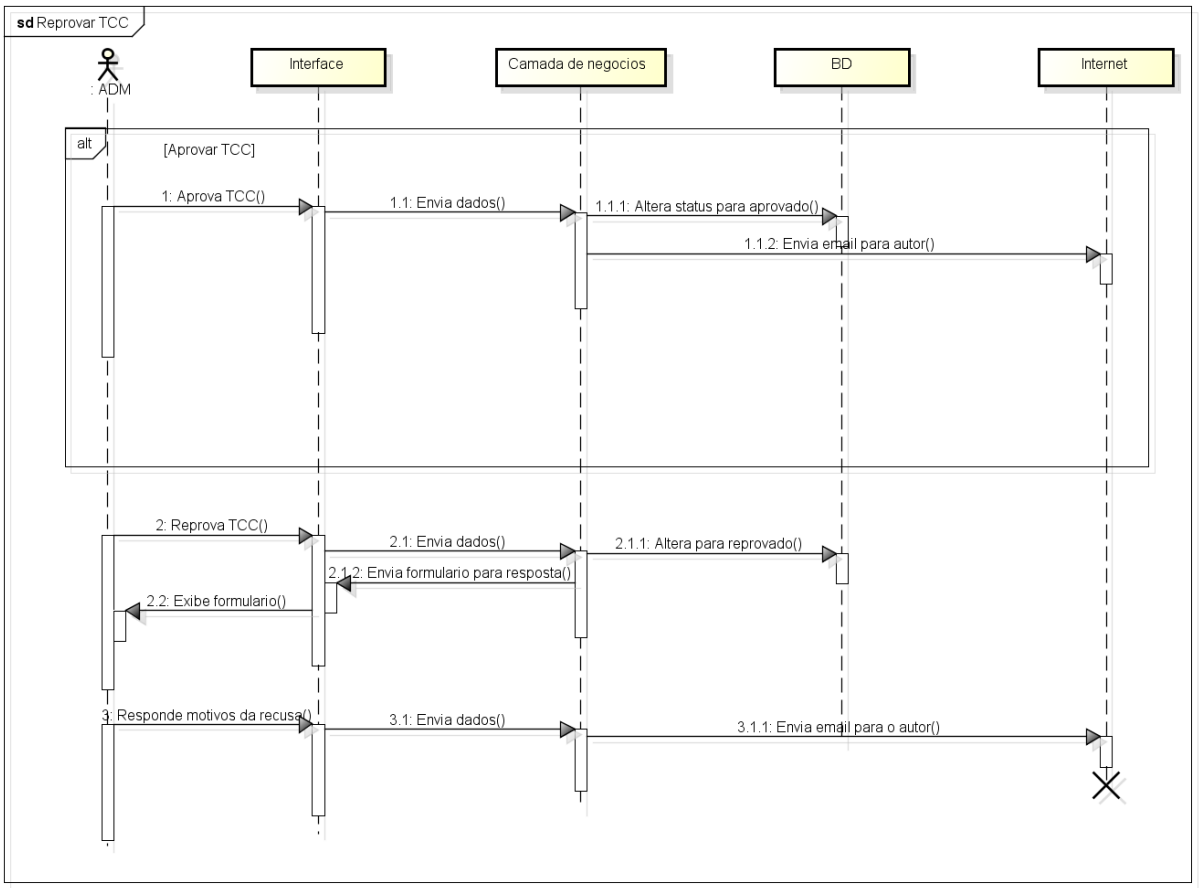
Figura 45: Digrama de sequencia pesquisa de usuário



Fonte: Autores

No diagrama de sequencia apresentado na Figura 46, o administrador envia os dados dos TCCs aprovados para o banco de dados. Após esse processo é enviado o e-mail para o autor com as informações aprovadas. Se reprovar o TCC o administrador envia os dados reprovados para a camada de negócios, a qual altera o status do TCC para reprovado. Logo após é enviado o formulário para resposta do administrador, que responde os motivos de reprovação. O administrador envia um e-mail para o usuário informando o status do TCC, se foi aprovado ou não.

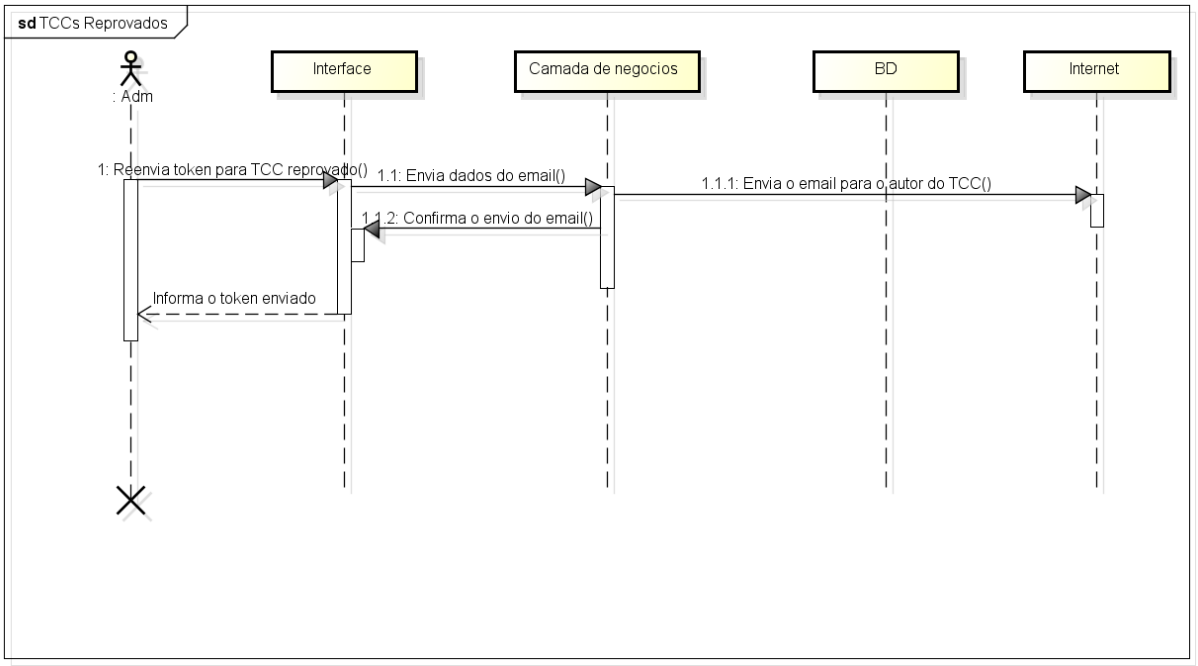
Figura 46: Diagrama de sequencia Aprovar ou Reprovar



Fonte: Autores

No diagrama de sequencias da Figura 47, o administrador reenvia o *token* para os TCC reprovados. A interface receberá os dados cadastrados do e-mail e a camada de negócios envia o e-mail para o autor do TCC. Após o envio a camada de negócios informa o *token* enviado.

Figura 47: Diagrama de sequência reprovados



Fonte: Autores

1.23 Diagrama de classe

Abaixo, na figura 48 está representado o diagrama de classes do sistema, que corresponde às classes construídas sob o padrão MVC adotado para a construção do mesmo.

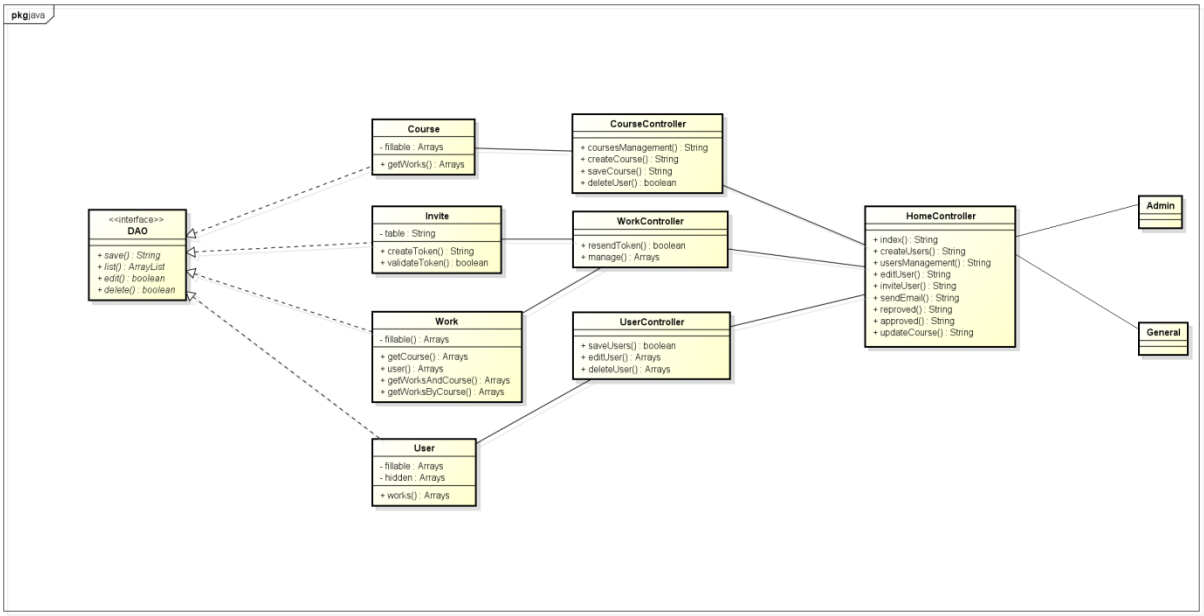
As classes criadas no sistema foram divididas entre *model*, *view* e *controller*. As *views* são as classes responsáveis por todas as telas do sistema. No sistema apresentado possuímos duas classes principais para gerenciar os arquivos das telas do sistema, são elas “Admin” e “*General*”.

