

UNIVERSIDADE ESTADUAL DO CEARÁ

WESLEY JEFFERSON OLIVEIRA ALCOFORADO

**TCC-MANAGER - UM GERENCIADOR DE TRABALHOS
DE CONCLUSÃO DE CURSO PARA O CURSO DE
CIÊNCIAS DE COMPUTAÇÃO DA UECE**

FORTALEZA, CEARÁ

2011

WESLEY JEFFERSON OLIVEIRA ALCOFORADO

**TCC-MANAGER - UM GERENCIADOR DE TRABALHOS DE
CONCLUSÃO DE CURSO PARA O CURSO DE CIÊNCIAS DE
COMPUTAÇÃO DA UECE**

Monografia submetida à coordenação do curso de Ciências da Computação da Universidade Estadual do Ceará, no ano de 2011, como requisito parcial para obtenção do grau de Bacharel em Ciências da Computação.

Orientador: Mariela Cortés

FORTALEZA, CEARÁ

2011

A000z

TCC-Manager - um gerenciador de trabalhos de conclusão de curso para o curso de Ciências de Computação da UECE / Wesley Jefferson Oliveira Alcoforado. – Fortaleza, 2011.

67p.;il.

Orientador: Prof^a. Dr^a. Mariela Cortés

Monografia (Graduação em Ciências da Computação) -
Universidade Estadual do Ceará, Centro de Ciências e Tec-
nologia.

1. Gerenciador 2. Processo 3. Projeto final I. Universidade Estadual do Ceará, Centro de Ciências e Tecnologia.

CDD:000.0

WESLEY JEFFERSON OLIVEIRA ALCOFORADO

**TCC-MANAGER - UM GERENCIADOR DE TRABALHOS DE CONCLUSÃO DE
CURSO PARA O CURSO DE CIÊNCIAS DE COMPUTAÇÃO DA UECE**

Monografia submetida à coordenação do curso de Ciências da Computação da Universidade Estadual do Ceará, no ano de 2011, como requisito parcial para obtenção do grau de Bacharel em Ciências da Computação.

Aprovada em: __/__/----

BANCA EXAMINADORA

Prof^a. Dr^a. Mariela Cortés
Universidade Estadual do Ceará - UECE
Orientador

Prof. Dr. Numero Um
Universidade Estadual do Ceará - UECE
Co-orientador

Prof. Me. Numero Dois
Universidade Estadual do Ceará - UECE
Co-orientador

Prof. Dr. Numero Tres
Universidade Estadual do Ceará - UECE

AGRADECIMENTOS

À minha esposa Raquel por ter me incentivado a escrever minha monografia, me apoiando nos momentos em que fraquejei.

Aos amigos de graduação mais próximos pelo apoio mútuo durante toda a jornada na universidade.

Aos professores por terem compartilhado seus conhecimentos conosco.

*“L’ordinateur obéit à vos ordres, pas
à vos intentions.”*

Anônimo

SUMÁRIO

1	Introdução	15
1.1	Motivação	15
1.2	Objetivo	16
1.2.1	Objetivos específicos	16
1.3	Metodologia	16
1.4	Organização do trabalho	17
2	Regulamento de Projeto Final	18
2.1	Entidades envolvidas e suas atribuições	18
2.2	Desenvolvimento do projeto	19
2.2.1	Proposta do projeto	19
2.2.2	Defesa do projeto	20
2.2.3	Resumo do fluxo do desenvolvimento do projeto final	20
3	Processo de desenvolvimento da aplicação	22
3.1	Escolha das tecnologias	22
3.1.1	Linguagem de programação	22
3.1.2	Framework	23
3.1.2.1	MVC	23
3.1.2.2	ORM	24
3.1.2.3	<i>Convention over configuration</i>	24
3.1.3	SGBD	25
3.1.4	IDE	26
3.1.5	Outros requisitos de plataforma	26
3.2	Casos de uso e especificação de requisitos do sistema	27
3.2.1	Diagrama de classes	27

3.2.1.1	Autenticação de usuários	30
3.2.1.2	Formulários	30
3.2.1.3	Casos de uso	31
4	Utilização do sistema	32
4.1	Utilização do sistema	32
4.1.1	Telas em comum entre todos os perfis	32
4.1.2	Perfil de administrador	34
4.1.3	Perfil de estudante	37
4.1.4	Perfil de professor (orientador)	39
4.1.5	Perfil de comissão	45
5	Conclusões e trabalhos futuros	52
5.1	Trabalhos futuros	52
	Referências Bibliográficas	53
6	Apêndice	54
6.1	Casos de uso	55
6.1.1	Manter Professores	55
6.1.2	Manter Estudantes	57
6.1.3	Manter Projetos	59
6.1.4	Manter Propostas	61
6.1.5	Manter Defesas	63
7	Anexos	65
7.1	Formulário de proposta de projeto final	66
7.2	Formulário de solicitação de defesa e banca	67

LISTA DE FIGURAS

Figura 2.1	Fluxograma do desenvolvimento do projeto final	21
Figura 3.1	Arquitetura MVC	24
Figura 3.2	Definição de um mapeamento no Hibernate	25
Figura 3.3	Tabela Users no banco de dados	25
Figura 3.4	Descrição de uma tabela no Symfony	27
Figura 3.5	Diagrama de classes base	28
Figura 3.6	Diagrama de classes extendidas	28
Figura 3.7	Diagrama de classes da camada de controle	29
Figura 3.8	Diagrama de classes dos formulários	30
Figura 4.1	Tela de acesso ao sistema	33
Figura 4.2	Tela de acesso negado	33
Figura 4.3	Tela de cadastro de professores	34
Figura 4.4	Tela de cadastro de professores após a inserção de um professor	35
Figura 4.5	Confirmação de remoção de um professor	35
Figura 4.6	Tela de cadastro de estudantes	36

Figura 4.7 Tela de cadastro de semestres	36
Figura 4.8 Tela de cadastro de semestres após o cadastro de um novo item	37
Figura 4.9 Tela de visualização de propostas no perfil do administrador	38
Figura 4.10 Tela de visualização de defesas no perfil do administrador	38
Figura 4.11 Tela de consulta de relatórios	39
Figura 4.12 Tela de cadastro de projetos	40
Figura 4.13 Tela de cadastro de projetos após a inserção de novo item	40
Figura 4.14 Tela de anexo de propostas	41
Figura 4.15 Tela de cadastro de projetos após a anexação de uma proposta	41
Figura 4.16 Tela de cadastro de projetos após a aprovação do orientador	42
Figura 4.17 Tela de cadastro de projetos após a aprovação da comissão	42
Figura 4.18 Tela de solicitação de defesa	43
Figura 4.19 Tela de cadastro de projetos após a solicitação de defesa	43
Figura 4.20 Tela de cadastro de projetos após a aprovação do orientador	44
Figura 4.21 Tela de cadastro de projetos após a aprovação da comissão	44
Figura 4.22 Tela de cadastro de projetos após a comissão ter indicado que a defesa foi concluída	45
Figura 4.23 Tela de listagem de propostas submetidas pelos orientandos	46

Figura 4.24 Proposta sendo aprovada pelo orientador	46
Figura 4.25 Tela de listagem de propostas após o orientador ter aprovado uma proposta .	47
Figura 4.26 Tela de listagem de requisições de defesa submetidas pelos orientandos	47
Figura 4.27 Tela de listagem de propostas no perfil da comissão	48
Figura 4.28 Proposta sendo avaliada pela comissão	49
Figura 4.29 Tela de listagem de propostas após a comissão ter avaliado positivamente uma proposta	49
Figura 4.30 Tela de listagem de requisições de defesa submetidas pelos orientandos	50
Figura 4.31 Tela de listagem de requisições de defesa no perfil da comissão	50
Figura 4.32 Comissão marcando projeto como defendido	51

LISTA DE SIGLAS

UECE	Universidade Estadual do Ceará
PHP	PHP: Hypertext Preprocessor
TCC	Trabalho de Conclusão de Curso
IES	Instituição de Ensino Superior
MVC	Model-View-Controller
HTML	HyperText Markup Language
ORM	Object-relational mapping
SQL	Structured Query Language
XML	eXtensible Markup Language
SGBD	Sistema de Gerenciamento de Banco de Dados
DARPA	Advanced Research Project Agency
ARO	Army Research Office
IDE	Integrated Development Environment
HTTP	Hypertext Transfer Protocol
PDO	PHP Data Objects
SMTP	Simple Mail Transfer Protocol
YAML	YAML Ain't Markup Language
URL	Uniform Resource Locator
CRUD	Create, Retrieve, Update e Delete

RESUMO

Este trabalho tem como objetivo apresentar uma ferramenta que informatiza o processo de avaliação de trabalhos de conclusão da graduação em Ciências da Computação da UECE. Essa ferramenta foi projetada para minimizar a quantidade de documentos impressos e para proporcionar uma visão geral do andamento das atividades. Além disso, ela é capaz de notificar as pessoas envolvidas automaticamente, alertando sobre o cumprimento dos prazos.

Palavras-Chave: Gerenciador, Processo, Projeto final

ABSTRACT

This work aims to present a manager tool for the process of evaluating the course completion assignments of the undergraduating students in Computer Science at Ceara's State University. This tool was projected to minimize the quantity of printed documents and to provide an overview on the progress of activities. Besides, it is capable of notifying involved people automatically, making them aware of deadlines.

Keywords: Manager, Process, Course completion assignment

1 INTRODUÇÃO

Através da informatização de processos, é possível otimizar tarefas que demandam tempo, um bem escasso, e também diminuir a quantidade de documentos impressos nas nossas mesas, papéis que muitas vezes se perdem no meio de outros documentos.

Segundo (FREITAS, 1997), sistemas de informação são mecanismos cuja função é coletar, guardar e distribuir informações para suportar as funções gerenciais e operacionais das organizações. No contexto da universidade, temos muitos procedimentos padronizados por regulamentos estabelecidos pela instituição que poderiam ser controlados por sistemas de informação, diminuindo a dependência humana, que naturalmente está sujeita a erros e atrasos, economizando recursos humanos, assim como recursos materiais e centralizando a informação.

Com a manutenção dos dados em uma mesma base é possível ter uma visão geral do presente e do passado e analisar meios de melhorar ainda mais o processo. Por esse motivo, sistemas de apoio computacional são extremamente importantes, pois ajudam a coletar dados que de outra forma estariam dispersos, e que devido ao pouco tempo disponível dos coordenadores, a relação custo-benefício dessa coleta e análise manual seria desestimulante.

1.1 Motivação

A disciplina de Projeto Final que faz parte do último semestre do curso de Ciências da Computação tem por objetivo o desenvolvimento do TCC, que é um pré-requisito para a obtenção do título de Bacharel.

Considerando a importância da disciplina neste cenário, o desenvolvimento dos trabalhos é regido por um regulamento que define o processo de acompanhamento necessário para atingir os objetivos propostos. Todo aluno matriculado nessa disciplina deve seguir a regulamentação específica. Idealmente todo o processo pode ser informatizado, de forma a organizar melhor as tarefas dos professores e alunos envolvidos.

A construção de um sistema que gerencie o processo de submissão de projetos ajudaria não apenas na organização dos documentos e datas, mas também possibilitaria aos orientadores e à Comissão de Projeto Final ter um controle dos alunos que, por um motivo ou outro, não sabem por onde começar ou estão parados no meio do caminho, além de manter todos informados sobre o andamento dos seus projetos.

1.2 Objetivo

Com o desenvolvimento de uma ferramenta de apoio que gerencie a submissão de TCCs, espera-se uma diminuição no esforço despendido no controle do processo, acesso centralizado a documentos e datas, tanto para os estudantes e seus orientadores, quanto para a Comissão de Projeto Final.

Também espera-se fornecer à Comissão uma visão geral do andamento de todos os TCCs, quais foram iniciados, quais estão pendentes, quais serão concluídos em breve, quais estudantes já defenderam mas ainda não entregaram a versão final das suas monografias e um histórico/prospecção de quantos estudantes se formaram/formarão por semestre.

1.2.1 Objetivos específicos

Para conseguir desenvolver esta ferramenta de apoio, é necessário levantar os requisitos junto aos principais envolvidos, escrever casos de uso que sejam detalhados apenas o suficiente para descrever as funcionalidades do sistema, evitando perda de tempo com documentação inútil. Após o desenvolvimento são necessários testes na ferramenta e elaboração de um manual de apoio com sua instalação e utilização.

1.3 Metodologia

Para o desenvolvimento deste trabalho foi seguido o modelo de desenvolvimento em cascata que é dividido nas seguintes etapas, segundo (SOMMERVILLE, 2003):

- Análise e definição de requisitos
- Projeto de sistemas e de software
- Implementação e testes de unidade
- Integração e testes de sistema
- Operação e Manutenção

Na fase de análise e definição de requisitos foram construídos casos de usos simplificados, onde as funcionalidades foram expostas com mais clareza, mas sem muito aprofundamento para garantir que o projeto fosse concluído em tempo hábil. A partir dos casos de uso, o banco de dados foi modelado, assim como as classes de acesso a dados, o que corresponde à etapa de Projeto. Após o término da modelagem, foi iniciada a fase de codificação dos casos de uso, seguidos pelo desenvolvimento do design da aplicação. Neste passo não foram feitos testes de unidade codificados, apenas testes manuais. Após o desenvolvimento foi feita a integração dos módulos e os testes de sistema.

No momento da elaboração desta monografia o sistema ainda não se encontra em uso, mas espera-se que o próximo passo seja o último passo do modelo cascata, a operação e manutenção.

O modelo cascata, apesar de inflexível, foi bastante apropriado pois havia um entendimento claro dos requisitos do sistema e ofereceu uma previsibilidade maior dos prazos.

Para a construção da aplicação foi utilizada a linguagem de script PHP, na sua versão 5.3.3 em conjunto com o framework Symfony 1.4.2. Como IDE, foi utilizado o NetBeans 6.9, um framework para desenvolvimento e testes com PHP. O banco de dados escolhido foi o PostgreSQL versão 8.4.

Estas ferramentas foram escolhidas devido a serem gratuitas e possuírem versões estáveis para Linux, que é a plataforma onde se pretende implantar a versão final da aplicação. Mais detalhes sobre as tecnologias são expostas na Seção 3.1.

1.4 Organização do trabalho

Além deste capítulo introdutório, o presente trabalho consiste em mais 4 capítulos. No Capítulo 2 é feita uma apresentação do processo de entrega de TCCs na UECE, de acordo com o regulamento da instituição. No Capítulo 3 é descrito como foi o processo de desenvolvimento da aplicação, assim como a organização do banco de dados e os requisitos do sistema. O Capítulo 4 demonstra como utilizar o sistema. O Capítulo 5 apresenta as conclusões e uma breve discussão sobre trabalhos futuros.

2 REGULAMENTO DE PROJETO FINAL

O processo de submissão e avaliação de projetos finais no curso de Ciências da Computação da UECE segue um regulamento que normatiza o tipo do conteúdo da monografia, o procedimento para o desenvolvimento e para a aprovação do projeto, a constituição da Comissão de Projeto Final e as atribuições desta, do orientador e do aluno.

2.1 Entidades envolvidas e suas atribuições

O desenvolvimento do projeto final deve ser desempenhado individualmente pelo aluno, sob a orientação de um docente, o orientador. O orientador deve ser um docente lotado no curso de Ciências da Computação da UECE, seja ele professor efetivo, substituto ou visitante. O aluno pode ainda contar com a colaboração de co-orientadores, podendo estes serem docentes da UECE ou de outras IES ou ainda profissionais com graduação plena em Ciências da Computação ou cursos afins e com no mínimo 3 (três) anos de experiência em orientação de alunos ou coordenação de projetos.

A Comissão de Projeto Final, ou simplesmente Comissão, é o órgão responsável pelo acompanhamento do processo de desenvolvimento do projeto final. Ela é composta por 3 (três) docentes efetivos pertencentes ao curso de Bacharelado em Ciência da Computação da UECE, havendo ainda 2 (dois) membros suplentes, sendo todos esses (permanentes e suplentes) escolhidos através de eleição no Colegiado do curso de Ciência da Computação e nomeados pelo Coordenador da graduação. O membro da Comissão fica impedido de emitir parecer sobre o trabalho de seus orientandos, que, neste caso, deverão ser avaliados por um membro suplente.