A classe “Admin” é responsável por renderizar todas as telas relacionadas a parte administrativa do site, já a classe “General” é responsável por renderizar as telas gerais do sistema, as que estão fora da parte administrativa e que permitem o acesso de todos.

As classes de *controller* são responsáveis por intermediar as comunicações do que foi requisitado na *view* com os dados que estão no banco de dados, fazendo assim validações e retornando o resultado das operações que foram encaminhadas para as *Models*. O sistema é composto das seguintes *controllers*: “*CourseController*”, “*WorkController*”, “*UserController*” e “*HomeController*”.

As classes de *model* são responsáveis por realizar os processamentos de leitura, inserção, atualização e exclusão do sistema. O sistema é composto pelas seguintes *models*: "*Invite*" que representa a entidade referente à tabela *Invites* e às regras de negócios relacionadas aos convites enviados por e-mail para fazer o upload de trabalhos na plataforma. "*Course*" representa a entidade referente à tabela Courses e às regras de negócio relacionadas aos cursos. "*User*" é a *model* responsável por abstrair as regras de negócio e a entidade referente aos usuários da parte administrativa do sistema. "*Works*" é a *model* responsável por abstrair as informações referentes os trabalhos que serão manipulados na plataforma.

Figura 48: Diagrama de classe do sistema



Fonte: Autores

1.24 **Diagrama entidade e relacionamento**

O Diagrama de entidades e relacionamentos (DER) tem por finalidade representar as tabelas do banco de dados e os seus relacionamentos entre si. O DER pode descrever altos níveis de abstração do sistema além de ser capaz de representar o modelo conceitual da estrutura de banco de dados. Esse modelo serve de grande auxílio para tarefas mais complexas na abstração do banco de dados, pois facilita muito o entendimento das tabelas para mais tarde poder se fazer o mapeamento objeto relacional (ORM).

Abaixo na Figura 49 procurou-se representar a estrutura de banco de dados desenvolvida. O sistema é composto de 4 tabelas principais, são elas: *users*, *invites*, *works*, e *courses*.

Para esta estrutura, descrevemos abaixo o uso de cada tabela:

- *Users*: Representa a tabela dos usuários do sistema, todos aqueles que tem acesso ao administrativo do mesmo. É composta pelos campos:
  - Id: identificador de cada registro;
  - *Name:* representa o nome completo do usuário;
  - E-mail: e-mail do usuário que acessa o sistema, é o campo necessário na autenticação do mesmo;
  - *Password*: senha do usuário, necessária no momento do *login*;
  - Role: campo utilizado para o controle de acessos dentro do sistema, este campo representa quais autorizações um determinado usuário poderá ter;
  - *Created_at*: campo para controle interno, identifica quando o registro foi criado;
  - *Updated_at*: campo para controle interno, identifica houve qualquer alteração no registro;

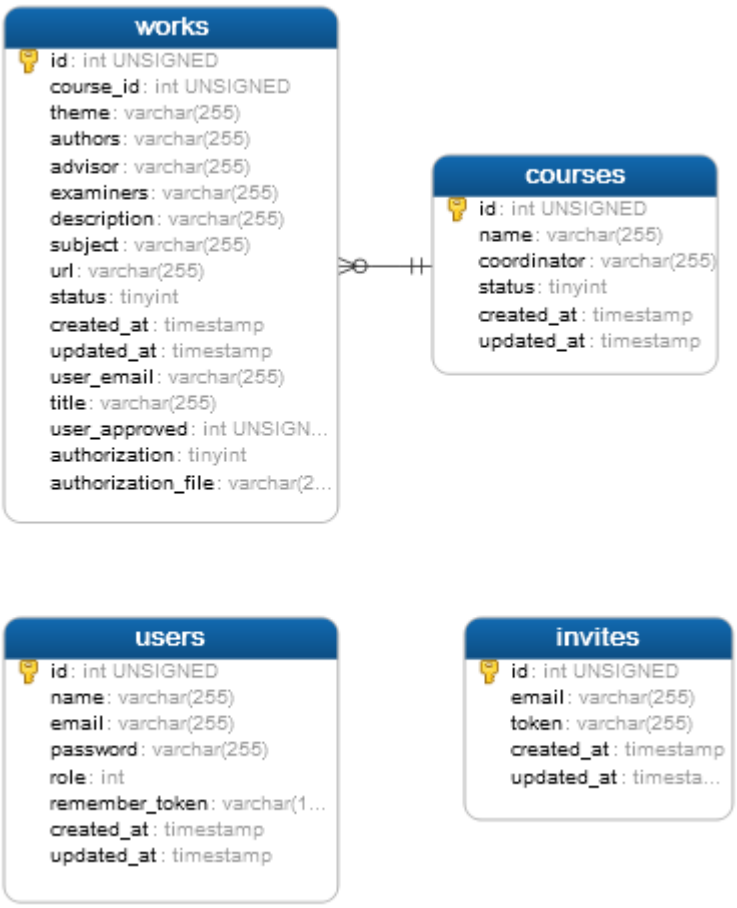
- *Invites:* Representa a tabela de convites, ou seja, quando se deseja que um determinado aluno faça o upload do seu trabalho é gerado um registro nessa tabela contendo as informações necessárias para que o mesmo tenha acesso à página de *upload*;
  - Id: identificador de cada registro;
  - E-mail: e-mail que foi enviado o *link* para fazer o *upload* do trabalho, este *link* possui um *token* único que é resgatado e validado no momento de acessar a página de *upload*;
  - *Token*: informação única com validade definida que é gerada no momento de se enviar o e-mail de convite ao usuário. Essa informação é resgatada e validada no momento que o usuário acessa o *link* que foi enviado em seu e-mail;
  - *Created_at*: campo para controle interno, identifica quando o registro foi criado;
  - *Updated_at:* campo para controle interno, identifica houve qualquer alteração no registro;

- *Courses:* Tabela que guarda as informações referentes aos cursos que existem na instituição, estes cursos podem ser adicionados, alterados ou inabilitados da instituição a qualquer momento.
  - Id: Identificador de cada registro;
  - *Name*: nome do curso cadastrado na instituição;
  - *Coordinator:* coordenador do curso;

  - *Status:* representa o status do curso, se o mesmo esta ativo ou não. Caso esteja ativo irá aparecer na página inicial do sistema e permitirá que sejam feitas buscas e filtragens de trabalhos relacionados a este curso;
  - *Created_at*: campo para controle interno, identifica quando o registro foi criado;
  - *Updated_at*: campo para controle interno, identifica houve qualquer alteração no registro;

- Works: Tabela que guarda as informações referentes aos TCC's que estão disponíveis no sistema.
  - Id: Identificador de cada registro;
  - *Course_id:* id do curso que pertence o trabalho;
  - *Theme:* tema do trabalho, este campo também é utilizado como indexador nas buscas;
  - *Authors*: nomes dos autores do trabalho;
  - *Advisor*: nome do orientador do trabalho;
  - *Examiners:* nome dos membros da banca avaliadora;
  - *Description:* descrição do trabalho;
  - *Subject*: assunto do trabalho;
  - *Url: link* que esta armazenado o trabalho;
  - *Status: status* do trabalho, se esta pendente (0), aprovado (1) ou reprovado (2);
  - *Created_at:* campo para controle interno, identifica quando o registro foi criado;
  - *Updated_at:* campo para controle interno, identifica houve qualquer alteração no registro;
  - *User*_email: e-mail dos membros da equipe;
  - *Title:* título do trabalho;
  - *User_approved*: usuário que aprovou ou reprovou um trabalho;
  - *Authorization*: campo que define se a equipe autoriza ou não o trabalho estar disponível para qualquer pessoa consultar no sistema;
  - *Authorization_file:* foto do termo de autorização assinado pelo(s) membro(s);

Figura 49: Diagrama entidade e relacionamento das tabelas do banco de dados do sistema



Fonte: Autores

CAPÍTULO III

Apresentação do sistema

Nesse capítulo se apresenta os processos e metodologias envolvidas para o desenvolvimento desse projeto.

3.1 Telas do sistema administrativo

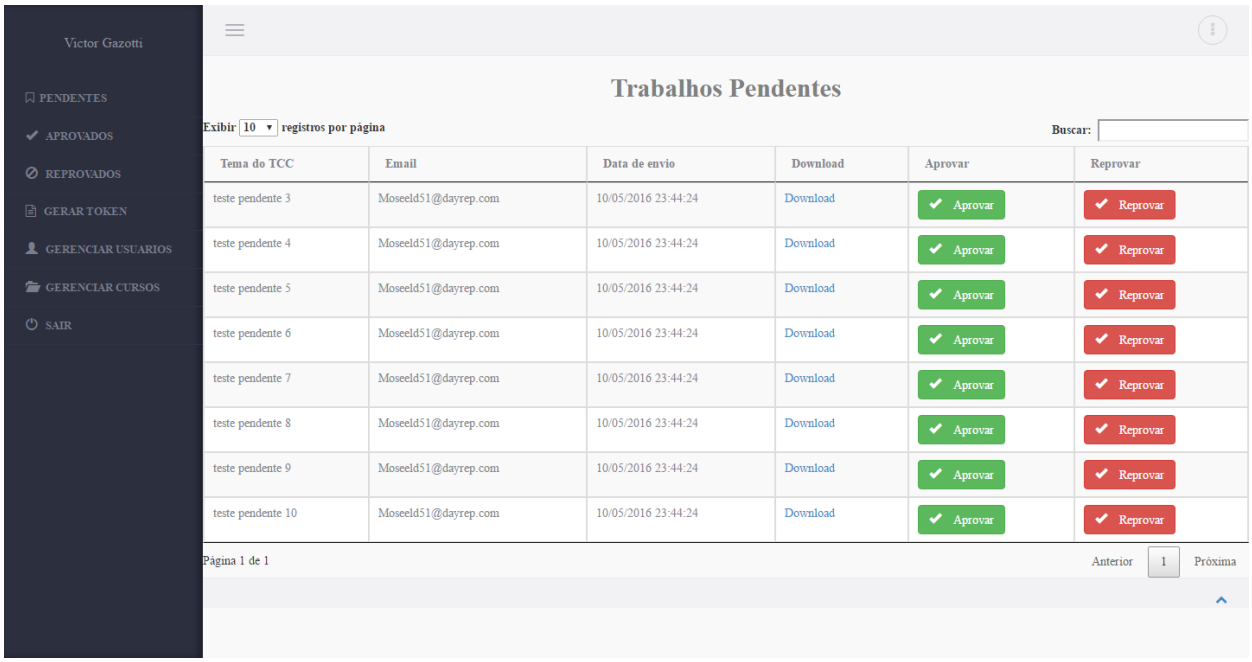
A interface do administrador é acessada por meio do navegador da web e também tem suporte integrado em *tablets*, iOS e *Android* com um design responsivo. O sistema inicialmente possui telas do painel administrativa sendo a primeira encontrada na Figura 50 onde a tela de *login* que possibilitará o de controle de acesso, onde o usuário irá realizar *login* preenchendo Usuário e Senha de acesso para a realização do acesso ao sistema na área administrativa. Para entrar no sistema, o usuário deverá apertar o botão “ENTRAR” que é apresentado abaixo do *login*.

Figura 50: Tela de *login*

Fonte: Autores

A Figura 51 apresenta a tela com os trabalhos que já foram cadastrados no sistema e estão aguardando aprovação, mostrando o tema do TCC, o e-mail do usuário, a data correspondente do envio, e o *link* para *download.* São definidos como pendentes os trabalhos que ainda não foram classificados como aprovados ou reprovados. O usuário/administrador pode aprovar um TCC apertando o botão “APROVAR”, e recusar um TCC apertando o botão “RECUSAR”. Após o TCC ser aprovado, ele irá aparecer na página de trabalhos aprovados e será disponibilizado na página principal, se o aluno tiver autorizado.

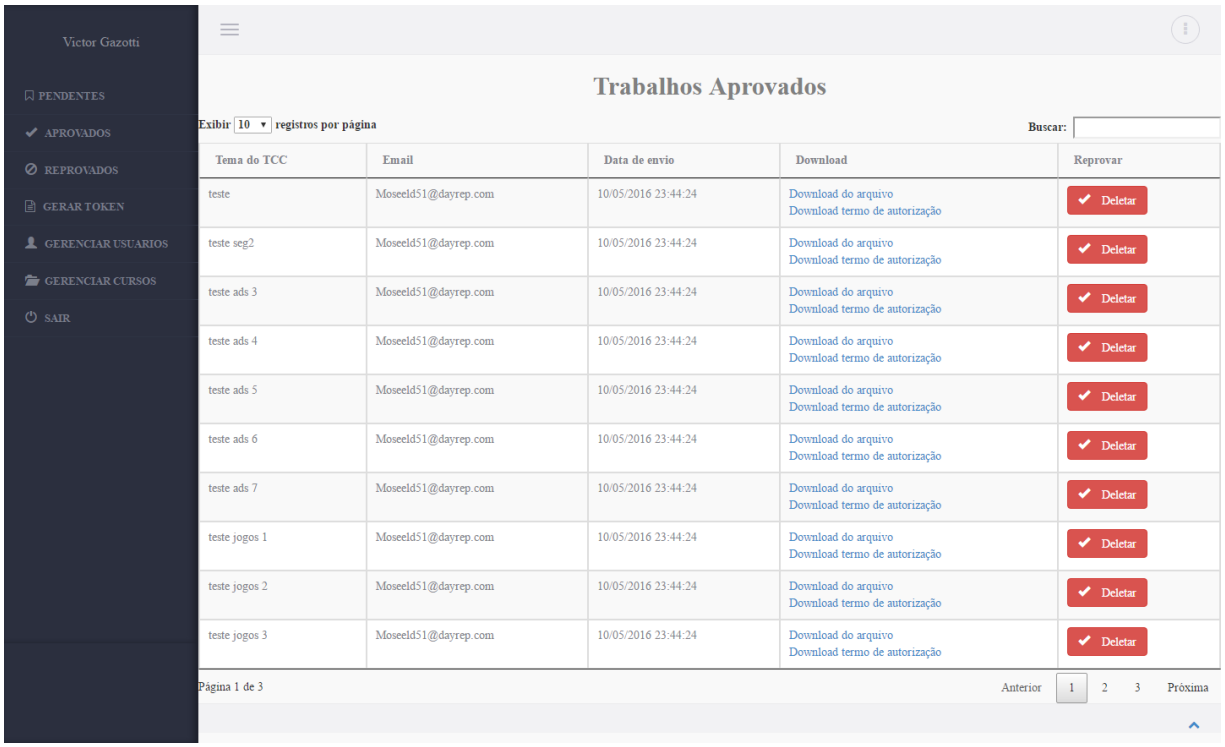
Figura 51: Tela dos trabalhos pendentes



Fonte: Autores

A Figura 52 apresenta a tela dos trabalhos que já passaram pela aprovação dos administradores e estão aguardando a publicação. Contem as características do documento como tema do TCC, e-mail do usuário, a data de envio e o campo do *download* para armazenar os arquivos.

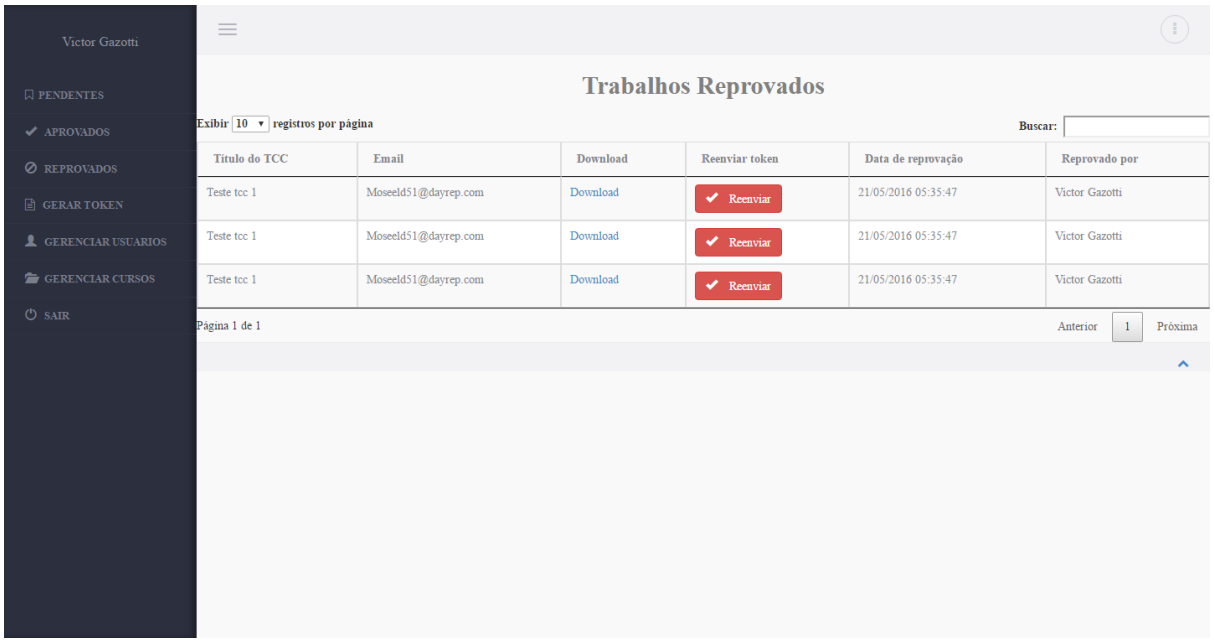
Figura 52: Tela dos trabalhos aprovados



Fonte: Autores

A tela apresentada na Figura 53 mostra os trabalhos classificados como reprovados, contendo o tema do TCC reprovado, o e-mail do autor, data de envio e link para *download*. Para o administrador enviar uma mensagem de reprovação ao autor.