Compete ao aluno:

- a. elaborar projeto de Proposta de Projeto Final;
- b. conduzir e executar o Projeto Final;
- c. cumprir os prazos estabelecidos no cronograma pré-estabelecido;
- d. redigir e defender o Projeto Final;
- e. entregar cópia corrigida do Projeto Final à secretaria;
- f. tomar ciência dos prazos estabelecidos pela Comissão de Projeto Final e cumpri-los.

Compete ao orientador e co-orientador:

- a. orientar o aluno na organização de seu plano de estudo, pesquisa e assisti-lo na preparação da monografia;
- b. viabilizar a realização do Projeto Final;
- c. encaminhar a Proposta de Projeto Final e a Solicitação de Defesa à Comissão;
- d. propor à Comissão a composição da Banca Examinadora;
- e. encaminhar a Ata de Defesa, devidamente preenchida e assinada, ao Coordenador do curso.

Compete à Comissão:

- a. aprovar a proposta e plano de trabalho de Projeto Final;
- b. aprovar as indicações dos orientadores de Projeto Final que não sejam docentes do curso;
- c. aprovar os membros das bancas avaliadoras do Projeto Final;
- d. autorizar a defesa de monografia de Projeto Final;

2.2 Desenvolvimento do projeto

O projeto pode ser iniciado antes do aluno se matricular na disciplina de Projeto Final, porém o processo de desenvolvimento do trabalho deve ser realizado no último ano do curso. O desenvolvimento do projeto final é constituído das seguintes partes:

- a. Apresentação da proposta de projeto final à comissão de projeto final;
- b. Solicitação de defesa do projeto final e indicação de comissão examinadora à comissão de projeto final;
- c. Defesa do projeto final em seção pública diante da comissão examinadora;
- d. Entrega do texto final da monografia.

2.2.1 Proposta do projeto

O aluno deve apresentar uma proposta de projeto (ver Anexo 7.1) a partir do início do período letivo em que se matriculou na disciplina de Projeto Final. A data máxima de apresentação da proposta é de 100 (cem) dias antes da data da colação oficial ou especial.

A apresentação da proposta deve ter a anuência do orientador e é avaliada pela comissão. Após a aprovação da proposta pela comissão, fica autorizado o início do desenvolvimento do trabalho. A proposta deve conter os seguintes tópicos:

- a. Motivação e Objetivo;
- b. Fundamentação teórica;
- c. Metodologia;
- d. Bibliografia;
- e. Cronograma.

2.2.2 Defesa do projeto

O aluno, com a anuência do orientador, deve encaminhar uma solicitação de defesa à comissão (ver Anexo 7.2), no mínimo 60 (sessenta) dias após a aprovação da proposta e no máximo até 30 (trinta) dias antes da colação oficial ou especial. Junto da solicitação deverão seguir a data de sugestão da defesa, a indicação dos membros da comissão examinadora e 1 (uma) cópia da monografia.

Sendo o parecer da comissão favorável, o aluno tem um prazo de até 10 (dez) dias antes da colação oficial ou especial para realizar a defesa, e 7 (sete) dias antes desta para entregar à comissão examinadora exemplares da monografia.

Sendo o aluno aprovado na defesa, ele deverá entregar à secretaria do curso 3 (três) exemplares impressos da monografia e 1 (uma) cópia em meio eletrônico. Somente após isso é que será autorizada a emissão e entrega do diploma ao aluno.

2.2.3 Resumo do fluxo do desenvolvimento do projeto final

A Figura 2.1 apresenta um fluxo relativo ao processo de desenvolvimento do projeto final. Inicialmente, o aluno submete uma proposta de acordo com o modelo contido no Anexo 7.1, que deve ter a anuência do seu orientador. Caso o orientador não aprove a proposta, convém que o aluno converse com seu orientador para corrigir os eventuais problemas para que dessa forma possa submeter a proposta novamente. Após obter o aval do orientador, a proposta pode ser encaminhada à comissão. Surgindo algum problema na aprovação da proposta por parte da comissão, o aluno deve corrigir os problemas indicados e submetê-la novamente. Caso a proposta seja aprovada, o aluno pode começar a desenvolver o trabalho. Finalizado o desenvolvimento, o aluno pode solicitar a defesa do seu projeto enviando 1 (uma) cópia da monografia e preenchendo o formulário do Anexo 7.2. O pedido deve novamente contar com a anuência do orientador, para depois seguir para a comissão. Sendo a defesa aprovada pela comissão, o aluno está apto para defender seu projeto ante a comissão examinadora.

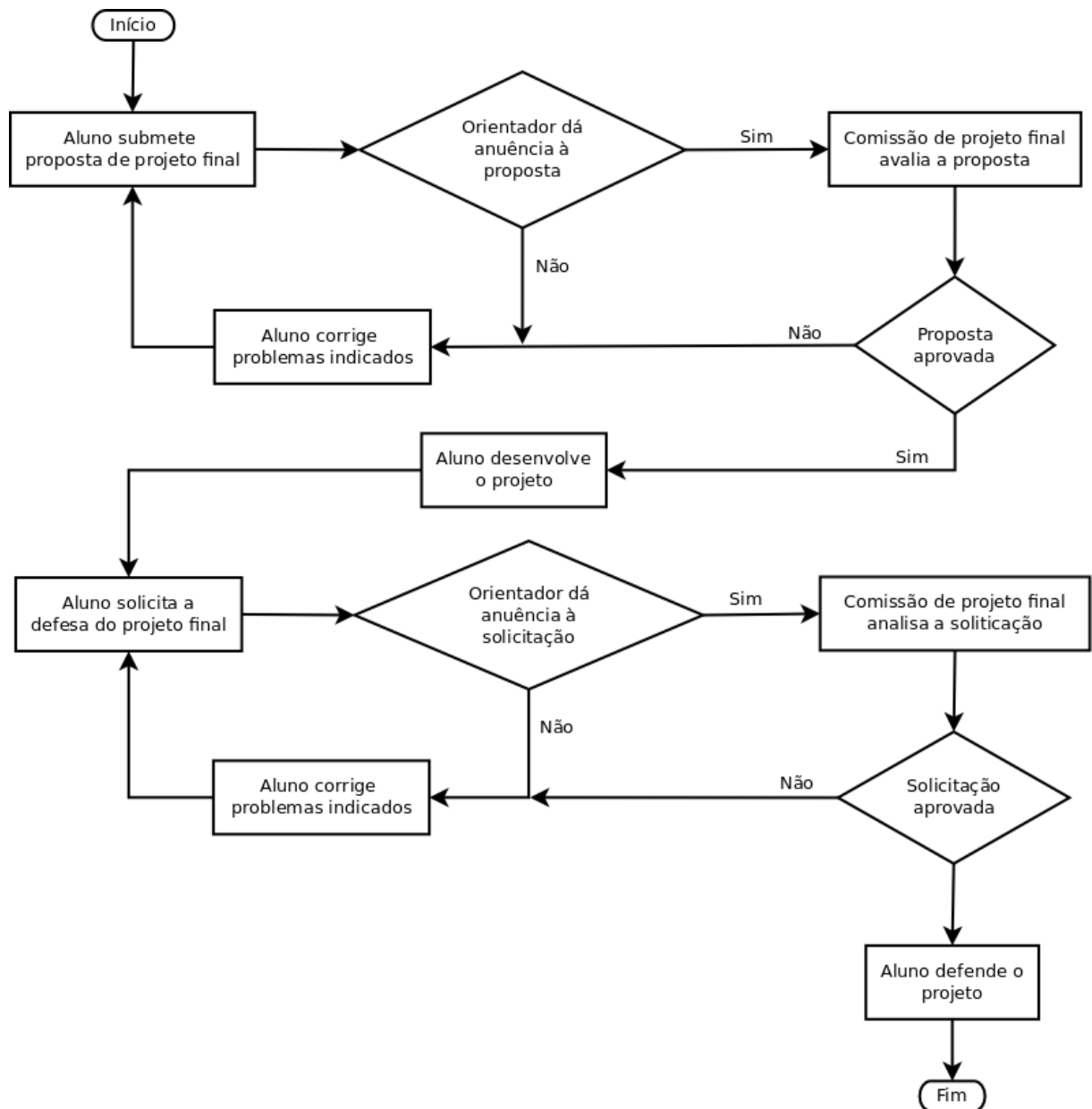


Figura 2.1: Fluxograma do desenvolvimento do projeto final

3 PROCESSO DE DESENVOLVIMENTO DA APLICAÇÃO

No capítulo anterior foi apresentado o processo de desenvolvimento e avaliação dos TCCs no curso de Ciências da Computação da UECE, como este se encontra atualmente.

A partir da visão geral fornecida pelo fluxograma da Figura 2.1 foi possível definir o comportamento básico da aplicação. Contudo, a aplicação não se detém a apenas seguir esse fluxo básico. Ela também possui funcionalidades de alerta no sistema, para que os envolvidos no processo sejam sempre notificados a cada vez que um estudante, orientador ou comissão executem alguma operação em cima de um projeto.

Neste capítulo discutiremos sobre como o sistema foi modelado, assim como sobre as estratégias delineadas para a notificação de eventos.

3.1 Escolha das tecnologias

A seguir são apresentadas as tecnologias escolhidas como parte da plataforma de desenvolvimento do sistema.

3.1.1 Linguagem de programação

O PHP é uma linguagem de programação interpretada, multiparadigma, de código aberto, e especialmente voltada para o desenvolvimento de aplicações para a Web. Possui uma sintaxe que lembra C, Java e Perl, e se distingue das demais por sua facilidade de aprendizado. Começou a ser desenvolvida em 1994 por Rasmus Lerdorf, mas o paradigma de orientação a objetos só foi introduzido a partir da versão 3 (três), amadurecendo na versão 5 (cinco) (PHP.NET, 2011; WIKIPEDIA, 2011a).

O PHP se encontra disponível na grande maioria dos servidores Web e, devido a sua facilidade de aprendizado, possui uma vasta comunidade de desenvolvedores. Ele é usado em alguns gigantes da Web, como Facebook, Wikipédia e Wordpress (INFOQ, 2009; WIKIPEDIA, 2011b; WORDPRESS, 2011).

Entre outros fatores, esta linguagem foi escolhida devido a sua difusão, sendo então mais provável encontrar outros desenvolvedores que possam dar continuidade ao projeto deste

trabalho, adicionando novas funcionalidades ou corrigindo eventuais problemas.

3.1.2 Framework

De acordo com (MINETTO, 2007), um framework de desenvolvimento é uma base de onde se pode desenvolver algo maior ou mais específico. É uma coleção de códigos-fonte, classes, funções, técnicas e metodologias que facilitam o desenvolvimento de novos softwares.

O uso de um framework é essencial para desenvolver uma aplicação rapidamente sem deixar de seguir boas práticas de programação. Além disso, um programador que conheça um framework não tem dificuldades para entender o código de outras pessoas, pois o framework obriga todos a seguirem as mesmas convenções. Dessa forma o programador que não conhece a aplicação pode se manter apenas no entendimento da lógica de negócio, sem se perder na arquitetura da aplicação.

Para a aplicação desenvolvida neste trabalho, o framework escolhido foi o Symfony. O Symfony é um framework que possui alta aceitação na comunidade PHP, boa documentação, é patrocinado pela empresa francesa Sensio Labs, que garante suporte técnico a longo prazo, e tem várias outras qualidades que o coloca entre os melhores frameworks PHP, como:

- a. suporte a PHP 5;
- b. suporte a MVC;
- c. validação de formulários;
- d. extensa documentação;
- e. suporte a plugins;
- f. geração de código;
- g. suporte a ORM e múltiplos bancos de dados;
- h. convention over configuration;
- i. suporte a testes unitários.

3.1.2.1 MVC

O MVC é um acrônimo para Model-View-Controller, um padrão de projeto que tem como intuito separar a lógica de negócio, a interface e os modelos de acesso a dados. Essa separação de conceitos tem como propósito evitar que o código fique difícil de manter e auxilia significativamente no aumento do reuso de código. A Figura 3.1 apresenta de maneira geral a estrutura desse tipo de arquitetura.

O modelo de acesso a dados é representado pela camada Model (ou camada de modelo). O modelo é um objeto que representa alguma informação sobre o domínio. É um objeto

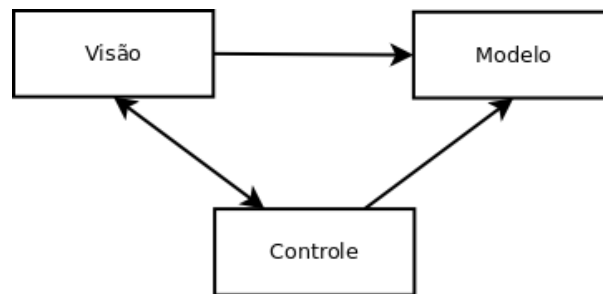


Figura 3.1: Arquitetura MVC

não-visual que contém todos os dados e comportamentos outros que não os utilizados pela interface (FOWLER, 2006).

A camada View (ou camada de visão) representa a interface da aplicação. No trabalho em questão, ela é a parte em HTML. Esta camada não deve possuir nenhuma lógica de negócio, detendo-se apenas à captura e exibição de dados.

A camada Controller (ou camada de controle) é a responsável por conectar as outras duas camadas. O controlador recebe a entrada do usuário (capturado pela visão), manipula o modelo e faz com que a visão seja atualizada apropriadamente (FOWLER, 2006).

3.1.2.2 ORM

Atualmente a maioria das aplicações é desenvolvida utilizando o paradigma de programação orientado a objetos apoiada por um banco de dados relacional. Essas aplicações precisam carregar dados de um banco de dados, criar objetos para representar esses dados em memória, executar operações em cima destes objetos e depois salvar de volta as alterações no banco.

Ferramentas de mapeamento objeto-relacional (ou ORM) são frameworks que recuperam e persistem objetos. Seu objetivo é dar suporte à complexa atividade de gerenciar conexões entre objetos e um banco de dados relacional. A persistência fica transparente ao desenvolvedor, já que ele não precisa se preocupar com os detalhes de implementação. A ponte entre objetos e seus relacionamentos é realizada pela ferramenta ORM segundo a especificação de mapeamento dos dados (CABIBBO; CAROSI, 2005).

3.1.2.3 *Convention over configuration*

Frameworks de propósito geral normalmente necessitam de um ou mais arquivos de configuração para serem utilizados adequadamente. Um arquivo de configuração mapeia uma classe e um recurso (uma tabela no banco de dados) ou um evento (uma requisição web). À medida em que a complexidade das aplicações cresce, os arquivos de configuração também crescem, tornando-se difíceis de manter (CHEN, 2006). Para evitar este mal desnecessário, muitos frameworks atualmente procuram seguir o modelo de desenvolvimento de software de Convenção sobre Configuração (*Convention over Configuration*). A idéia é basicamente

fazer com que o desenvolvedor só precise definir aquilo que não segue uma convenção pré-estabelecida.

```
<hibernate-mapping>
<class name="User" table="users">
  <id name="ID" column="id" type="string">
    <generator class="assigned"></generator>
  </id>
  <property name="password" column="password" type="string" />
</class>
</hibernate-mapping>
```

Figura 3.2: Definição de um mapeamento no Hibernate

```
CREATE TABLE users (
  id VARCHAR(20) NOT NULL,
  password VARCHAR(20),
  PRIMARY KEY(id)
);
```

Figura 3.3: Tabela Users no banco de dados

A Figura 3.2 apresenta um arquivo de mapeamento para Hibernate, um framework de mapeamento objeto-relacional para Java. O código da Figura 3.2 mapeia a classe User com a tabela Users no banco de dados. A tabela Users é descrita na Figura 3.3 usando SQL. Os campos da classe User também são mapeados para as colunas da tabela Users.

O ato de modificar arquivos de configuração, normalmente em XML, é tedioso e propenso a erros. A maioria dos problemas de configuração só vai ser detectado em tempo de execução, disparando exceções na aplicação, que tendem a diminuir o ritmo do desenvolvimento e consequentemente a produtividade. Mais importante ainda, uma grande parte do mapeamento poderia ser inferido facilmente pela estrutura da tabela sem a necessidade de configuração alguma. Por exemplo, pode-se estabelecer uma convenção de que:

1. Nomes de tabelas devem ser o nome da classe no plural.
2. As colunas na tabela devem ter nomes idênticos aos campos que a classe mapeia.