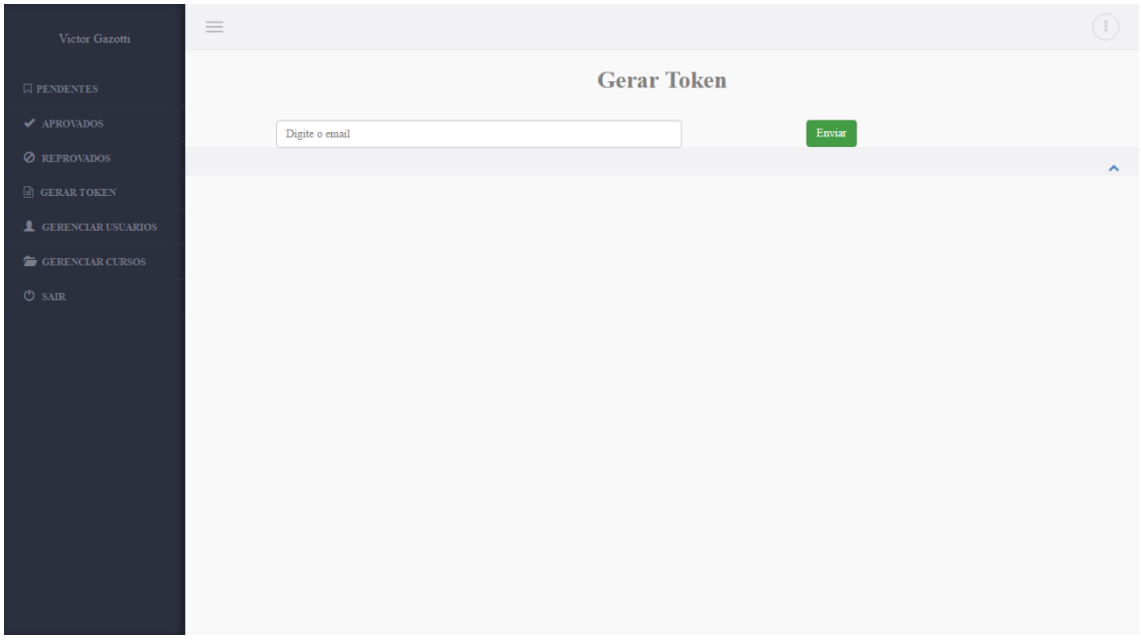
Figura 53: Tela dos trabalhos reprovados



Fonte: Autores

A Figura 54 apresenta o local para gerar o *token*, onde o administrador irá informar o e-mail no campo “DIGITE O EMAIL” e apertará o botão “ENVIAR”, onde automaticamente será enviado um e-mail para o usuário com um link de validade especifica, que redirecionará o usuário a tela de upload de TCC’s.

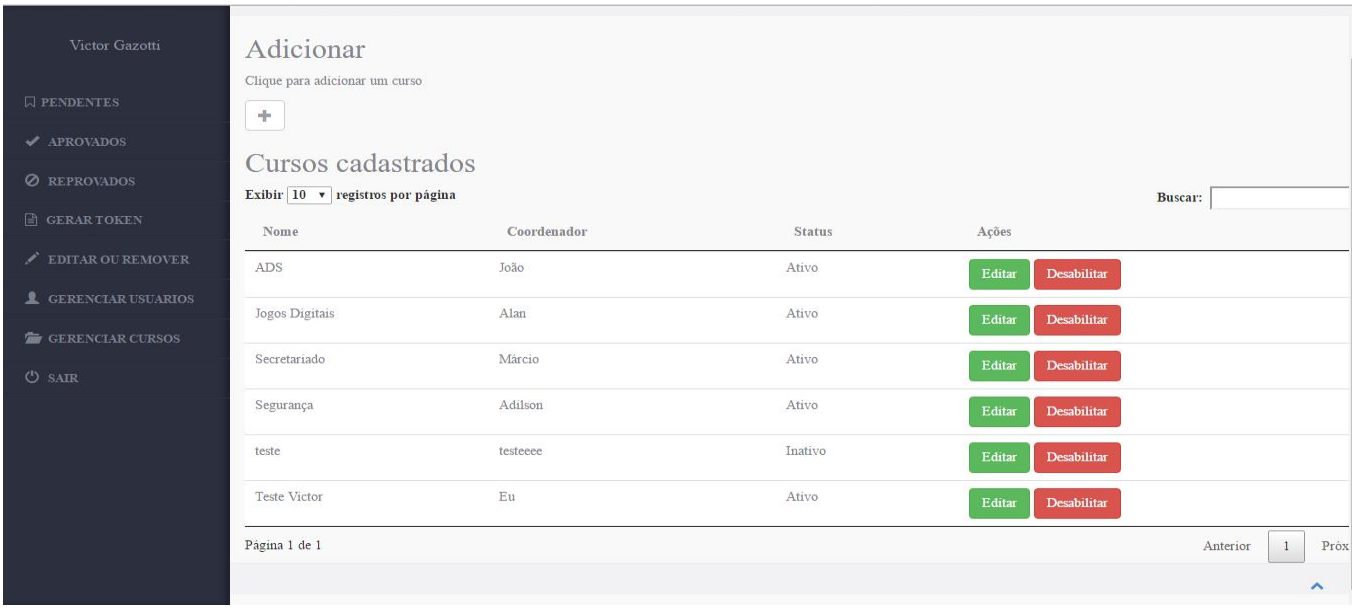
Figura 54: Tela de geral *Token*



Fonte: Autores

Na tela abaixo representada pela Figura 55, há uma lista dos cursos cadastrados, informando o nome do curso, o coordenador e o status. Possui um botão “+” que redirecionará o administrador a tela cadastrar curso, tendo também as opções de botões “EDITAR” que redirecionar o administrador para tela editar curso, e “DESABILITAR” que redirecionara o usuário para tela desabilitar curso.

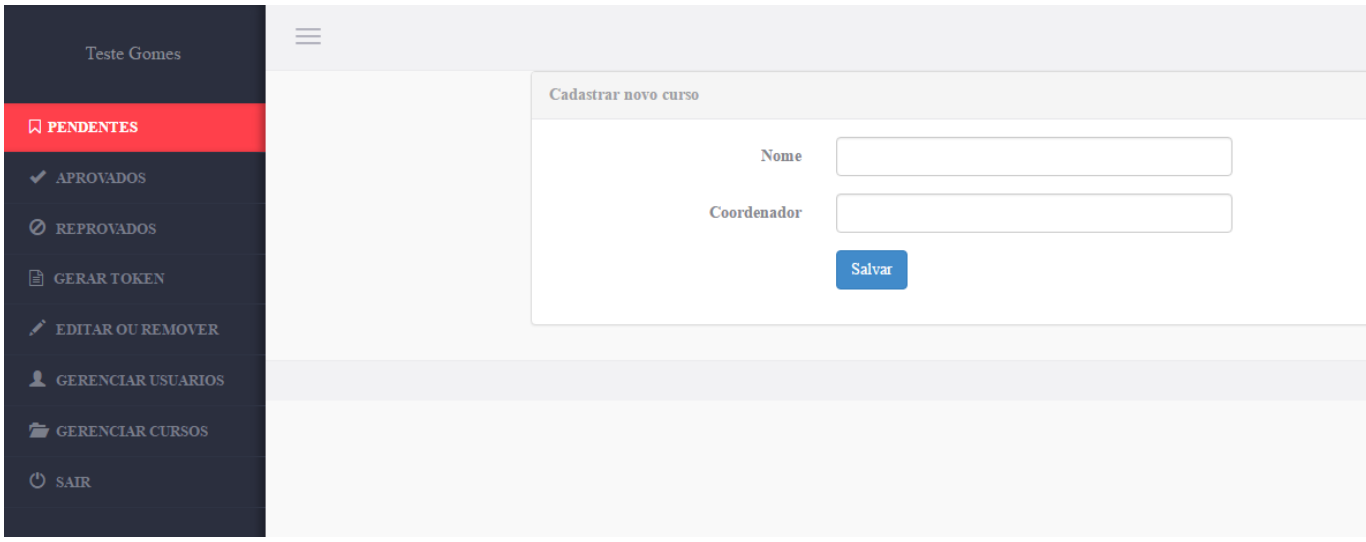
Figura 55: Listar cursos cadastrados



Fonte: Autores

A Figura 56 a seguir, apresenta a tela de cadastrar novo curso, onde o administrador poderá inserir um novo curso, preenchendo o campo “NOME”, com o nome do novo curso, e o campo “CORDENADOR”, com o nome do coordenador do novo curso, e por fim apertando o botão “SALVAR”.

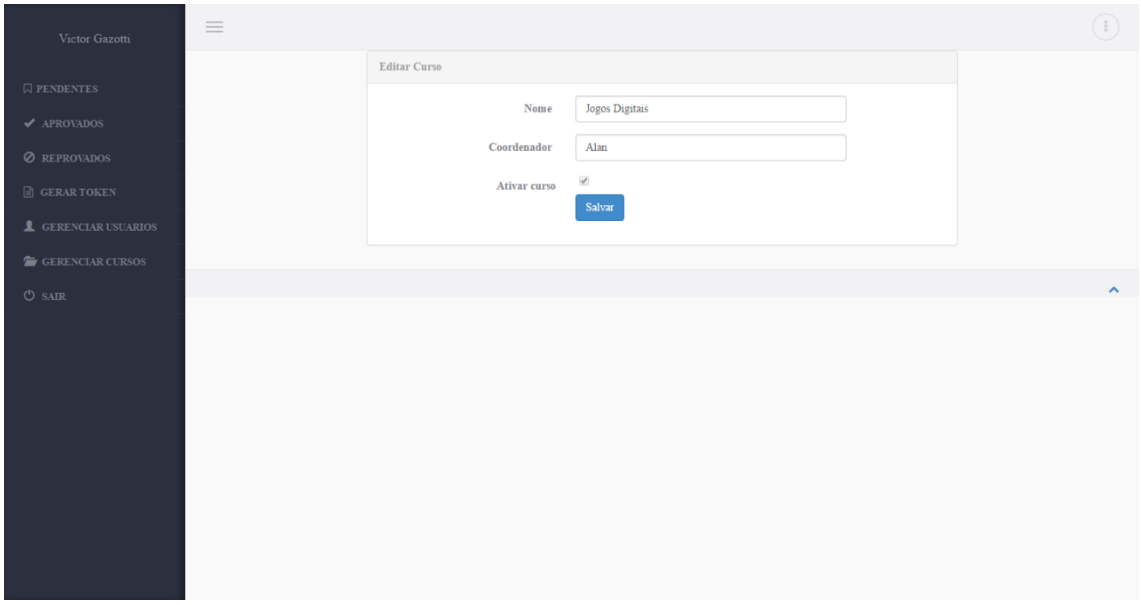
Figura 56: Tela cadastrar curso



Fonte: Autores

Na tela de editar curso, apresentada pela Figura 57, permitira que o administrador edite o nome do curso no campo “NOME”, o coordenador no campo “COORDENADOR”, tendo a opção de ativar ou desativar o curso selecionando *check box*, e concluindo a ação apertando o botão “SALVAR”.

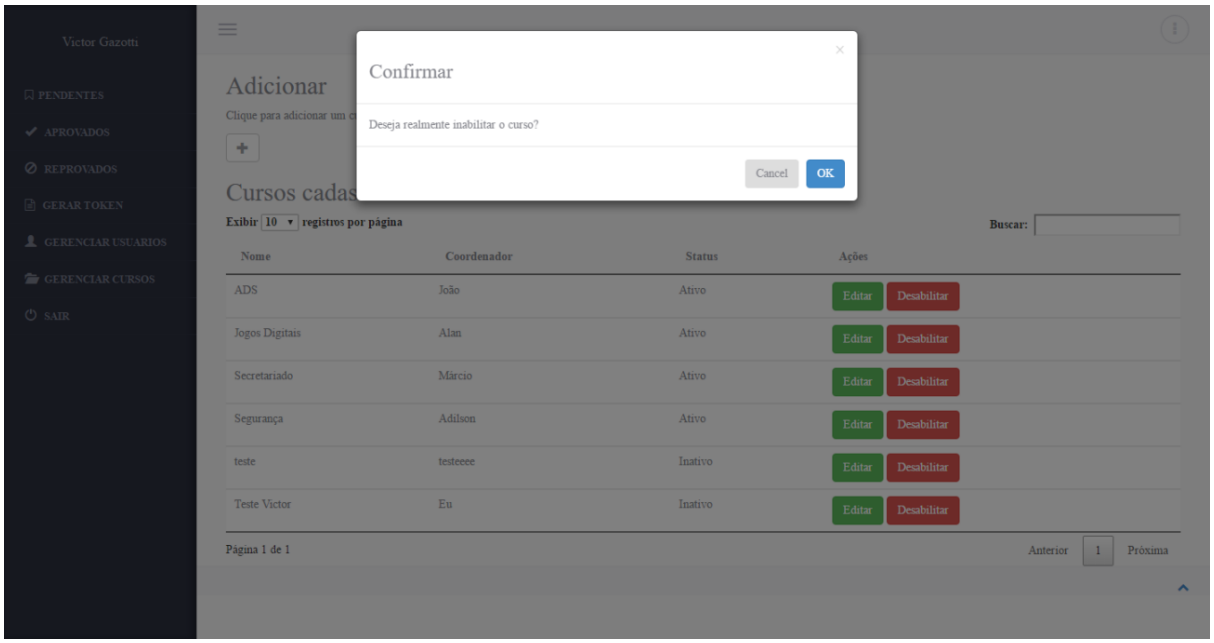
Figura 57: Tela editar curso



Fonte: Autores

A Figura 58 mostra a tela desabilitar, quando pressionado o botão “DESABILITAR”, abrirá uma tela de confirmação onde o usuário poderá apertar o botão “CANCELAR” caso deseje voltar a tela anterior sem fazer alterações, e o botão “OK”, se deseje continuar com a ação de excluir.

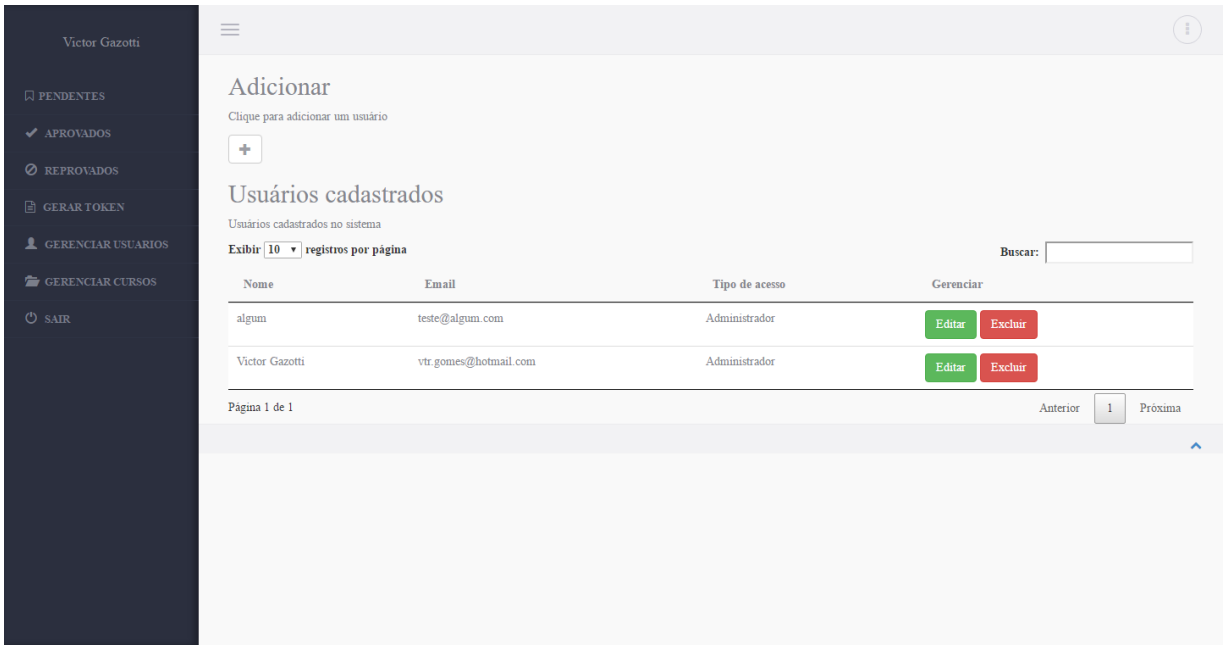
Figura 58: Tela DESABILITAR curso



Fonte: Autores

A Figura 59, mostra a tela gerenciar usuários, onde há um botão “+” que permite ao administrador adicionar um novo usuário, a figura 59 mostra também uma lista de todos os usuários cadastrados, contendo nome, e-mail, e tipo de acesso (administrador ou usuário). Nesta tela o administrador tem 2 opções de gerenciamento, o botão “EDITAR” e o botão “EXCLUIR”.

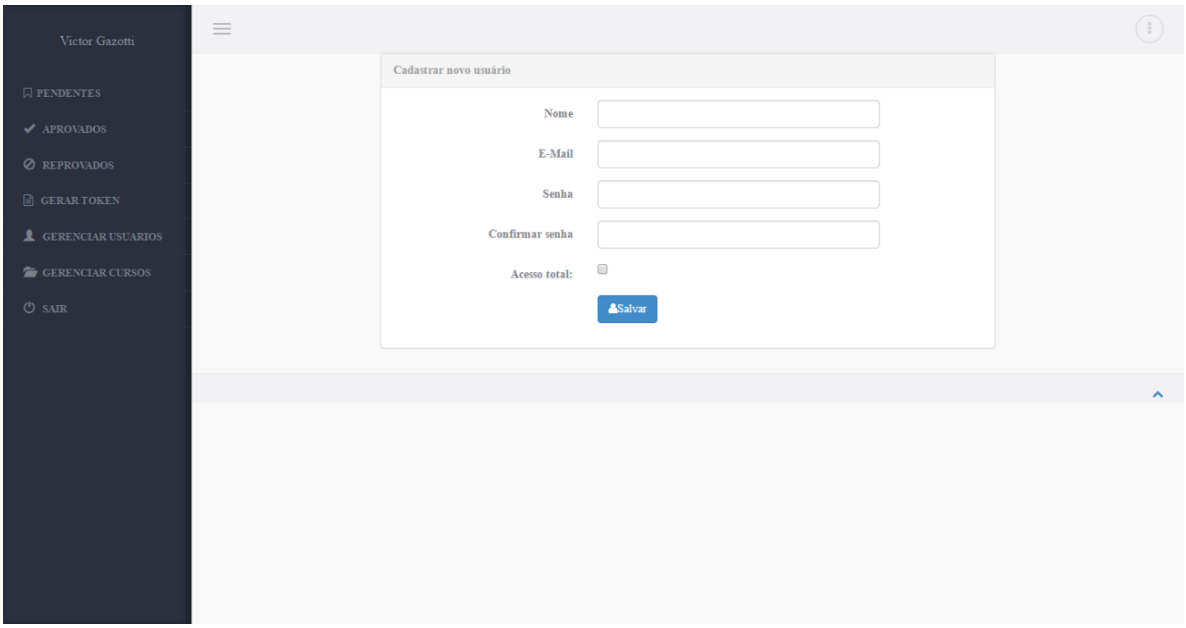
Figura 59: Tela gerenciar usuários



Fonte: Autores

Apertando o botão “+”, da fFigura 59, o sistema abrirá a tela de cadastrar novo usuário apresentada na Figura 60,onde o administrador devera informar o nome no campo “NOME”, o e-mail no campo “E-MAIL”, uma senha no campo “SENHA”, a confirmação da senha no campo “CONFIRMAR SENHA” e definindo o acesso do usuário ativando ou não a *check box*, apertando por ultimo o botão “SALVAR”.