Estas duas convenções são naturais, e, de fato, já são seguidas pela maioria dos desenvolvedores. O padrão de convenção sobre configuração reduz a quantidade de configuração ao estabelecer um conjunto de convenções de nomenclatura que todos os desenvolvedores devem seguir (CHEN, 2006).

3.1.3 SGBD

O PostgreSQL é um SGBD livre, de código aberto, bastante robusto e confiável. Derivou-se do projeto POSTGRES da universidade de Berkley, que iniciou-se em 1986 e foi patrocinado por instituições militares americanas como a DARPA (Agência de Projetos de Pesquisa Avançada para Defesa) e ARO (Departamento de Pesquisa Militar). A linguagem de consultas SQL foi inserida quando o nome do projeto era Postgres95, tendo sido rebatizado para o nome atual em 1996 para enfatizar a relação do SGBD com o SQL (POSTGRESQL, 2003).

Apesar do PostgreSQL ter sido escolhido para o desenvolvimento deste trabalho, outros SGBDs podem ser utilizados em seu lugar, devido ao framework Symfony possuir um

ORM que abstrai a comunicação da aplicação com o banco de dados. Basta configurar a conexão do banco de dados no Symfony e avisá-lo qual SGBD estará em uso que o seu ORM se encarregará de fazer a comunicação correta com o banco de dados.

3.1.4 IDE

O NetBeans é um ambiente integrado de desenvolvimento (IDE) gratuito, também de código aberto e atualmente patrocinado pela Oracle. Originalmente suportava apenas a linguagem Java, mas atualmente consegue trabalhar com diversas linguagem de programação, entre elas o PHP, e além disso possui plugins que facilitam a utilização de alguns frameworks, inclusive o Symfony.

Foi escolhido para este trabalho por fornecer um bom suporte ao PHP e ao Symfony, e por possuir fácil integração com a ferramenta de depuração Xdebug, específica para PHP.

3.1.5 Outros requisitos de plataforma

Para que a aplicação funcione corretamente, é necessário que o servidor onde ela será atenda a alguns pré-requisitos. São eles:

PHP Deve ser utilizada a versão 5.2.4 ou mais recente (exceto a versão 5.2.9).

Servidor Recomenda-se a utilização do servidor Apache versão 2 ou superior, com a extensão `mod_rewrite` instalada. O projeto não foi testado com outros servidores HTTP.

SGBD Recomenda-se PostgreSQL 8.4 como SGBD por ter sido utilizado durante o desenvolvimento da aplicação, mas de acordo com a documentação do Doctrine, o ORM do Symfony, qualquer banco de dados suportado pelo PHP através dos drivers PDO pode ser utilizado, já que ele utiliza PDO para se comunicar com o banco de dados. A aplicação foi seguramente testada com MySQL Community Edition 5.5.9 e SQLite 3.7.5, podendo estes também serem utilizados sem prejuízo algum ao funcionamento do sistema. Para qualquer que seja o banco escolhido, o driver PDO deve estar instalado e configurado no PHP.

E-mail É necessária uma conta de email em um servidor que aceite conexões externas via SMTP para o envio das mensagens eletrônicas. Alternativamente pode-se utilizar o Sendmail, caso este esteja configurado no servidor, ou deixar a cargo da função `mail` do PHP. Recomenda-se configurar um servidor SMTP, especialmente pela facilidade de configuração deste se comparado ao Sendmail. Não é recomendado utilizar a função `mail` do PHP, pois os emails enviados tendem a ser identificados como spam em muitos servidores de email.

3.2 Casos de uso e especificação de requisitos do sistema

Para aproveitar a funcionalidade de comunicação com diferentes tipos de SGBDs, o ORM do Symfony nos permite definir toda a estrutura do banco de dados textualmente, no formato YAML, que possui maior legibilidade que o SQL. A Figura 3.4 apresenta um exemplo de como as tabelas são descritas no Symfony. Após a descrição de todas as tabelas, executamos um comando no Symfony que gera o SQL necessário para a criação das tabelas de acordo com o SGBD escolhido. Ele também possui um comando que gera todas as classes e formulário necessários para que possamos trabalhar com essas tabelas que acabamos de definir.

```
Estudante:
columns:
  id:
    primary: true
    type: integer
  telefone:
    type: string(14)
relations:
  Usuario:
    local: id
    type: one
    foreignType: one
    cascade: [delete]
```

Figura 3.4: Descrição de uma tabela no Symfony

Essa geração automática de código é uma ferramenta extremamente poderosa que o framework nos fornece para evitar perder tempo com tarefas triviais. Mas e se já tivermos começado a codificar nas classes que ele gerou, e lembrarmos que precisamos de mais um campo na tabela? Será que vamos ter todo o código perdido, já que ele vai sobrescrever as classes geradas automaticamente? Isso não acontece, pois o Symfony gera duas classes para cada tabela que nós definirmos. Cada classe herda de uma classe base abstrata, que é sobrescrita toda vez que pedimos ao Symfony que gere as classes de acordo com a especificação. Nós temos que trabalhar em cima dessas classes que herdam as classes bases abstratas, pois essas últimas é que o Symfony sempre vai sobrescrever.

Pra ficar mais claro, damos uma olhada no diagrama da Figura 3.5. Ela apresenta todas as classes base que o Symfony gerou automaticamente para a aplicação deste trabalho. Para cada classe BaseEntidade, há uma classe Entidade, como exposto na Figura 3.6, que é criada uma única vez, caso ela ainda não exista. É nesta última que colocamos os métodos da nossa lógica de negócios. Tendo isso em mente, podemos prosseguir na explicação dos diagramas.

3.2.1 Diagrama de classes

A Figura 3.5 apresenta o diagrama de classes do TCC-Manager. Todas as classes herdam direta ou indiretamente da classe sfDoctrineRecord, pertencente ao framework ORM do Symfony. Essa associação não foi explicitada no diagrama para que ele ficasse mais claro. A classe sfDoctrineRecord é uma classe abstrata que possui todos os métodos de persistência dos objetos.

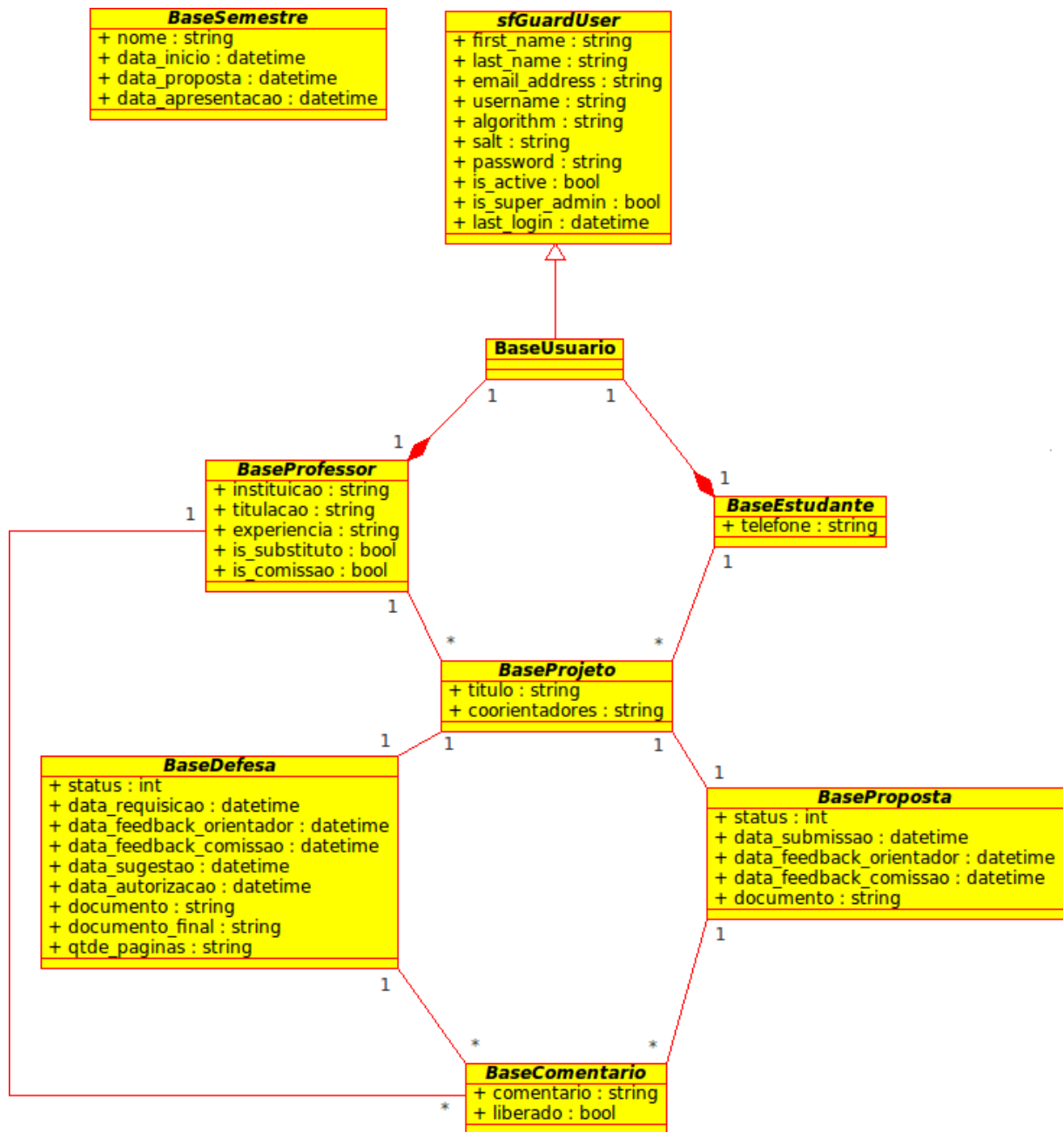


Figura 3.5: Diagrama de classes base

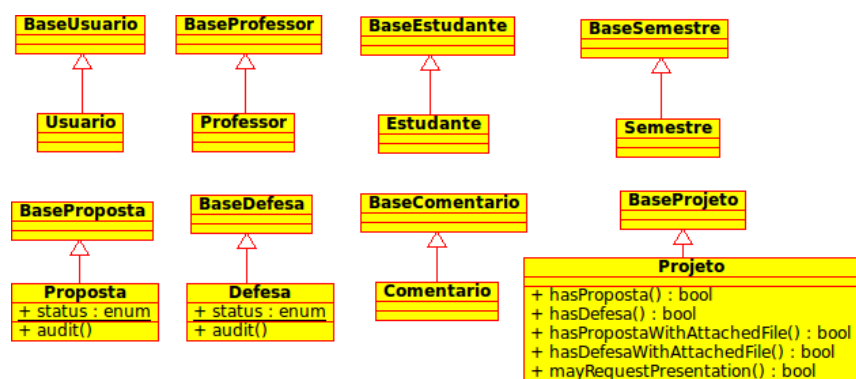


Figura 3.6: Diagrama de classes extendidas

A Figura 3.6 apresenta as classes das entidades de trabalho, isto é, aquelas que herdam das classes base. Nelas é que deve ser inserida qualquer lógica associada à entidade, pois o framework não escreve nessas classes, a não ser para criá-las pela primeira vez. As classes Proposta e Defesa possuem o método audit, que é invocado quando da avaliação, por parte da comissão, da proposta ou da solicitação de defesa. Cada vez que um professor integrante da comissão dá seu parecer, este método salva as informações do integrante, seu parecer e um possível comentário feito por ele sobre a proposta/defesa. Além disso, ele calcula o parecer final dependendo da decisão da maioria da comissão. Essas duas entidades também guardam uma enumeração que indica seu status, que podem ser os seguintes:

- NAO_ANALISADO: Proposta/Defesa não analisada
- APROVADO: Proposta/Defesa aprovada pelo orientador
- REPROVADO: Proposta/Defesa reprovada pelo orientador
- LIBERADO: Proposta/Defesa aprovada pela comissão
- NAO_LIBERADO: Proposta/Defesa reprovada pela comissão

A classe Projeto possui alguns métodos de verificação de situação. Os nomes dos métodos são bem autodescritivos e indicam se o projeto possui uma proposta/defesa, se o estudante anexou o documento da proposta ou a cópia da monografia na solicitação de defesa, e se o projeto já pode ser defendido.

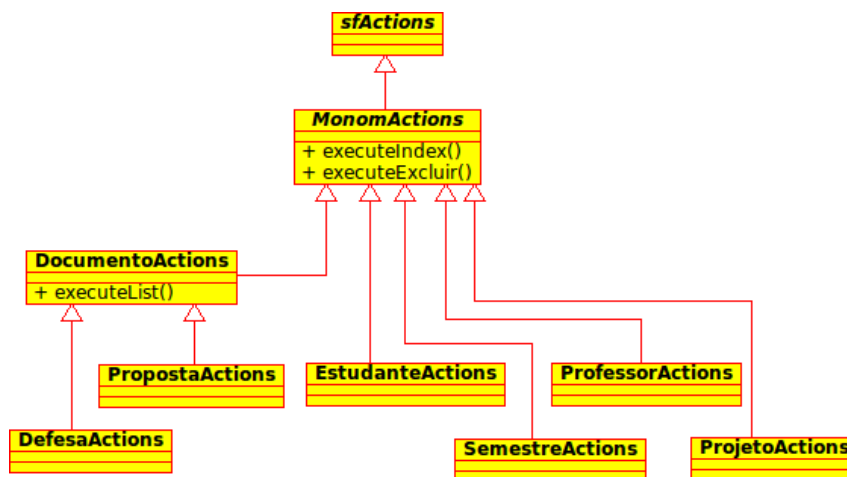


Figura 3.7: Diagrama de classes da camada de controle

A Figura 3.7 apresenta o diagrama de classes para a camada de controle. A classe sfActions pertence ao Symfony e cuida da comunicação entre o navegador e a aplicação. Internamente é ela que instancia as classes necessárias para a execução do framework; decodifica a URL da requisição, de forma a determinar qual ação e qual módulo estão sendo requisitados, e se não existirem, exibe uma mensagem de página não encontrada; recebe os parâmetros de entrada; chama a ação do módulo requisitado e renderiza a saída.

Na classe de controle `monomActions` encontram-se métodos de CRUD padronizados, e que são usados pela maior parte dos módulos da aplicação. A classe de controle `documentoActions` contém métodos de controle de upload de arquivos, que são usados nos módulos de proposta e defesa. Dessa forma, foi possível fazer um grande reuso de código.

3.2.1.1 Autenticação de usuários

Na Figura 3.5 podemos observar a classe `sfGuardUser`, da qual a classe `Usuario` estende. Ela vem de um plugin para o Symfony de controle de usuários, que oferece recursos de segurança e autorização em cima dos recursos de segurança oferecidos pelo framework. Aproveitamos esse plugin para fazer um bom reuso de código e contar com toda a segurança que o plugin já disponibiliza (SYMFONY, 2010).

As classes `Estudante` e `Professor` possuem um relacionamento de composição com a classe `Usuario`. Inicialmente foi pensado em fazer com que essas classes herdassem da classe `Usuario`, mas visto que existe o usuário administrador que não é nem estudante, nem professor, a modelagem não pôde ser feita dessa forma. De qualquer forma, só podem existir estudantes e professores se houver um usuário associado a eles.

Na classe `Usuario` encontram-se os dados mais básicos de cada usuário, como nome, endereço de email e senha. Há ainda outras informações que dizem respeito a como a senha é criptografada no banco de dados. O campo `algorithm` contém o algoritmo utilizado para criptografar a senha, que por padrão é o SHA1. Outros algoritmos podem ser utilizados, como o MD5, mas optou-se por deixar o algoritmo padrão. O campo `salt` é uma string que é concatenada à senha antes da criptografia, para dificultar a descoberta do valor original.

3.2.1.2 Formulários

Além de gerar as entidades, o Symfony também possui a funcionalidade de gerar formulários que possuem validação própria. Dessa forma, só é necessário ajustar detalhes de aparência ou validações mais complexas, como datas que devem preceder umas às outras ou valores que devem estar dentro de um conjunto fechado.