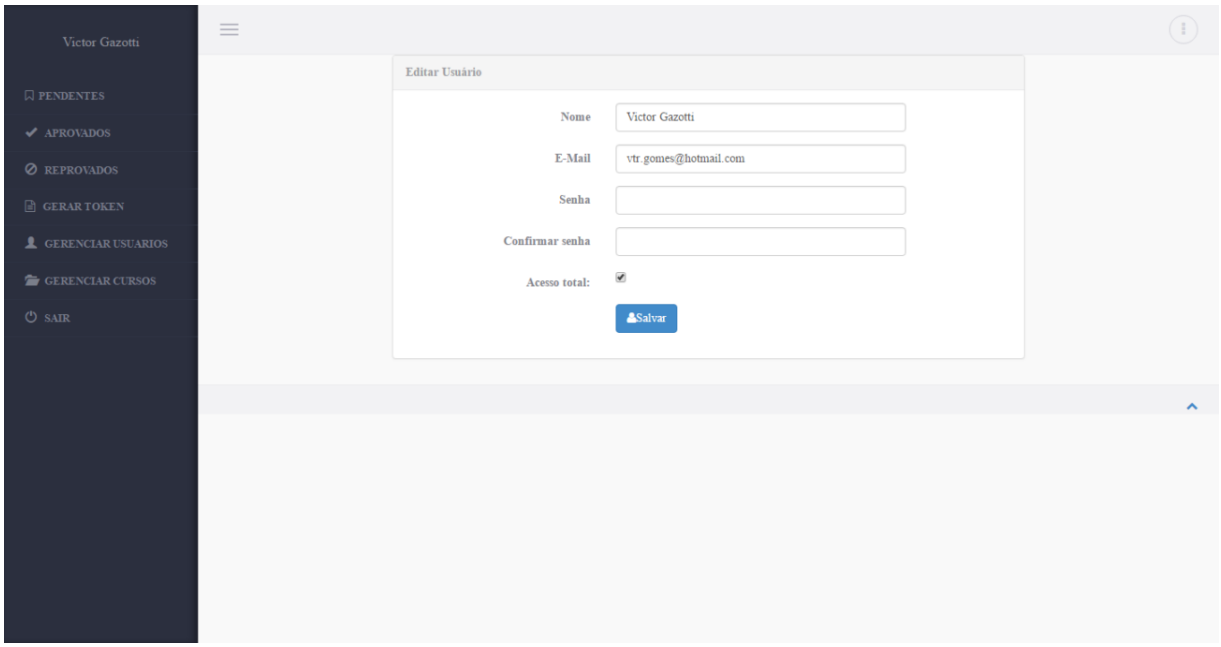
Figura 60: Cadastrar novo usuário



Fonte: Autores

Apertando a opção “EDITAR”, na tela de gerenciamento de usuáros na figura 59, o usuário será redirecionado para a tela editar usuário na Figura 61, que apresentará as opções cadastrais podendo ser analsadas ou atualizadas.

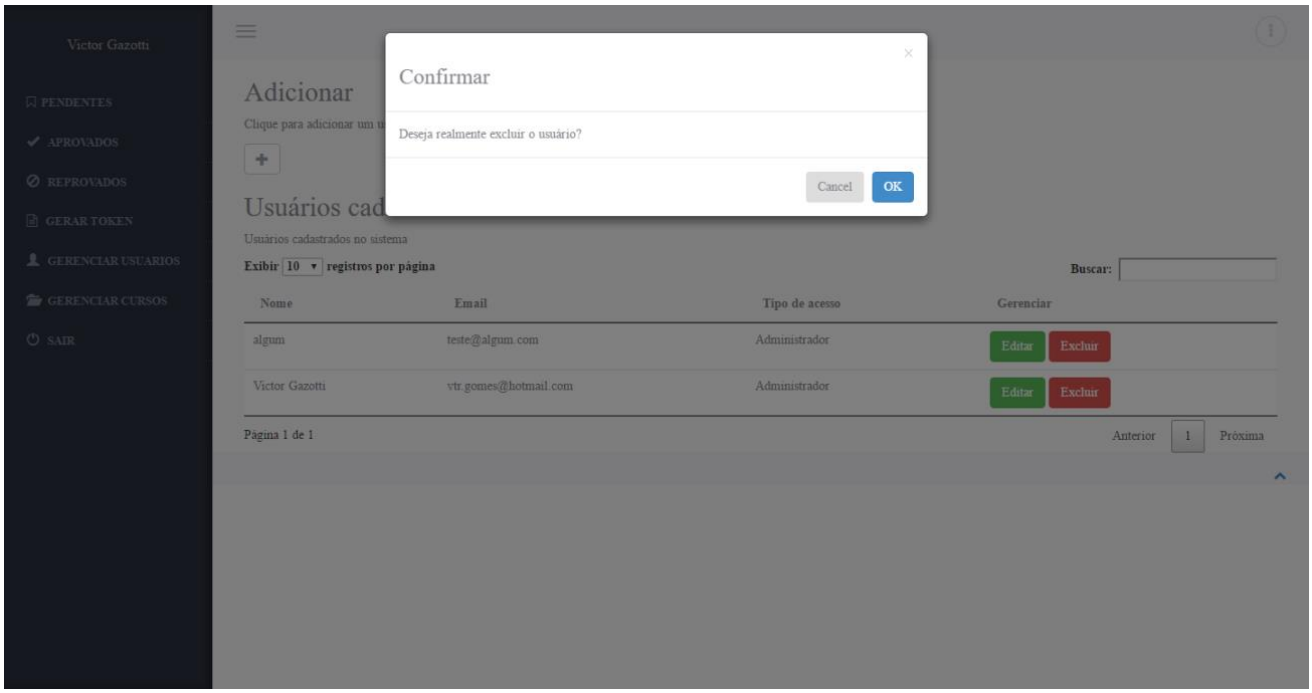
Figura 61: Tela editar usuário



Fonte: Autores

Apertando a opção “EXCLUIR”, na tela de gerenciamento de usuáros na figura 59, o usuário será redirecionado para a tela de excluir usuário na Figura 62, no qual deve apertar o botão “CANCELAR” caso deseje voltar a tela anterior ou o botã “OK” caso deseje prosseguir com a exclusão.

Figura 62: Tela de excluir usuário

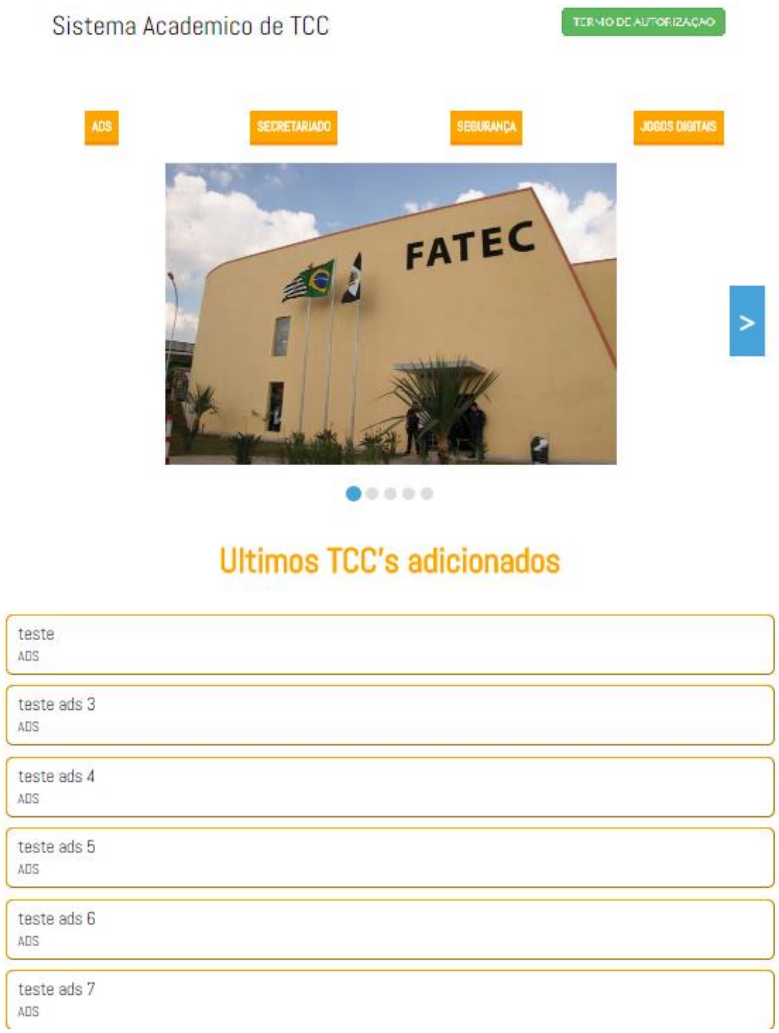


Fonte: Autores

3.2 Telas do painel de usuários

A seguir estão encontradas as telas do painel do usuário inicialmente na Figura 63 onde é apresentado os ultimos TCC's adicionados em lista mostrando o tema do TCC e o curso que ele foi apresentado. Quando o usuário clicar no tema do tcc ele será direcionado para a proxima página na Figura 64. O usuário poderá tamblem clicar nos botões "A.D.S", "JOGOS", "SECRETÁRIADO" e"SEGURANÇA", no qual redicionará para a página do curso.