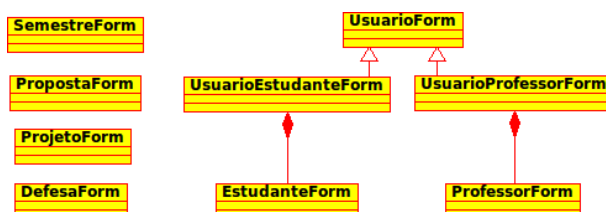


Figura 3.8: Diagrama de classes dos formulários

A Figura 3.8 apresenta os formulários que foram gerados a partir da descrição do esquema do banco de dados. Todas as classes desse diagrama herdam das suas correspondentes classes base, assim como acontece com as classes da Figura 3.6. Elas não são apresentadas

aqui por efeito de clareza no diagrama. As classes base, por sua vez, herdam da classe Base-FormDoctrine, do framework, que possui os métodos básicos de validação e persistência dos formulários.

O formulário do Anexo 7.1 é representado no sistema pelos formulários de estudante, professor, projeto e proposta. O formulário do Anexo 7.2 é representado no sistema pelos formulários de estudante, professor, projeto e defesa.

3.2.1.3 Casos de uso

Para a execução deste projeto, foram escritos os casos de uso de uma forma básica, de maneira a não tomar muito tempo que seria usado na codificação. Os casos de uso encontram-se listados no Apêndice 6.1.

4 UTILIZAÇÃO DO SISTEMA

Neste capítulo descrevemos inicialmente a utilização do sistema. A descrição tende a ser sucinta, visto que a ferramenta é bastante simples e as telas são auto-explicativas. Posteriormente, descrevemos os principais passos para a instalação e configuração do sistema no servidor web.

4.1 Utilização do sistema

O TCC-Manager consiste de vários módulos, que possuem comportamentos diferentes dependendo do tipo de usuário que está autenticado no sistema. São 4 (quatro) perfis possíveis:

Administrador Adminstrador do sistema. Cabe a este perfil cadastrar alunos e professores. Ele também tem acesso somente-leitura às propostas e defesas.

Estudante Possui acesso somente a manutenção de seus próprios projetos.

Professor Possui acesso às propostas e solicitações de defesa de seus orientandos.

Comissão Possui acesso a todas as propostas e solicitações de defesa do sistema.

Devido a essas diferenças de perfis, dividiremos a utilização do sistema de acordo com cada um.

4.1.1 Telas em comum entre todos os perfis

A tela de acesso ao sistema (Figura 4.1) e a tela de acesso negado (Figura 4.2) são as mesmas para todos os perfis. Na tela de acesso ao sistema (Figura 4.1), o campo login espera receber o nome de usuário para todos os perfis exceto para estudantes, que devem digitar a sua matrícula.

A tela de acesso negado (Figura 4.2) é exibida quando um usuário tenta acessar uma tela de outro perfil, sem ter a permissão necessária para isso.

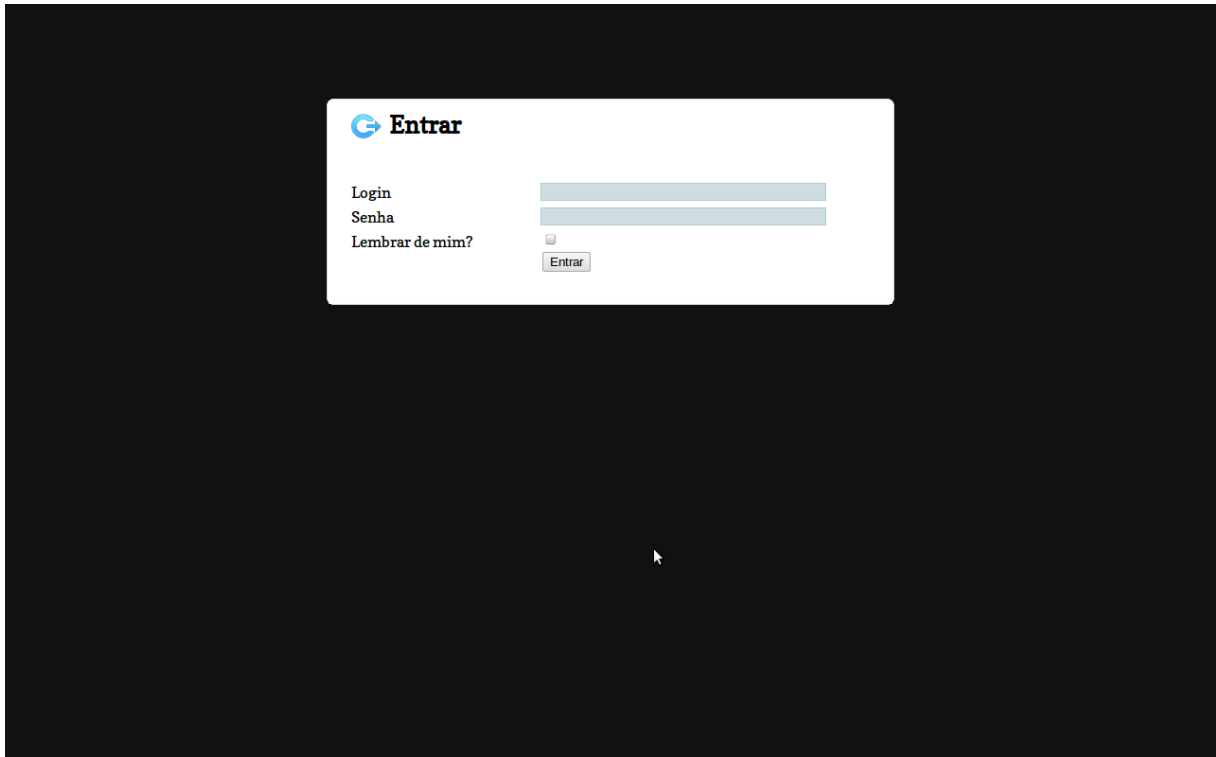


Figura 4.1: Tela de acesso ao sistema

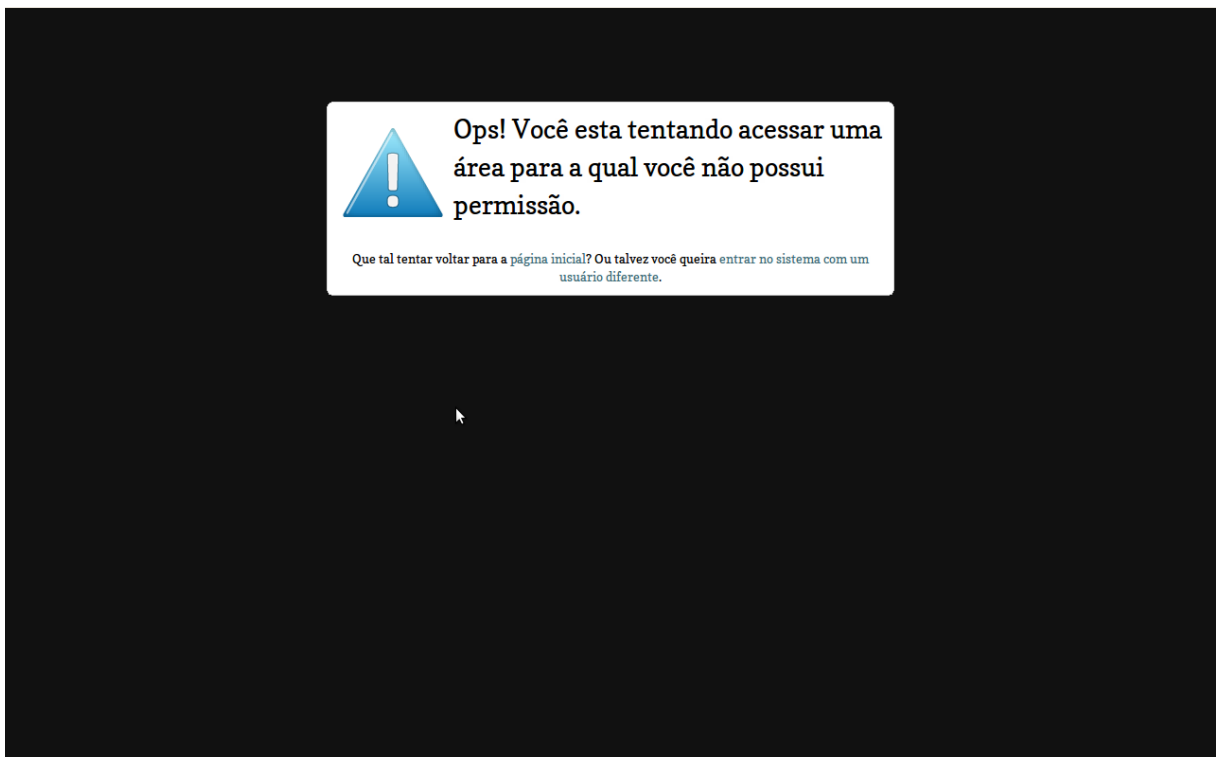


Figura 4.2: Tela de acesso negado

4.1.2 Perfil de administrador

O perfil do administrador consiste basicamente dos cadastros do sistema. A seguir veremos as telas de cadastro de professores, alunos e semestres. Em seguida vemos que este perfil também pode visualizar quais propostas e defesas estão cadastradas no banco de dados e pode consultar relatórios.

Nome	Sobrenome	Nome de usuário	Email	Ações
Wesley	Alcoforado	wesleyjefferson	wesleyjefferson@gmail.com	
Clécio	Fontenele	clecio	clecio@uece.br	

Figura 4.3: Tela de cadastro de professores

Na Figura 4.3 mostra a tela de cadastro de professores, que é acessada ao clicar na opção Professor no menu do sistema. Podemos ver que na tela já se encontram disponíveis o formulário de edição e a listagem de todos os professores cadastrados no sistema. Na coluna Ações, na listagem de professores, estão disponíveis as ações de edição e remoção de registro. A Figura 4.3 exibe o formulário sendo preenchido para a inserção de um novo professor, ao passo que a Figura 4.4 exibe a listagem atualizada após a inserção de um novo professor no sistema. Se o administrador clicar no ícone de remoção de registro, uma janela de confirmação é exibida para que o administrador confirme seu desejo de remoção do registro, visto que esta ação não pode ser revertida. Este processo pode ser visualizado na Figura 4.5.

A Figura 4.6 exibe a tela de cadastro de estudantes. O mecanismo de funcionamento desta é exatamente igual ao de cadastro de professores, portanto não há necessidade de mais detalhes.

A Figura 4.7 mostra a tela de cadastro de semestres. Esta possui um funcionamento muito parecido com o cadastro de professores e estudantes, porém possui uma particularidade que convém explicitar. O único campo obrigatório é a data de colação normal. A partir da informação desta data, pode-se inferir todas as outras. O cálculo é feito a partir das informações

Professores

Nome de usuário

Email

Senha

Confirmação da senha

Nome

Sobrenome

Instituição

Titulação

Experiência

Substituto? ☐

Comissão? ☐

Ativo? ☒

Superusuário? ☐







Nome	Sobrenome	Nome de usuário	Email	Ações
Wesley	Alcoforado	wesleyjefferson	wesleyjefferson@gmail.com	 
Clécio	Fontenele	clecio	clecio@uece.br	 
Mariela	Cortés	mariela	mariela@larces.uece.br	 

Figura 4.4: Tela de cadastro de professores após a inserção de um professor

Nome de usuário

Email

Senha

Confirmação da senha

Nome

Sobrenome

Instituição

Titulação

Experiência

Substituto? ☐

Comissão? ☐

Ativo? ☒

Superusuário? ☐

A página em localhost diz:

?

Você tem certeza de que deseja excluir este registro?







Nome	Sobrenome	Nome de usuário	Email	Ações
Wesley	Alcoforado	wesleyjefferson	wesleyjefferson@gmail.com	 
Clécio	Fontenele	clecio	clecio@uece.br	 
Mariela	Cortés	mariela	mariela@larces.uece.br	 

Figura 4.5: Confirmação de remoção de um professor

Início

Professores

Estudantes

Semestres

Propostas

Defesas

Relatórios

Sair

Estudantes

Matrícula

Email

Senha

Confirmação da senha

Nome

Sobrenome

Telefone

Ativo? ☒

Salvar



Nome	Sobrenome	Matrícula	Telefone	Email	Ações
Raquel	Andrade	0881801	85 88633619	raquelandr@gmail.com	 

Figura 4.6: Tela de cadastro de estudantes

Início

Professores

Estudantes

Semestres

Propostas

Defesas

Relatórios

Sair

Semestres

Nome

Colação normal *

Colação especial

Entrega da proposta

Entrega do copião

Defesa

Salvar



Semestre	Proposta	Copião	Defesa	Colação	Especial	Ações
teste	12/01/2010	12/02/2011	05/03/2011	12/03/2011	12/04/2011	 

Figura 4.7: Tela de cadastro de semestres

Semestres

Nome

Colação normal * 27 03 2011

Colação especial

Entrega da proposta

Entrega do copião

Defesa

Semestre	Proposta	Copião	Defesa	Colação	Especial	Ações
teste	12/01/2010	12/02/2011	05/03/2011	12/03/2011	12/04/2011	
2011.2	27/12/2010	27/02/2011	20/03/2011	27/03/2011	27/04/2011	

Figura 4.8: Tela de cadastro de semestres após o cadastro de um novo item

estabelecidas no regulamento da universidade, que pode ser conferido nas Seções 2.2.1 e 2.2.2. A Figura 4.8 exibe o item inserido no banco de dados com as datas calculadas a partir da data de colação normal. Se por algum motivo as datas forem diferentes, o administrador pode informá-las separadamente. O sistema só tenta calcular aquelas que não foram informadas.

A Figura 4.9 exibe a tela de visualização de propostas e a Figura 4.10 exibe a tela de visualização de defesas no perfil do administrador. Ambas possuem um ícone em formato de lupa que permite visualizar a proposta do projeto ou o copião (no caso de uma solicitação de defesa), e possuem um ícone para remover o registro do banco de dados. A Figura 4.11 exibe a tela de consulta de relatórios, que é compartilhada entre os perfis de administrador e comissão.

4.1.3 Perfil de estudante

O estudante só precisa se preocupar com o desenvolvimento do seu próprio projeto, por isso, o único módulo que ele precisa ter acesso é o de projetos. A Figura 4.12 exibe o formulário de projetos sendo preenchido pelo estudante, onde ele só precisa informar o nome do seu projeto e quem são seus orientadores. A Figura 4.13 exibe como fica a listagem de projetos após a inserção do novo item. Podemos notar que quando o projeto é cadastrado, seu status indica que a proposta está pendente. A Figura 4.14 exibe a tela de anexo de propostas, que pode ser acessada pelo ícone em formato de clipe na coluna Proposta. Após a anexação da proposta (Figura 4.15), um email é enviado ao orientador, informando-o que um de seus alunos acabou de anexar uma proposta e que sua aprovação é necessária. O aluno deve aguardar que o orientador e a comissão aprovem sua proposta para iniciar o desenvolvimento do trabalho.

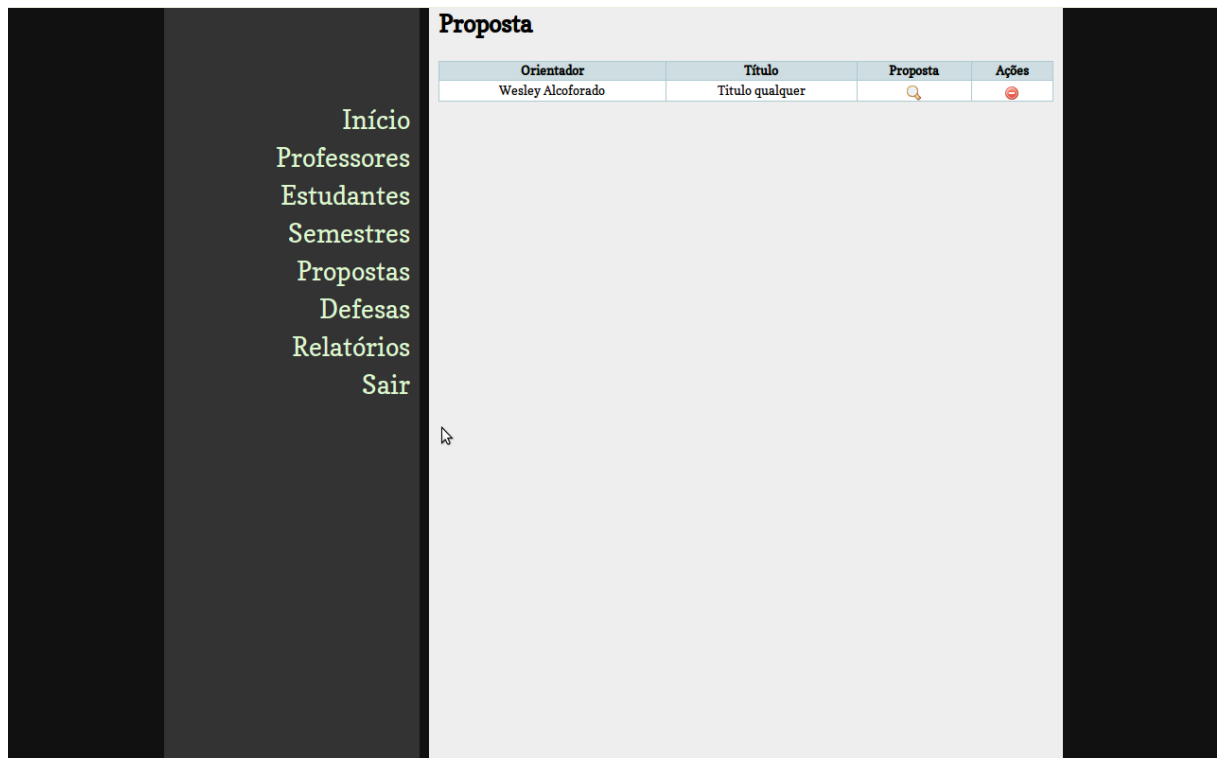


Figura 4.9: Tela de visualização de propostas no perfil do administrador

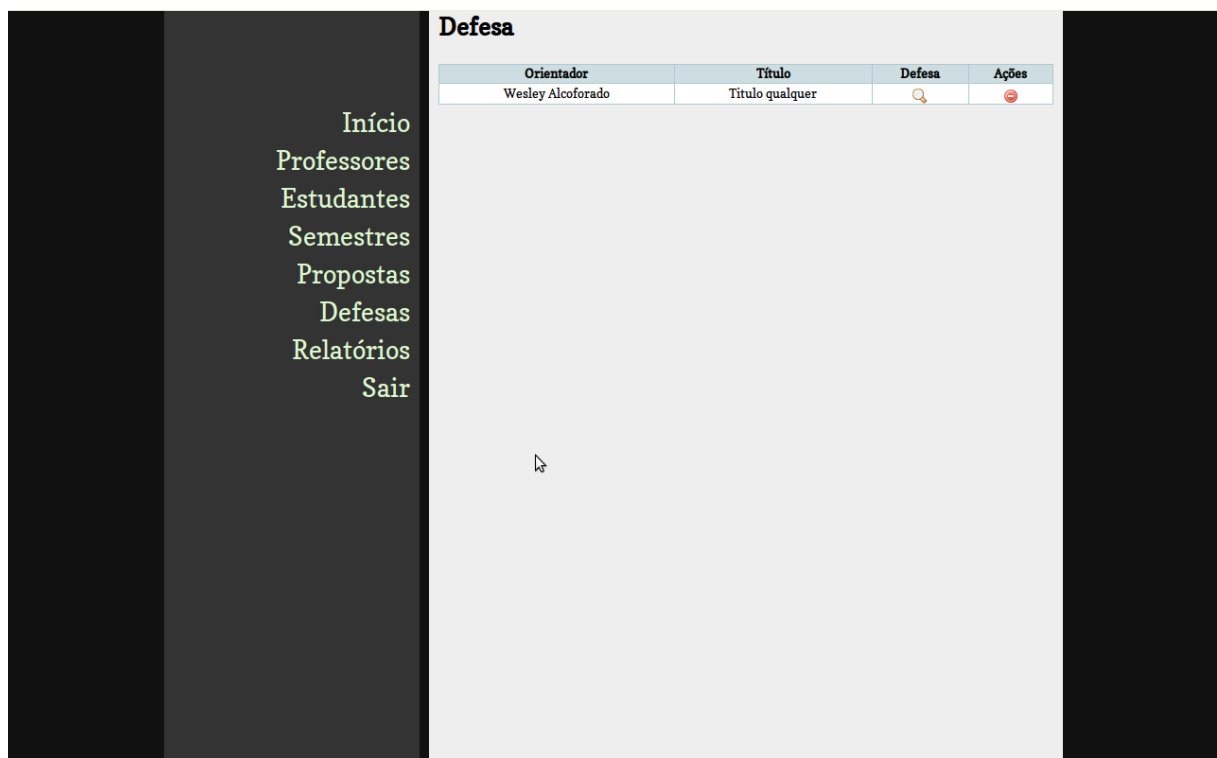


Figura 4.10: Tela de visualização de defesas no perfil do administrador

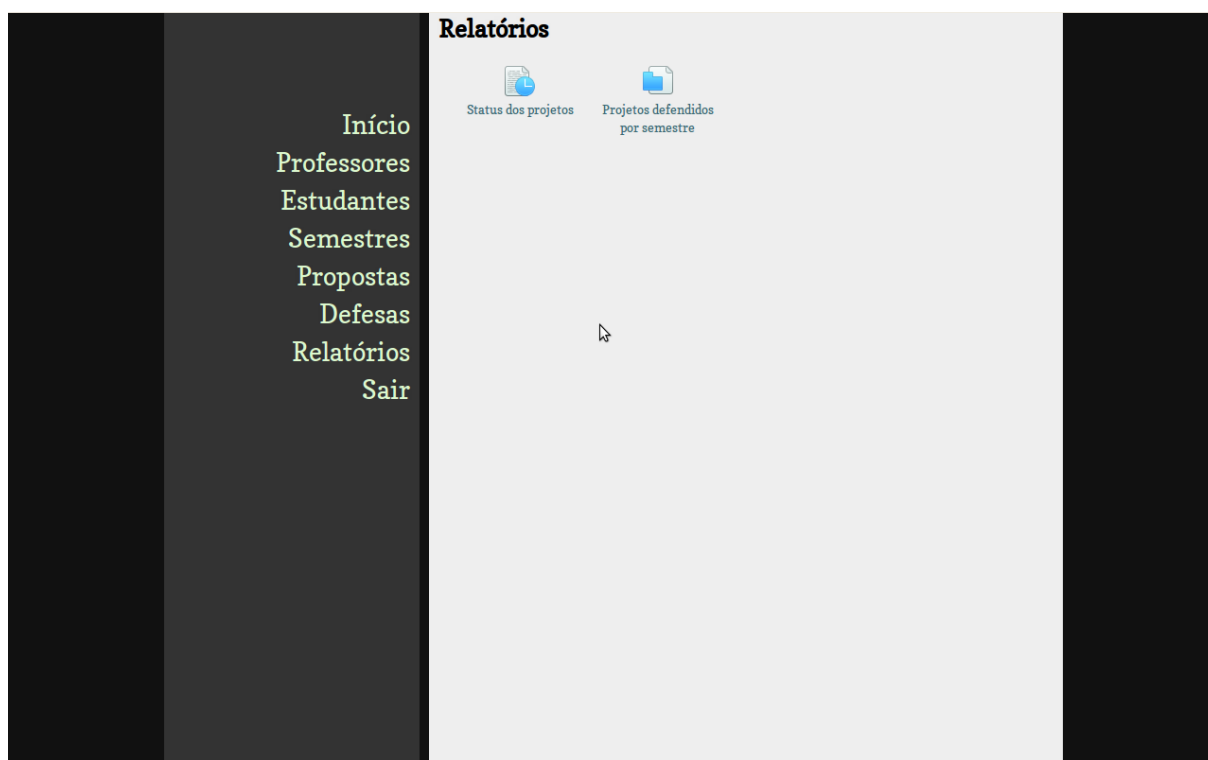


Figura 4.11: Tela de consulta de relatórios

Quando isso acontecer, o estudante deverá receber um email notificando a novidade. As Figuras 4.16 e 4.17 demonstram as modificações do status do projeto à medida que o orientador e a comissão avaliam a sua proposta.

A partir da tela da Figura 4.17, o ícone em formato de apresentação de slides fica disponível para o estudante, significando que ele pode submeter uma solicitação de defesa. Ao clicar no ícone, a tela da Figura 4.18 é carregada, solicitando ao estudante os itens necessários para a solicitação da defesa, como o documento do copião.

As Figuras 4.19, 4.20, 4.21 e 4.22 mostram respectivamente a evolução do status do projeto à medida em que o estudante solicita a defesa, o orientador a aprova, a comissão a aprova e a comissão identifica que a defesa foi efetivamente concluída.

4.1.4 Perfil de professor (orientador)

O orientador possui acesso aos projetos de seus orientandos, e deve dar o seu aval para quaisquer submissões que o estudante envie para a comissão de projeto final.

A Figura 4.23 exibe a tela de listagem de propostas enviadas pelos orientandos. Propostas ainda não avaliadas pelo professor aparecem com dois ícones de polegar para cima e polegar para baixo, que indicam que o professor aprova ou não a proposta, respectivamente. Se o professor clicar no ícone em formato de lupa, ele pode visualizar o documento associado ao pedido.

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores









Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Título qualquer	  	  	Projeto defendido	 

Figura 4.12: Tela de cadastro de projetos

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores













Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Título qualquer	  	  	Projeto defendido	 
Mariela Cortés	Novo projeto			Proposta pendente	 

Figura 4.13: Tela de cadastro de projetos após a inserção de novo item

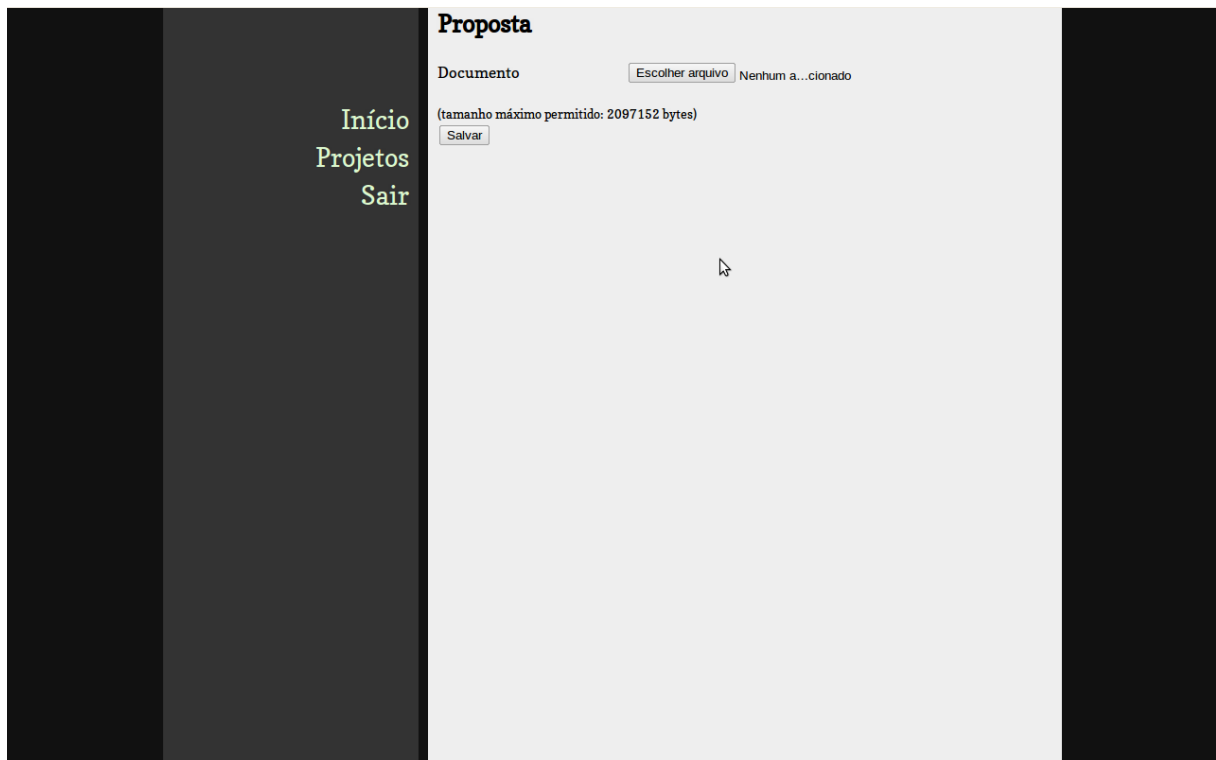


Figura 4.14: Tela de anexo de propostas

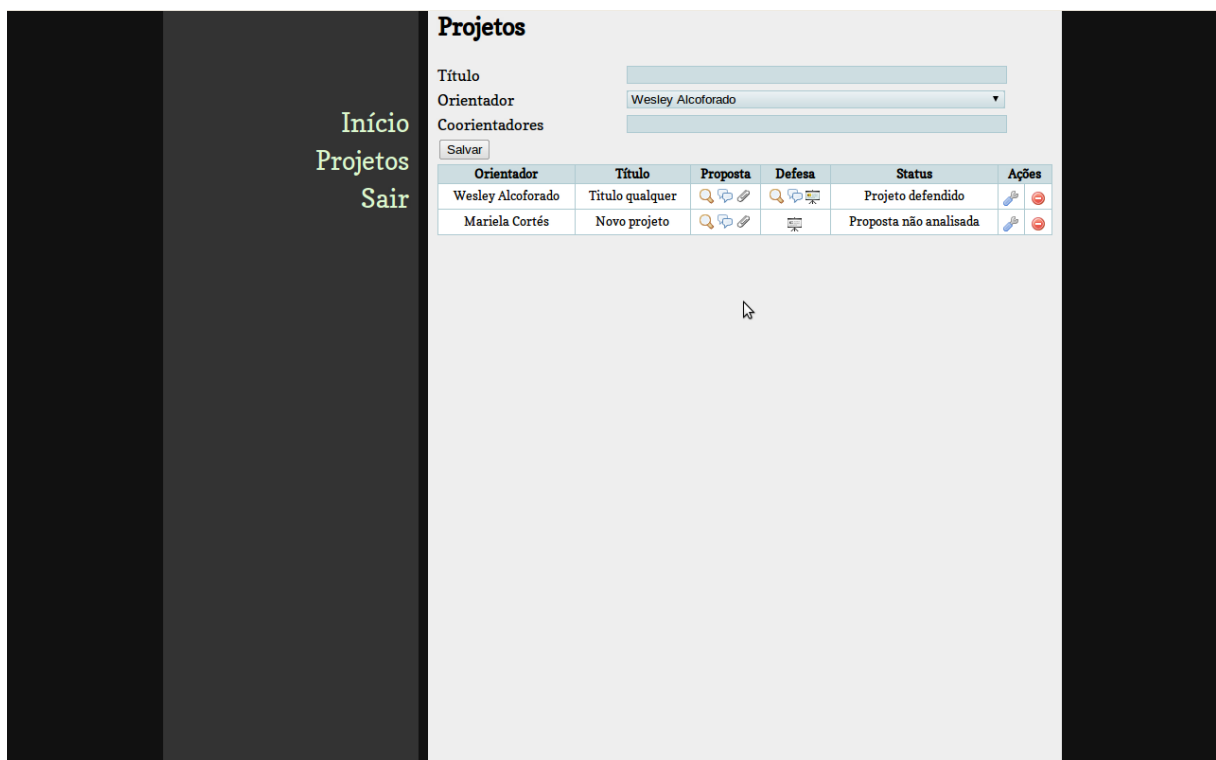


Figura 4.15: Tela de cadastro de projetos após a anexação de uma proposta

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores

Salvar

Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Titulo qualquer			Projeto defendido	
Mariela Cortés	Novo projeto			Proposta aprovada pelo orientador	

Figura 4.16: Tela de cadastro de projetos após a aprovação do orientador

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores

Salvar

Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Titulo qualquer			Projeto defendido	
Mariela Cortés	Novo projeto			Proposta aprovada pela comissão	

Figura 4.17: Tela de cadastro de projetos após a aprovação da comissão

Início
Projetos
Sair

Defesas

Data sugerida: 31 / 03 / 2011

Documento: Escolher arquivo 15713

Qtde. de páginas: 55

(tamanho máximo permitido: 2097152 bytes)

Salvar

MARCH 2011

<<	<	TODAY							>	>>
mon	tue	wed	thu	fri	sat	sun				
	1	2	3	4	5	6				
7	8	9	10	11	12	13				
14	15	16	17	18	19	20				
21	22	23	24	25	26	27				
28	29	30	31							

March 31, 2011

localhost/monom/frontend_dev.php/projeto/5/defesa#defesa_data_sugestao_year

Figura 4.18: Tela de solicitação de defesa

Início
Projetos
Sair

Projetos

Título:

Orientador: Wesley Alcoforado

Coorientadores:

Salvar













Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Titulo qualquer	 	 	Projeto defendido	 
Mariela Cortés	Novo projeto	 	 	Defesa não analisada	 

Figura 4.19: Tela de cadastro de projetos após a solicitação de defesa

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores

Salvar

Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Título qualquer			Projeto defendido	
Mariela Cortés	Novo projeto			Defesa aprovada pelo orientador	

Figura 4.20: Tela de cadastro de projetos após a aprovação do orientador

Início
Projetos
Sair

Projetos

Título

Orientador

Coorientadores

Salvar

Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Título qualquer			Projeto defendido	
Mariela Cortés	Novo projeto			Defesa aprovada pela comissão	

Figura 4.21: Tela de cadastro de projetos após a aprovação da comissão

Projetos

Título

Orientador

Coorientadores

Orientador	Título	Proposta	Defesa	Status	Ações
Wesley Alcoforado	Título qualquer			Projeto defendido	
Mariela Cortés	Novo projeto			Projeto defendido	

Figura 4.22: Tela de cadastro de projetos após a comissão ter indicado que a defesa foi concluída

A Figura 4.24 mostra o momento em que o professor clica na opção de aprovar a proposta e a Figura 4.25 o momento seguinte a esta ação.