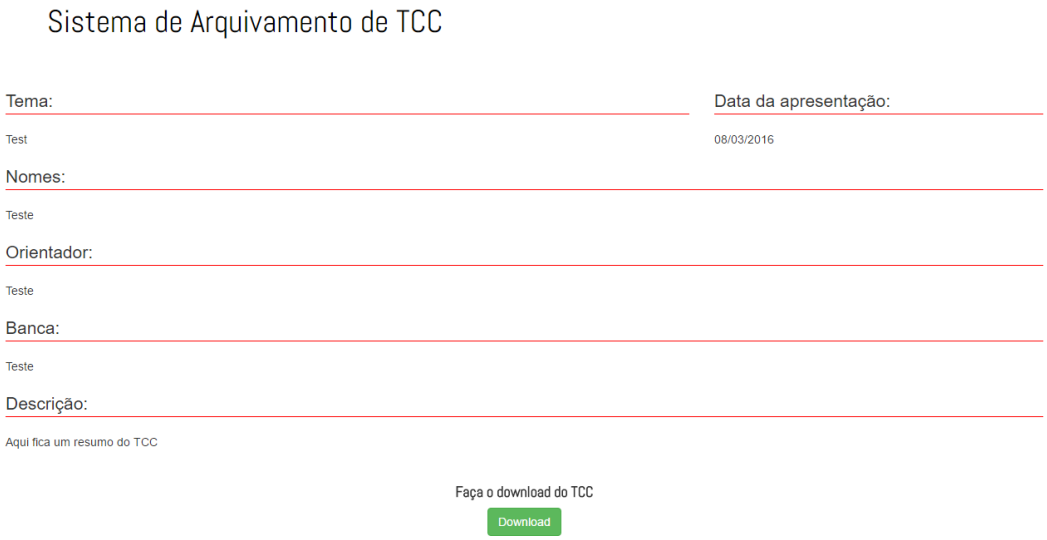
Figura 63: Tela *index*



Fonte: Autores

Nesta tela, Figura 64 o usuário poderá ver a descrição sobre o tema de TCC escolhido, podendo ver o nome dos autores, o orientador, a banca avaliatória, e a descrição do trabalho. Clicando no botão “ *DOWNLOAD*” o usuário poderá fazer o *download* do TCC selecionado.

Figura 64: Tela da descrição do tema



Fonte: Autores

Quando o usuário clica nos botões “A.D.S”, ”JOGOS”, “SECRETÁRIADO” e “SEGURANÇA” na tela *index*, Figura 63, ele é redirecionado para as telas abaixo nas figuras Figura 65, Figura 6666, 67 e 68.

Figura 65: Tela dos temas do curso de Segurança da Informação

Fonte: Autores

Figura 66: Tela dos temas do curso de Jogos

Fonte: Autores

Figura 67: Tela dos temas do curso de Análise e desenvolvimento de sistemas

Fonte: Autores

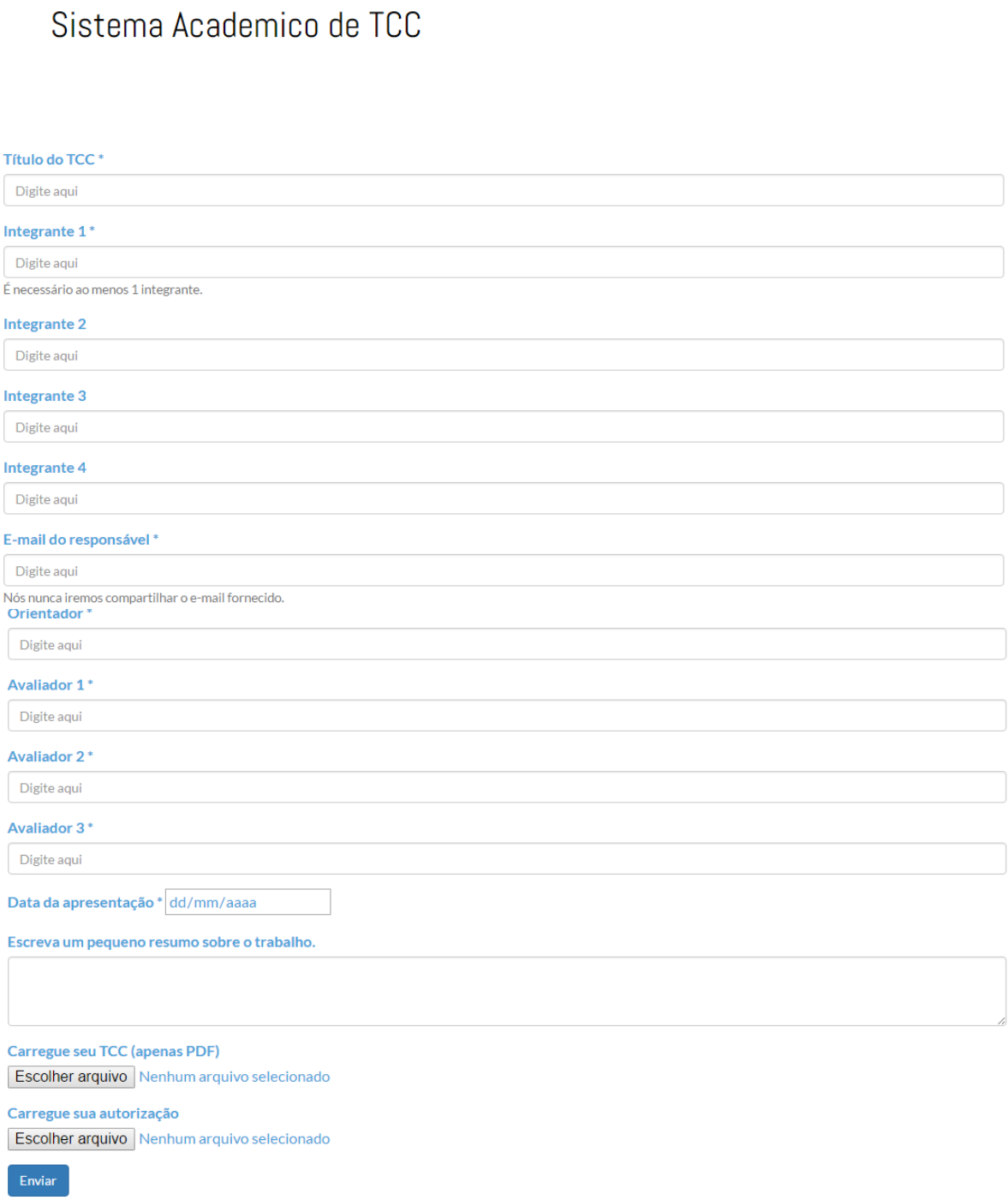
Figura 68: Tela dos temas do curso de secretariado

Fonte: Autores

A

Figura 69 apresenta a tela de formulário para o usuário que é convidado através de um link. O usuário deverá preencher os campos com o titulo do TCC, o nome dos integrantes, e-mail, o nome do orientador, os avaliadores da banca, a data de apresentação e o resumo do trabalho. E Através do botão “ESCOLHER ARQUIVOS”, o usuário poderá fazer o *upload* do seu trabalho no formato .pdf, e clicando no botão “ENVIAR” para enviar os dados para o administrador.

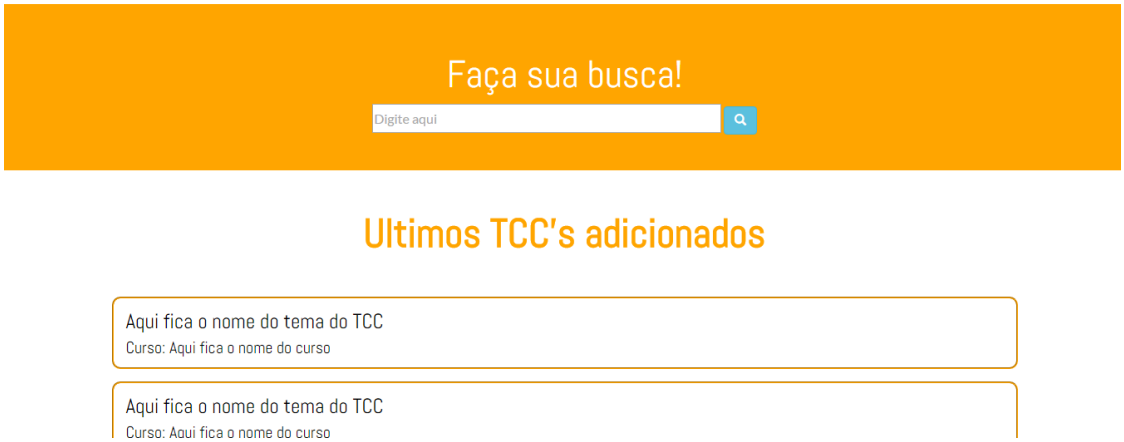
Figura 69: formulário de *upload* de TCC



Fonte: Autores

A Figura 70 representa a página de busca de trabalhos. Os usuários poderão realizar buscas por titulo ou palavras chaves relacionadas ao título. Após clicar no ícone de busca os trabalhos encontrados serão listados e o usuário poderá visualizar os resultados de sua pesquisa.

Figura 70: Tela de busca dos trabalhos



Fonte: Autores

O termo de autorização contido na Figura 71 tem como finalidade obter a ciência do aluno de como sua monografia será utilizada no sistema. O aluno conseguirá obter o termo através do sistema na página inicial, clicando no botão de termo de autorização. O aluno deverá imprimir, preencher a mão, escanear e fazer o *upload* do termo juntamente com a monografia no momento que fizer o *upload* para o sistema.

Figura 71: Termo de autorização disponível para download do sistema

TERMO DE AUTORIZAÇÃO DE PUBLICAÇÃO DE MONOGRAFIA

Neste ato, _____, nacionalidade _____, estado civil _____, portador da Cédula de identidade RG nº._____ , inscrito no CPF/MF sob nº _____, residente à Av/Rua _____ , nº. _____, município de _____/São Paulo. AUTORIZO a publicação da minha monografia _____ no sistema de arquivamento virtual de TCCs da FATEC – SÃO CAETANO DO SUL – ANTONIO RUSSO, para consultas e *download* para qualquer pessoa que tenha acesso ao sistema.

_____, _____ de _____ de _____

_____

Assinatura do aluno

Fonte: Autores

CONSIDERAÇÕES FINAIS

A concepção dessa monografia teve como ponto central a elaboração de um sistema web para a consulta de TCC's da instituição Fatec São Caetano do Sul- Antonio Russo com a intensão de auxiliar as pesquisas de trabalhos de conclusão de curso e facilitar o acesso de informações a alunos, docentes, e a comunidade acadêmica em geral.

Baseado nas revisões bibliográficas realizadas nessa pesquisa, verificou-se que o sistema web desenvolvido utiliza diversas tecnologias como a linguagem PHP que ajudou na elaboração da lógica de funcionamento e aplicação das regras de negócios envolvidas no sistema.

A aplicação de um framework de larga utilização no mercado com ORM integrado que permitiu uma abstração simplificada das entidades e das logicas do banco de dados.

A metodologia UML permitiu representar as ações dos usuários, o comportamento do sistema, suas funcionalidades, as suas regras de negócios, e as suas classes. Criando assim uma documentação baseada nos princípios da engenharia de software envolvidos nas etapas do desenvolvimento do sistema.

O sistema desenvolvido permite que não somente a comunidade acadêmica da FATEC São Caetano do Sul- Antonio Russo tenha acesso aos trabalhos desenvolvidos na instituição, mas também está disponível para toda a sociedade, aumentando a acessibilidade entre as informações tecnológicas do âmbito acadêmico e a sociedade.

A conclusão do sistema permitiu uma simplificação das consultas dos trabalhos desenvolvidos pelos alunos da instituição com a melhora do processo anteriormente adotado. A melhora do processo de consulta demostra que os conceitos de analise de necessidades foram bem compreendidos e aplicados através de um sistema que automatizasse, simplificasse e solucionasse uma demanda da instituição.

Referências

ALECRIM, Emerson. **Conhecendo o Servidor Apache (HTTP Server Project).** Disponível em: <http://www.infowester.com/servapach.php>. Acesso em: 05 de maio de 2016.

AMARAL, Rodrigo. **10 motivos para usar o Sublime Text**. Sublime text dicas. Disponível em: <http://sublimetextdicas.com.br/10-motivos-para-usar-o-sublime-text/>. Acesso em: 04 de maio de 2016.

APACHE SOFTWARE FOUNDATION. **The apache software foundation.** Disponível em: <http://www.apache.org/>. Acesso em: 05 de maio de 2016.

BOLINGER, D., BRONSON, T. **Applying RCS and SCCS**: From Source Control to Project Control, 526p, Pub. O'Reilly Media, September, 1995.

CANALTECH. **O que é servidor apache**. Disponível em: <http://canaltech.com.br/o-que-e/internet/o-que-e-servidor-apache/>. Acesso em: 04 de maio de 2016.

CANALTECH. **O que é e como funciona a linguagem javascript?.** Disponível em: <http://canaltech.com.br/materia/internet/o-que-e-e-como-funciona-a-linguagem-javascript/>. Acesso em: 04 de maio de 2016.

CARVALHO, L., S; LUCAS, ELAINE.R.O. **Serviço de referência e informação: do tradicional ao on-line.** Universidade do Estado de Santa Catarina. Florianópolis, 2004. Disponível em: < http://www.cinform-anteriores.ufba.br/vi_anais/docs/LidianeElaineServicoReferencia.pdf>. Acesso em: 10 de março 2016.

COSTA, Gabriel. **O QUE É BOOTSTRAP?.** Tutorial web designer. Disponível em: <http://www.tutorialwebdesign.com.br/o-que-e-bootstrap/>. Acesso em: 04 de maio de 2016.

COMPUTAÇÃO UFCG. **Unified modeling language (UML**). Disponível em: <http://www.computacao.ufcg.edu.br/>. Acesso em: 16 maio 2016.

CUENCA, A. M. B. et al. Biblioteca virtual e o acesso às informações científicas e acadêmicas. **Revista USP**, São Paulo, dez./fev. 2009. Disponível em: <http://www.revistas.usp.br/revusp/article/viewFile/13717/15535>. Acesso em: 10 de maio de 2016.

DALL'OGLIO, Pablo. **PHP: Programando com Orientação a Objetos.** São Paulo: Novatec Editora, 2007.

DEBONI, José Eduardo Zindel. **Modelagem orientada a objetos com a UML**. São Paulo: Futura, 2003.

DEVMEDIA. **Sistemas de controle de versão**. Disponível em: <http://www.devmedia.com.br/sistemas-de-controle-de-versao/24574>. Acesso em: 25 de abril de 2016.

DEVMEDIA. **Os 4 pilares da programação orientada a objetos**. Disponível em: <http://www.devmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264>. Acesso em: 01 de maio de 2016.

FAYAD, M.; SCMIDT, D.; JOHNSON, R. ***Building Applications Frameworks*.** John Willey, 1999.

GALBIATTI, Tiago. **MySQL Workbench**. Disponível em: <http://www.thiagovespa.com.br/blog/2010/09/18/mysql-workbench/>. Acesso em: 05 de maio de 2016.

JOHNSON, Ralph E. *Frameworks = (Components + Patterns). Communications of the ACM*, 1997.

LOPES, Sérgio. **Flexibilidade em páginas para dispositivos móveis com media queries**. Disponível em: <http://blog.caelum.com.br/flexibilidade-em-paginas-para-dispositivos-moveis-com-media-queries/>. Acesso em: 04 de maio de 2016.

LOPES, Camilo. **Modelagem de dados WorkBench MySQL**. Disponível em: <http://imasters.com.br/artigo/20668/mysql/modelagem-de-dados-workbench-mysql?trace=1519021197&source=single>. Acesso em: 05 de maio de 2016.

MARCHIORI, Patricia Zeni. "**Ciberteca" ou biblioteca virtual:** uma perspectiva de gerenciamento de recursos de informação. Online. Ciência da Informação, v.26, n.2.1997. Disponível em: <http://www.ibict.br/cionline/artigos/2629701.htm>. Acesso em: 21 março de 2016.

MILANI, André. **Construindo aplicações web com php e mysql.**: Novatec, 2010.

NASCIMENTO, Thiago. **Desenvolvendo com Bootstrap 3**: um framework front-end que vale a pena!. Disponível em: <http://thiagonasc.com/desenvolvimento-web/desenvolvendo-com-bootstrap-3-um-framework-front-end-que-vale-a-pena>. Acesso em: 04 de maio de 2016.

OFICINA DA NET. **Vantagens e desvantagens no uso de frameworks.** Disponível em: <https://www.oficinadanet.com.br/artigo/682/vantagens_e_desvantagens_no_uso_de_frameworks>. Acesso em: 11 de maio de 2016.

PEREIRA, M.H.R. **AngularJS. Uma abordagem prática e objetiva.** 208f.1 ed. São Paulo: Novatec Editora Ltda, 2014

PRINCIWEB. **Veja o que é ORM e os frameworks disponíveis para .net.** Disponível em: <http://www.princiweb.com.br/blog/programacao/aspnet/veja-o-que-e-orm-e-os-frameworks-disponiveis-para-net.html>. Acesso em: 30 de abril de 2016.

RAMOS, ALLAN. MVC – Afinal, é o quê ?. Disponível em: <http://tableless.com.br/mvc-afinal-e-o-que/>. Acesso em: 09 de maio de 2016.

REZENDE, Ricardo. **Conceitos Fundamentais de Banco de Dados**. Disponível em: <http://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649>. Acesso em: 05 de maio de 2016.

RIBEIRO, Leandro. **O que é UML e Diagramas de Caso de Uso**: Introdução Prática à UML. Disponível em: <http://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408>. Acesso em: 10 de maio de 2016.

ROWLEY, J. A **Biblioteca Eletrônica**. Trad. Antonio A. Briquet de Lemos. Brasília, Briquet de Lemos, 2002.

SÊMOLA, Marcos. **Gestão da segurança da informação:** Uma visão executiva. 17 ed. Rio de Janeiro: CAMPUS, 2003.

SPAIN, P. **Document Version Control with CVS.** Disponível em http://www.xpro.com.au/Presentations/UsingCVS/Document%20Version%20Control%20with%20CVS.htm. Acesso em 01 maio de 2016.

TABLELESS. **Git**. Disponível em: <http://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/>. Acesso em: 18 de abril de 2016.

TARGETTRUST. **O que é uml?.** Disponível em: <http://www.targettrust.com.br/blog/desenvolvimento/o-que-e-uml/>. Acesso em: 10 de maio de 2016.

W3C. **W3C**. Disponível em: <https://www.w3.org/>. Acesso em: 03 de maio de 2016.

ZEMEL, Tárcio. **Web design responsivo**: Páginas adaptáveis para todos os dispositivos. 1 ed. São Paulo: Casa do código, 2012.