Para o perfil de professor, o módulo de Defesas é completamente idêntico ao módulo de propostas, o que podemos perceber pela Figura 4.26.

4.1.5 Perfil de comissão

A comissão precisa dar o parecer final em qualquer submissão feito pelo estudante. É ela quem decide se uma proposta de projeto pode ser desenvolvida ou não, da mesma forma que é ela quem decide se o estudante já pode defender seu projeto.

A Figura 4.27 exibe a tela de listagem de propostas e uma coluna de ações. Na primeira subcoluna, há um ícone em formato de lupa que permite à comissão visualizar o documento da proposta anexada. O segundo ícone, em formato de balões de conversação permite a um integrante da comissão saber quais foram os comentários feitos pelos outros integrantes sobre a proposta em questão. O último ícone assume o formato de martelo de decisão quando a proposta ainda não foi analisada e assume o formato de polegar para cima ou para baixo dependendo da avaliação final. Ao clicar no ícone do martelo, a tela da Figura 4.28 é exibida, permitindo ao integrante da comissão dar seu parecer e fornecer um comentário opcional sobre a proposta. Uma tela de confirmação é exibida quando o integrante seleciona as opções Aprovar ou Desaprovar, como pode ser observado na mesma figura.

A Figura 4.29 exibe o status da listagem após a aprovação da proposta e a Figura 4.30

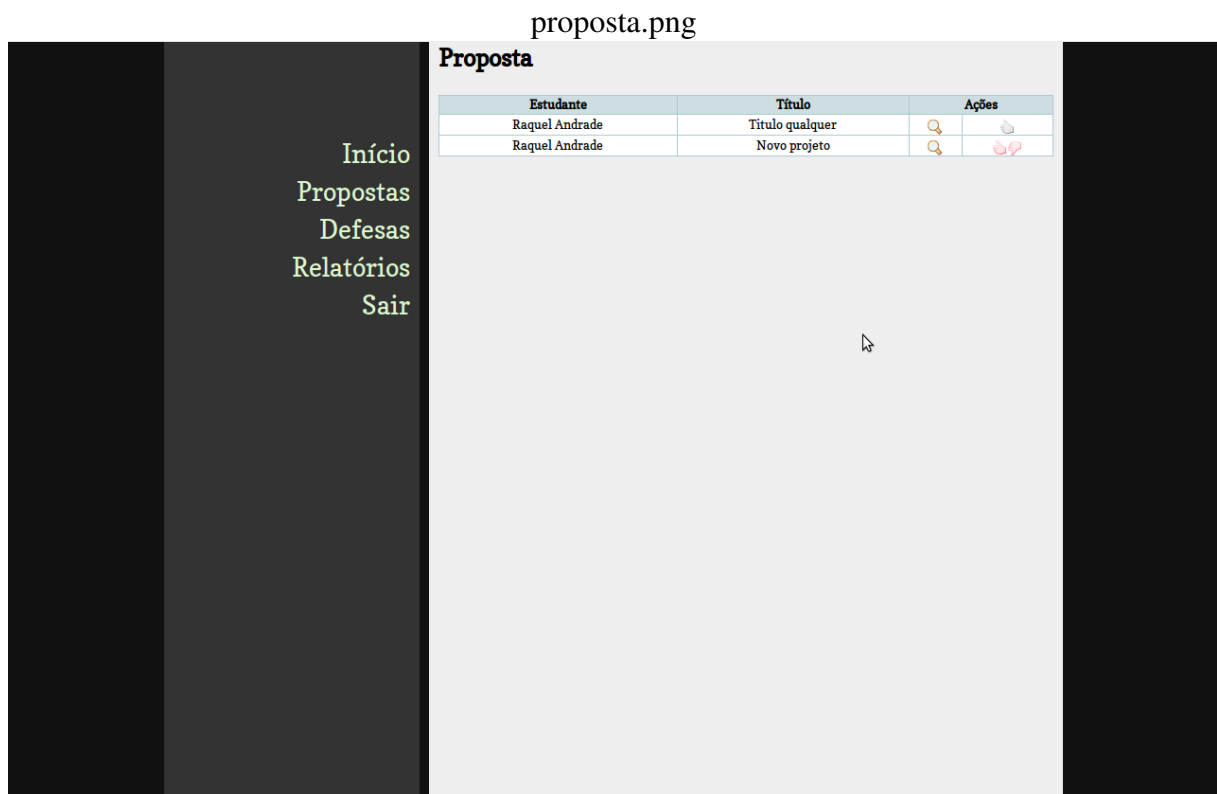


Figura 4.23: Tela de listagem de propostas submetidas pelos orientandos

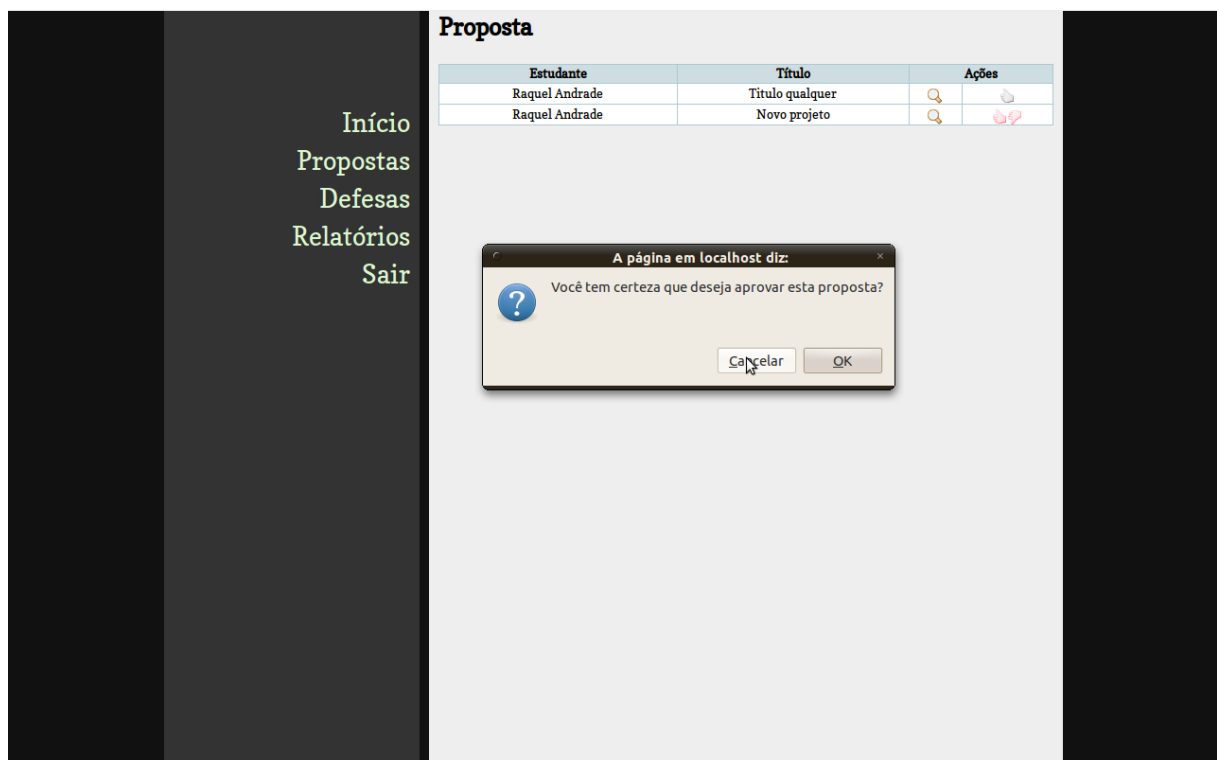


Figura 4.24: Proposta sendo aprovada pelo orientador

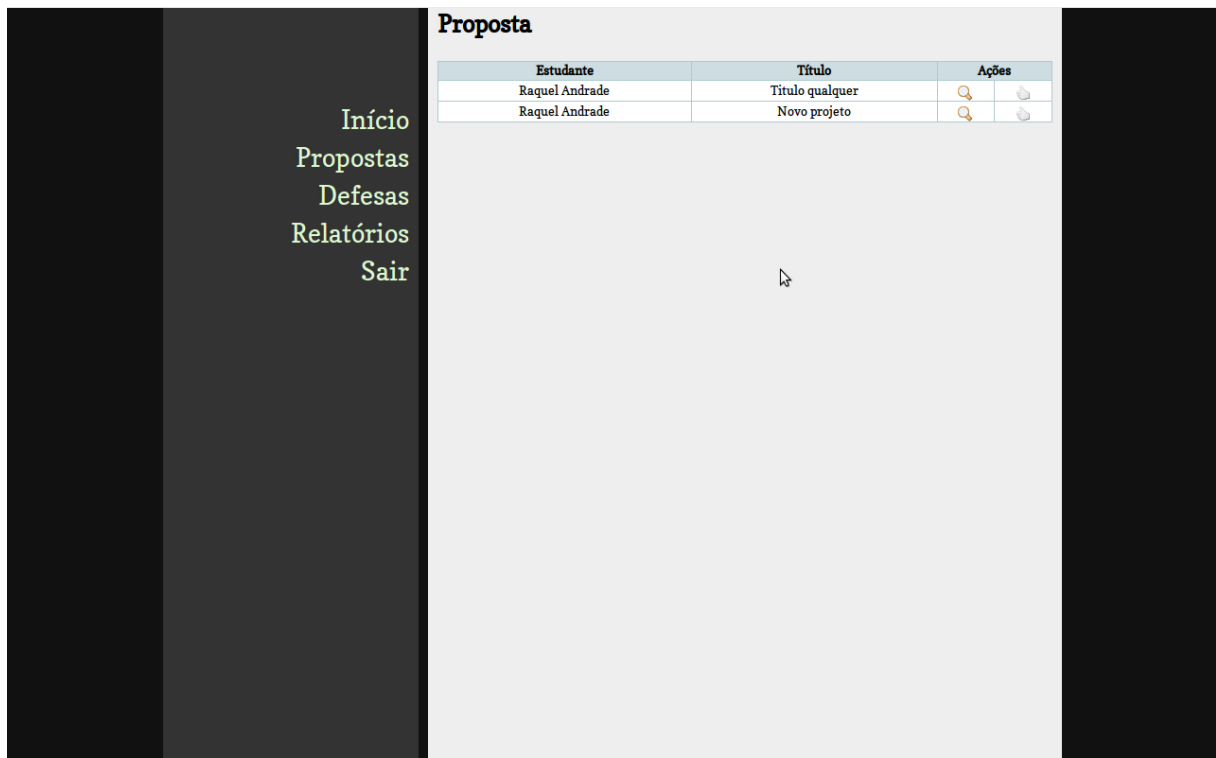


Figura 4.25: Tela de listagem de propostas após o orientador ter aprovado uma proposta

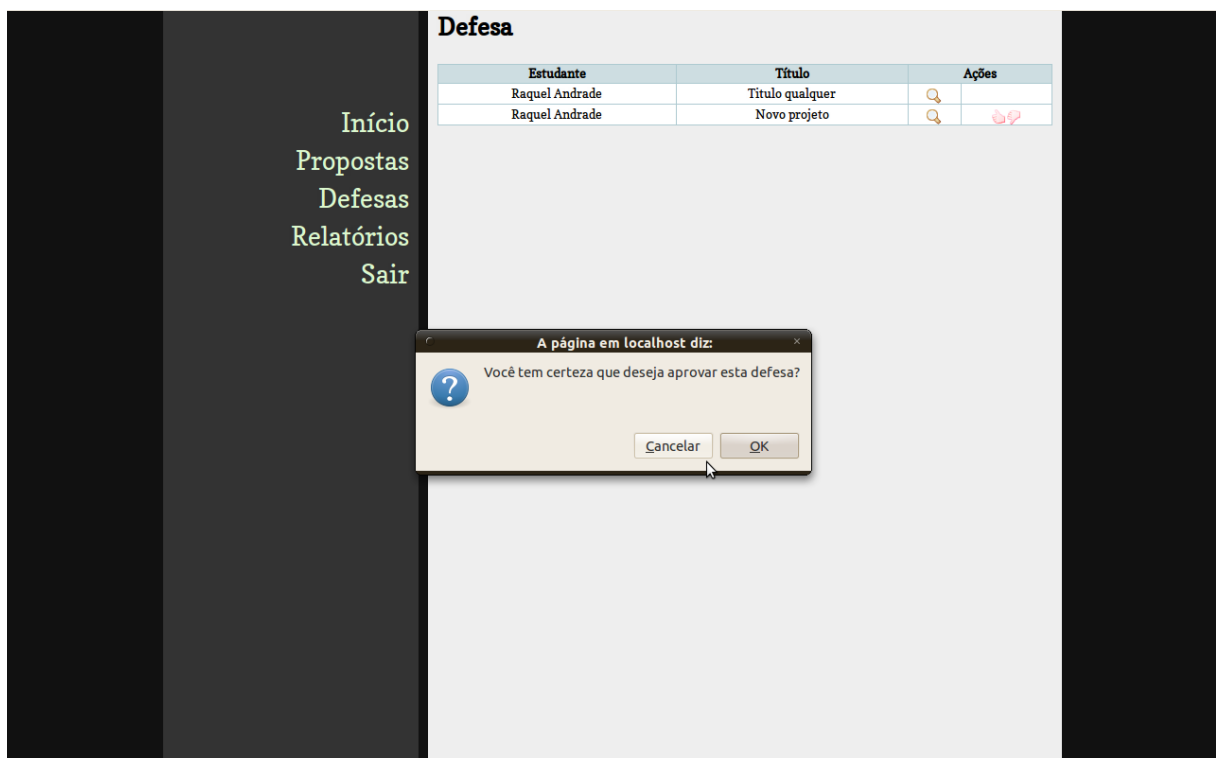
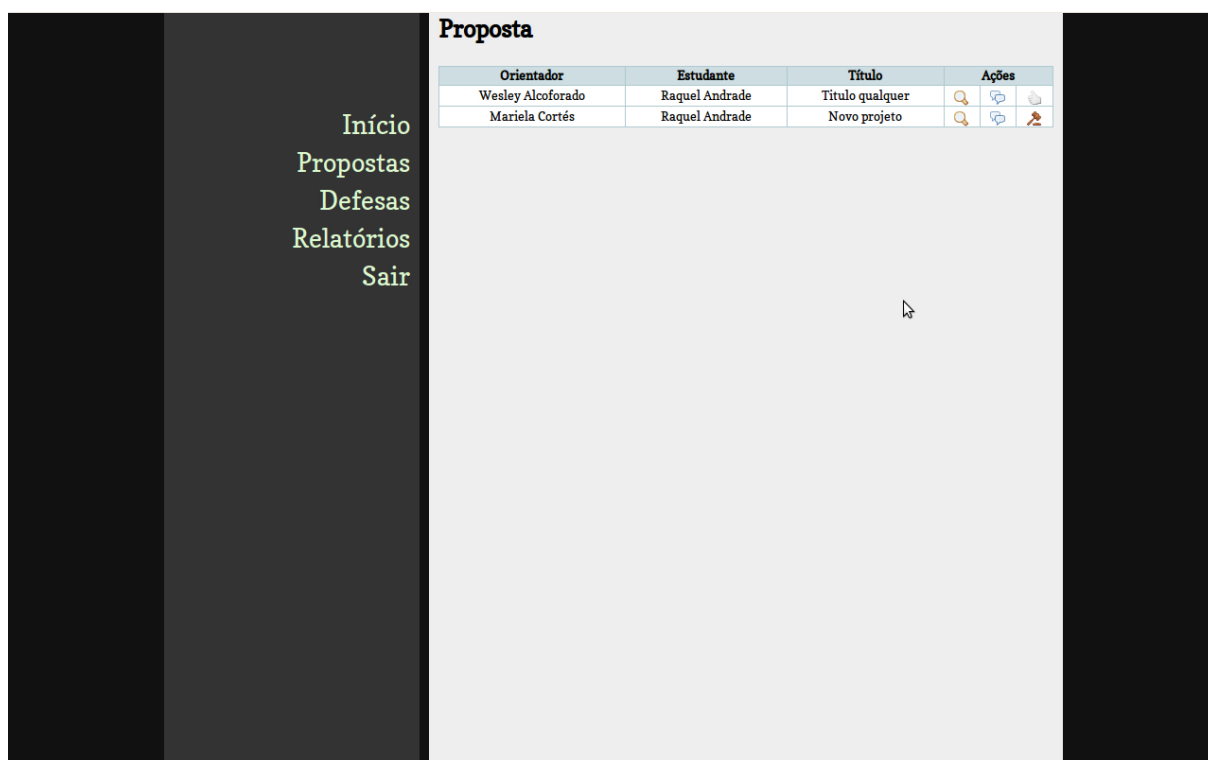


Figura 4.26: Tela de listagem de requisições de defesa submetidas pelos orientandos

exibe a tela de comentários, que surge após o integrante da comissão clicar no ícone em formato de balões de conversação.

Para a avaliação das solicitações de defesa, o esquema é idêntico ao das propostas, como pode ser observado na Figura 4.31. Existe uma pequena diferença no ícone do martelo de decisão após a comissão aprovar a solicitação de defesa, como pode ser visto na Figura 4.32: o ícone é substituído por um marcador verde, que deve ser clicado para que a comissão possa confirmar que o estudante efetivamente defendeu o TCC e todo o seu processo foi concluído. Após a marcação deste ícone, ele fica cinza e desabilitado.









Orientador	Estudante	Título	Ações
Wesley Alcoforado	Raquel Andrade	Titulo qualquer	  
Mariela Cortés	Raquel Andrade	Novo projeto	  

Figura 4.27: Tela de listagem de propostas no perfil da comissão

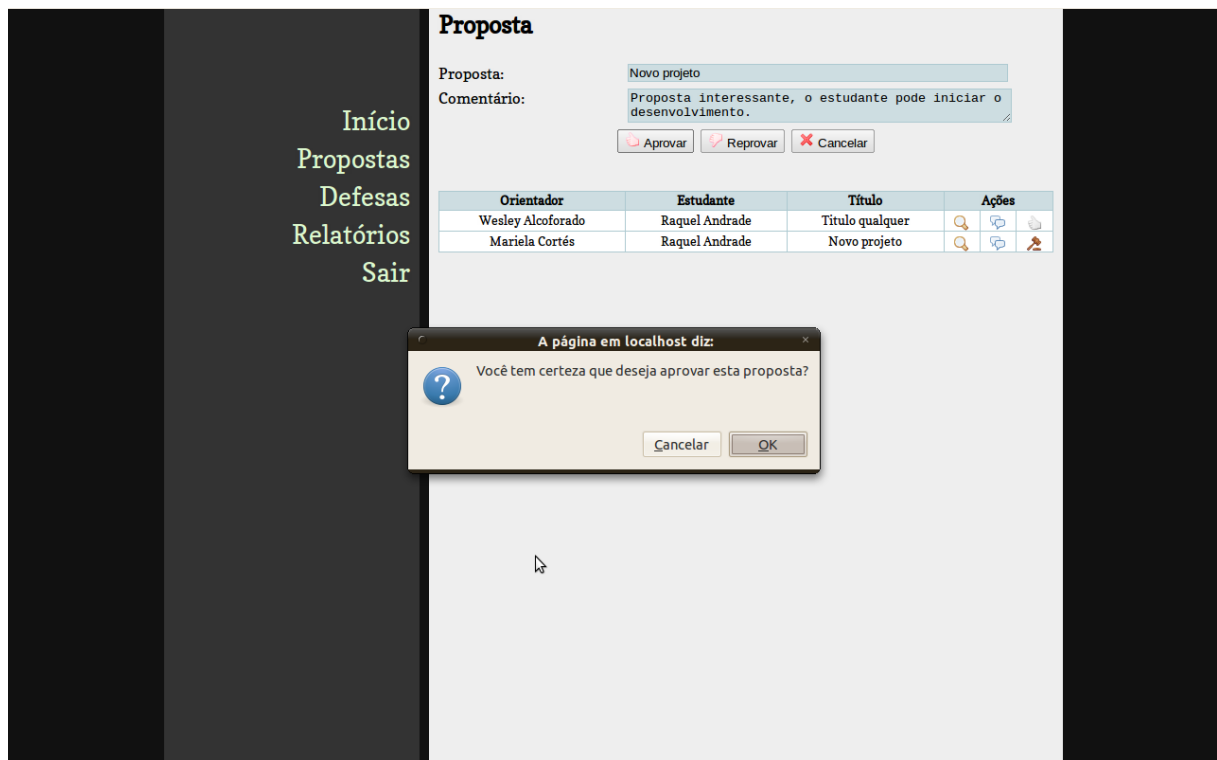


Figura 4.28: Proposta sendo avaliada pela comissão

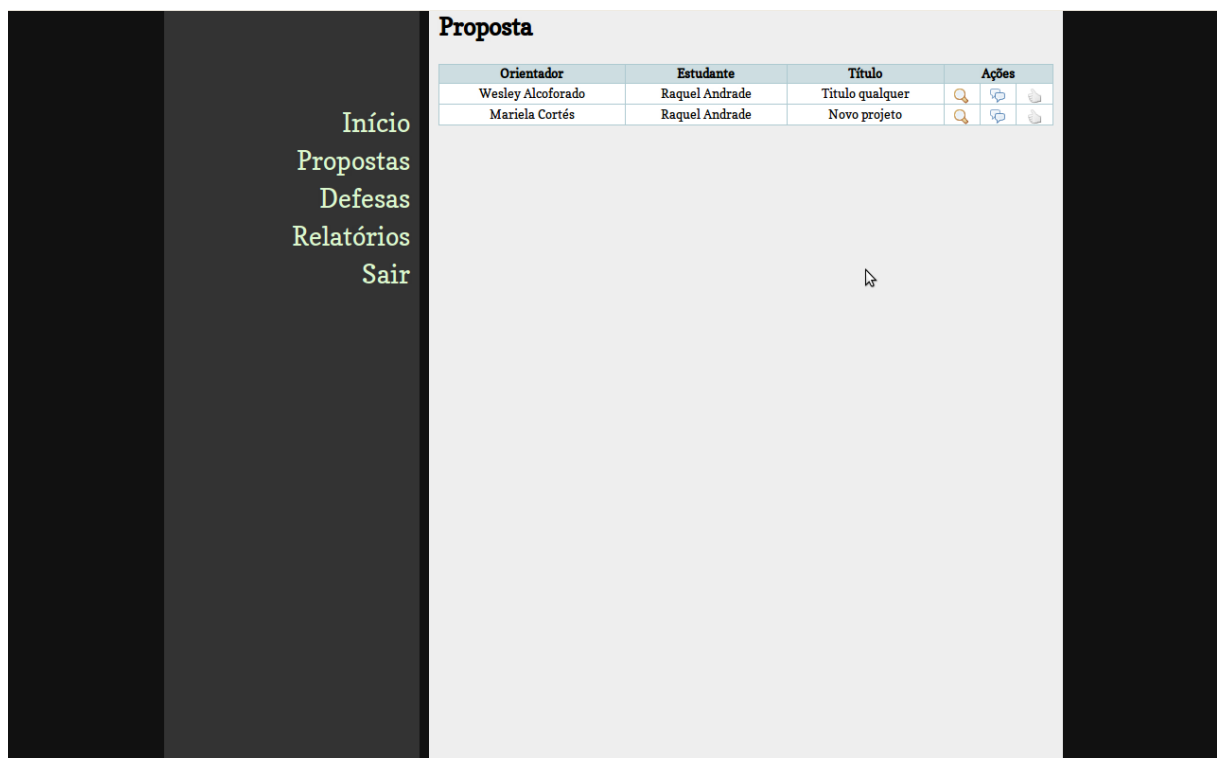


Figura 4.29: Tela de listagem de propostas após a comissão ter avaliado positivamente uma proposta

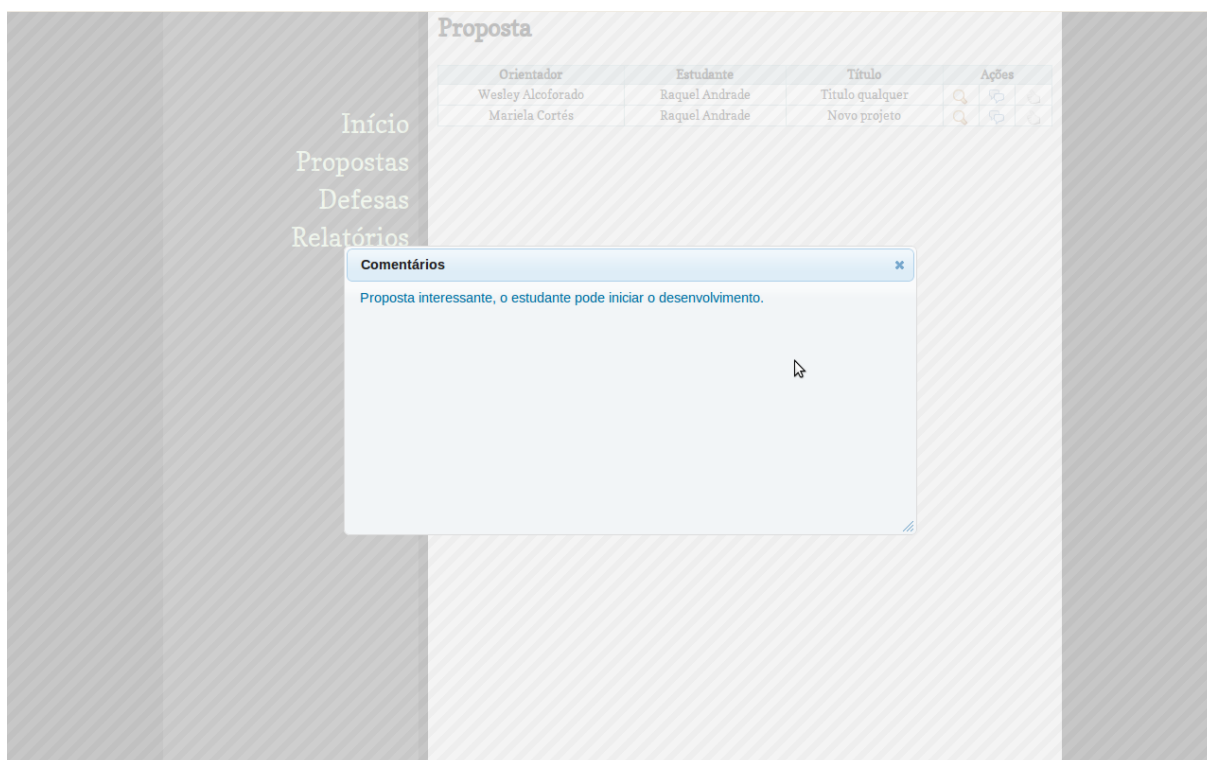


Figura 4.30: Tela de listagem de requisições de defesa submetidas pelos orientandos

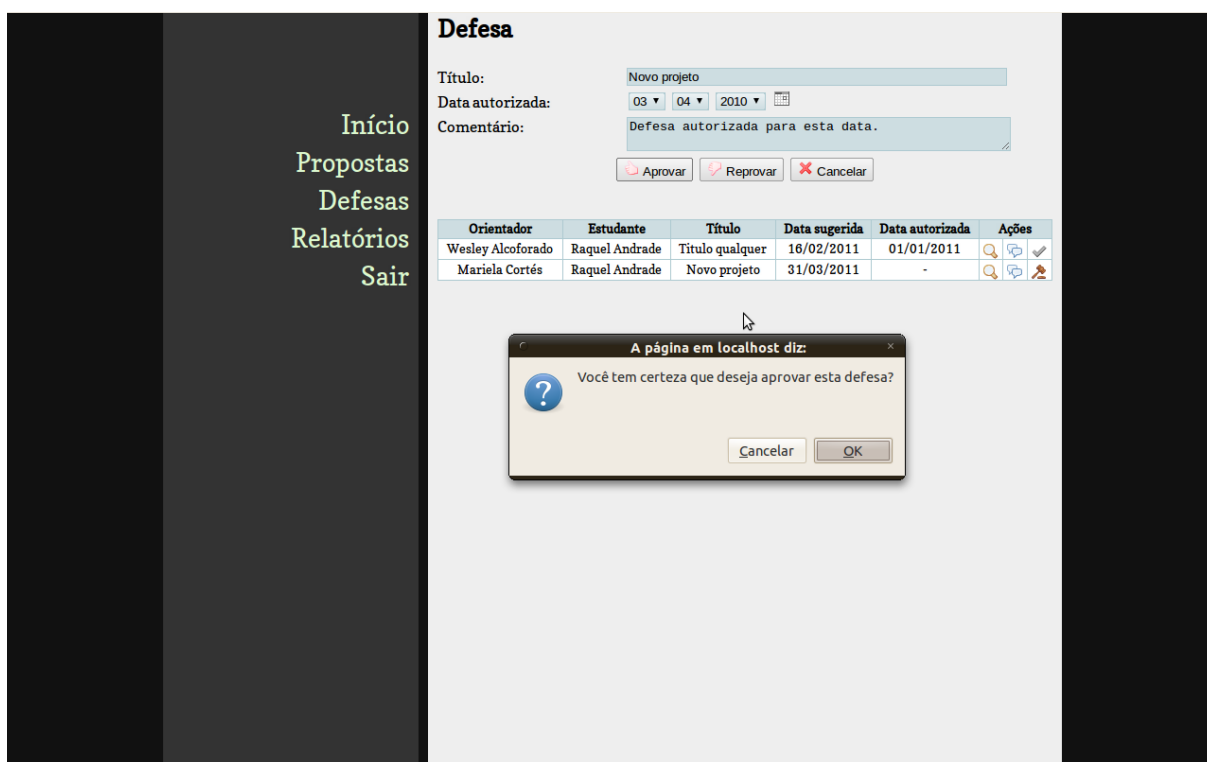


Figura 4.31: Tela de listagem de requisições de defesa no perfil da comissão

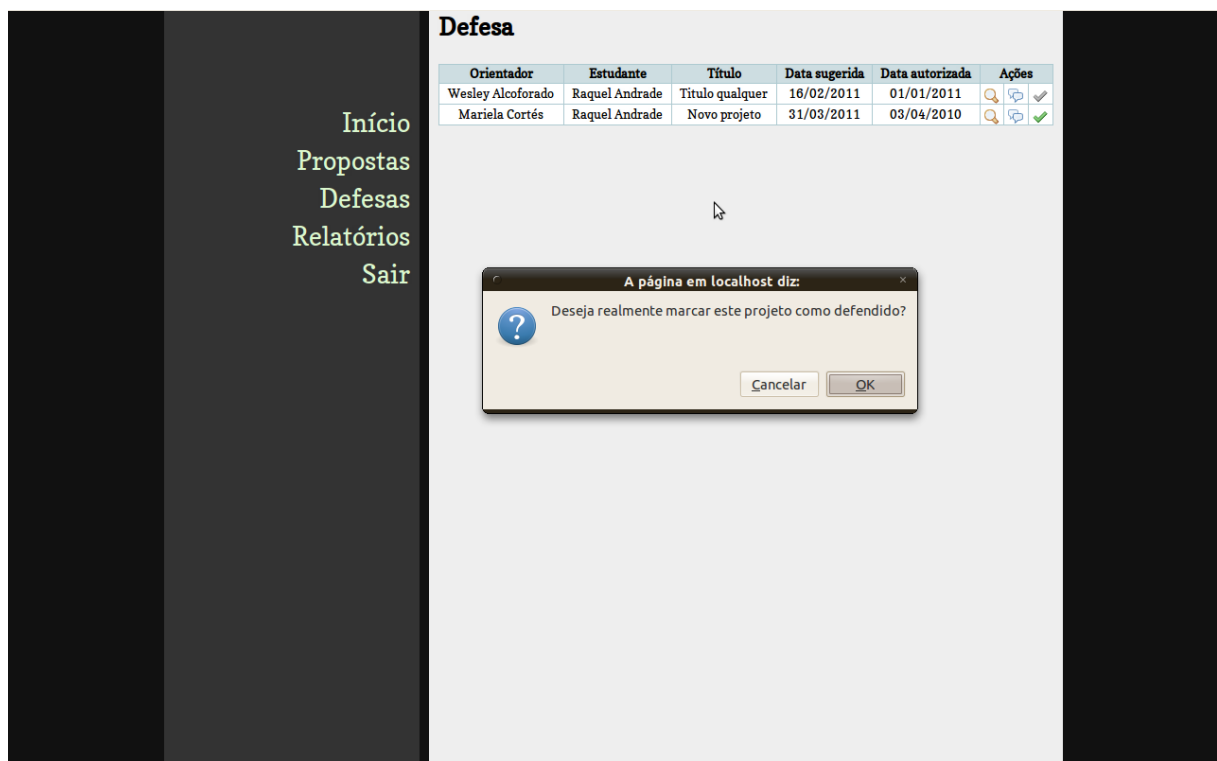


Figura 4.32: Comissão marcando projeto como defendido

5 CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi apresentada uma solução para um problema comum do dia-a-dia da coordenação do curso de Ciências da Computação da UECE - ter que lidar com muita papelada referente aos TCCs dos alunos.

A ferramenta se propõe a gerenciar o processo de submissão de projetos finais, seguindo o regulamento estabelecido pela universidade, e também a manter todos atentos ao cumprimento dos prazos.

Pessoalmente, espero que a aplicação desenvolvida neste trabalho seja útil para o curso de Ciências da Computação da UECE, público alvo desta monografia, e que eu tenha conseguido dar minha humilde contribuição para o nosso curso, de onde tirei a maior parte do meu conhecimento em computação.

5.1 Trabalhos futuros

A partir da contribuição apresentada pelo presente trabalho, alguns trabalhos futuros podem ser indicados visando estender sua funcionalidade como:

- Construir um sistema de visualização e pesquisa das monografias defendidas, com exportação de citações para diferentes formatos, como o $\text{BIB}\text{T}_{\text{E}}\text{X}$.
- Estender o funcionamento do sistema para também atender defesas de mestrado e doutorado.
- Generalizar a ferramenta para atender quaisquer tipo de defesa de qualquer curso universitário.

REFERÊNCIAS BIBLIOGRÁFICAS

CABIBBO, L.; CAROSI, A. Managing inheritance hierarchies in object/relational mapping tools. In: PASTOR, O.; CUNHA, J. Falcão e (Ed.). *Advanced Information Systems Engineering*. Springer Berlin / Heidelberg, 2005, (Lecture Notes in Computer Science, v. 3520). p. 93–124. Disponível em: <http://dx.doi.org/10.1007/11431855_11>.

CHEN, N. *Convention over Configuration*. 2006. Acesso em 24 de fevereiro de 2011. Disponível em: <<http://softwareengineering.vazexqi.com/files/pattern.html>>.

FOWLER, M. *Padrões de Arquitetura de Aplicações Corporativas*. São Paulo: Bookman, 2006.

FREITAS, H. *Informação e decisão: sistemas de apoio e seu impacto*. Porto Alegre: Ortiz, 1997.

INFOQ. *Facebook: Science and the Social Graph*. 2009. Acesso em 24 de fevereiro de 2011. Disponível em: <<http://www.infoq.com/presentations/Facebook-Software-Stack>>.

MINETTO, E. L. *Frameworks para Desenvolvimento em PHP*. Chapecó - SC: Novatec, 2007.

PHP.NET. *Manual do PHP*. 2011. Acesso em 24 de fevereiro de 2011. Disponível em: <http://www.php.net/manual/pt_BR>.

POSTGRESQL. *Introdução e Histórico*. 2003. Acesso em 25 de fevereiro de 2011. Disponível em: <http://wiki.postgresql.org/wiki/Introdu%C3%A7%C3%A3o_e_Hist%C3%B3rico>.

SOMMERVILLE, I. *Engenharia de Software*. São Paulo: Addison Wesley, 2003.

SYMFONY. *Plugins: sfGuardPlugin*. 2010. Acesso em 27 de fevereiro de 2011. Disponível em: <<http://www.symfony-project.org/plugins/sfGuardPlugin>>.

WIKIPEDIA. *PHP*. 2011. Acesso em 24 de fevereiro de 2011. Disponível em: <<http://en.wikipedia.org/wiki/PHP>>.

WIKIPEDIA. *Wikipedia:FAQ/Technical - What software is used to run Wikipedia?* 2011. Acesso em 24 de fevereiro de 2011. Disponível em: <http://en.wikipedia.org/wiki/Wikipedia:FAQ/Technical#What_software_is_used_to_run_Wikipedia.3F>.

WORDPRESS. *About Wordpress*. 2011. Acesso em 24 de fevereiro de 2011. Disponível em: <<http://wordpress.org/about/>>.

6 APÊNDICE

6.1 Casos de uso

6.1.1 Manter Professores

Atores	Administrador
Pré-condições	O administrador deve estar logado no sistema.
Fluxo básico	<ol style="list-style-type: none"> 1. O caso de uso se inicia quando o administrador seleciona manter professores no menu do sistema. 2. Uma vez que o administrador seleciona uma das opções disponíveis (incluir, alterar, excluir, listar): <ol style="list-style-type: none"> a) Se o administrador selecionar a opção incluir, o caso de uso segue para o sub-fluxo 2 - Incluir Professor. b) Se o administrador selecionar a opção alterar, o caso de uso segue para o sub-fluxo 3 - Alterar Professor. c) Se o administrador selecionar a opção excluir, o caso de uso segue para o sub-fluxo 4 - Excluir Professor. d) Se o administrador selecionar a opção listar, o caso de uso segue para o sub-fluxo 5 - Listar Professores. 3. O caso de uso se encerra.
Incluir Professor	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona incluir um novo professor. 2. O sistema exibe os seguintes campos (os campos com asterisco são obrigatórios): <ul style="list-style-type: none"> * Nome de usuário * Email * Senha * Confirmação de senha Nome Sobrenome * Instituição * Titulação Experiência Substituto - Campo de escolha única fechada (valores: sim, não) Comissão - Campo de escolha única fechada (valores: sim, não) Ativo - Campo de escolha única fechada (valores: sim, não) Superusuário - Campo de escolha única fechada (valores: sim, não) 3. O administrador preenche os campos e seleciona a opção salvar. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema inclui o professor no banco de dados. 6. O caso de uso se encerra.

Alterar Professor	<p>Pré-condições: O administrador deve ter selecionado um professor para a alteração.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona alterar professor. 2. O sistema exibe os campos preenchidos. 3. O administrador altera os dados e solicita salvar os dados. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema salva as alterações no banco de dados. 6. O caso de uso se encerra.
Excluir Professor	<p>Pré-condições: O administrador deve ter selecionado um professor para a exclusão.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona excluir professor. 2. O sistema solicita que o administrador confirme a exclusão. 3. O administrador confirma a mensagem. 4. O sistema exclui o professor do banco de dados. 5. O caso de uso se encerra.
Listar Professores	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona listar professores. 2. O sistema exibe a listagem dos professores, contendo os seguintes campos: <ol style="list-style-type: none"> a) Nome b) Sobrenome c) Nome de usuário d) Email 3. O caso de uso se encerra.
Fluxos alternativos	<p>Dados obrigatórios não preenchidos</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia no passo 4 dos sub-fluxos Incluir Professor e Alterar Professor, quando o usuário não informou todos os campos obrigatórios. 2. O sistema exibe ao lado do campo uma mensagem de que o campo deve ser preenchido e aguarda até que o administrador o preencha. 3. O subfluxo segue para o passo 2 do sub-fluxo do qual ele se originou.

6.1.2 Manter Estudantes

Atores	Administrador
Pré-condições	O administrador deve estar logado no sistema.
Fluxo básico	<ol style="list-style-type: none"> 1. O caso de uso se inicia quando o administrador seleciona manter estudantes no menu do sistema. 2. Uma vez que o administrador seleciona uma das opções disponíveis (incluir, alterar, excluir, listar): <ol style="list-style-type: none"> a) Se o administrador selecionar a opção incluir, o caso de uso segue para o sub-fluxo 2 - Incluir Estudante. b) Se o administrador selecionar a opção alterar, o caso de uso segue para o sub-fluxo 3 - Alterar Estudante. c) Se o administrador selecionar a opção excluir, o caso de uso segue para o sub-fluxo 4 - Excluir Estudante. d) Se o administrador selecionar a opção listar, o caso de uso segue para o sub-fluxo 5 - Listar Estudantes. 3. O caso de uso se encerra.
Incluir Estudante	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona incluir um novo estudante. 2. O sistema exibe os seguintes campos (os campos com asterisco são obrigatórios): <ul style="list-style-type: none"> * Matrícula * Email * Senha * Confirmação de senha Nome Sobrenome Telefone Ativo - Campo de escolha única fechada (valores: sim, não) 3. O administrador preenche os campos e seleciona a opção salvar. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema inclui o estudante no banco de dados. 6. O caso de uso se encerra.

Alterar Estudante	<p>Pré-condições: O administrador deve ter selecionado um estudante para a alteração.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona alterar estudante. 2. O sistema exibe os campos preenchidos. 3. O administrador altera os dados e solicita salvar os dados. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema salva as alterações no banco de dados. 6. O caso de uso se encerra.
Excluir Estudante	<p>Pré-condições: O administrador deve ter selecionado um estudante para a exclusão.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona excluir estudante. 2. O sistema solicita que o administrador confirme a exclusão. 3. O administrador confirma a mensagem. 4. O sistema exclui o estudante do banco de dados. 5. O caso de uso se encerra.
Listar Estudantes	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o administrador seleciona listar estudantes. 2. O sistema exibe a listagem dos estudantes, contendo os seguintes campos: <ol style="list-style-type: none"> a) Nome b) Sobrenome c) Matrícula d) Telefone e) Email 3. O caso de uso se encerra.
Fluxos alternativos	<p>Dados obrigatórios não preenchidos</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia no passo 4 dos sub-fluxos Incluir Estudante e Alterar Estudante, quando o usuário não informou todos os campos obrigatórios. 2. O sistema exibe ao lado do campo uma mensagem de que o campo deve ser preenchido e aguarda até que o administrador o preencha. 3. O subfluxo segue para o passo 2 do sub-fluxo do qual ele se originou.

6.1.3 Manter Projetos

Atores	Estudante
Pré-condições	O estudante deve estar logado no sistema.
Fluxo básico	<ol style="list-style-type: none"> 1. O caso de uso se inicia quando o estudante seleciona manter projetos no menu do sistema. 2. Uma vez que o estudante seleciona uma das opções disponíveis (incluir, alterar, excluir, listar, visualizar comentários): <ol style="list-style-type: none"> a) Se o estudante selecionar a opção incluir, o caso de uso segue para o sub-fluxo 2 - Incluir Projeto. b) Se o estudante selecionar a opção alterar, o caso de uso segue para o sub-fluxo 3 - Alterar Projeto. c) Se o estudante selecionar a opção excluir, o caso de uso segue para o sub-fluxo 4 - Excluir Projeto. d) Se o estudante selecionar a opção listar, o caso de uso segue para o sub-fluxo 5 - Listar Projetos. e) Se o estudante selecionar a opção visualizar comentários, o caso de uso segue para o sub-fluxo 6 - Visualizar Comentários. 3. O caso de uso se encerra.
Incluir Projeto	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o estudante seleciona incluir um novo projeto. 2. O sistema exibe os seguintes campos (os campos com asterisco são obrigatórios): <ol style="list-style-type: none"> * Título * Orientador * Coorientadores 3. O estudante preenche os campos e seleciona a opção salvar. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema inclui o projeto no banco de dados. 6. O caso de uso se encerra.
Alterar Projeto	<p>Pré-condições: O estudante deve ter selecionado um projeto para a alteração.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o estudante seleciona alterar projeto. 2. O sistema exibe os campos preenchidos. 3. O estudante altera os dados e solicita salvar os dados. 4. O sistema valida se os campos obrigatórios foram preenchidos. 5. O sistema salva as alterações no banco de dados. 6. O caso de uso se encerra.

Excluir Projeto	<p>Pré-condições: O estudante deve ter selecionado um projeto para a exclusão.</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o estudante seleciona excluir projeto. 2. O sistema solicita que o projeto confirme a exclusão. 3. O estudante confirma a mensagem. 4. O sistema exclui o projeto do banco de dados. 5. O caso de uso se encerra.
Listar Projetos	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o estudante seleciona listar projetos. 2. O sistema exibe a listagem dos projetos, contendo os seguintes campos: <ol style="list-style-type: none"> a) Orientador b) Título c) Proposta d) Defesa e) Status - Status mais atual da defesa, ou se esta não tiver sido iniciada, o status mais atual da proposta. 3. O caso de uso se encerra.
Visualizar Comentários	<ol style="list-style-type: none"> 1. Este sub-fluxo se inicia quando o estudante solicita visualizar os comentários. 2. Uma vez que o estudante seleciona uma das opções disponíveis (comentários do orientador, da comissão): <ol style="list-style-type: none"> a) Se o estudante selecionar visualizar os comentários do orientador, o sistema exibe uma listagem com os comentários do orientador. b) Se o estudante selecionar visualizar os comentários da comissão, o sistema exibe uma listagem com os comentários da comissão. 3. O caso de uso se encerra.
Fluxos alternativos	<p>Dados obrigatórios não preenchidos</p> <ol style="list-style-type: none"> 1. Este sub-fluxo se inicia no passo 4 dos sub-fluxos Incluir Estudante e Alterar Estudante, quando o usuário não informou todos os campos obrigatórios. 2. O sistema exibe ao lado do campo uma mensagem de que o campo deve ser preenchido e aguarda até que o administrador o preencha. 3. O subfluxo segue para o passo 2 do sub-fluxo do qual ele se originou.

6.1.4 Manter Propostas

Atores	Estudante, Orientador, Comissão
Pré-condições	<p>O estudante deve estar logado no sistema e ter selecionado um projeto.</p> <p>O orientador deve estar logado no sistema.</p> <p>A comissão deve estar logada no sistema.</p>
Fluxo básico	<ol style="list-style-type: none"> 1. O caso de uso se inicia quando o estudante seleciona a opção Anexar Proposta para o projeto selecionado. 2. O sistema exibe o campo Documento 3. O estudante escolhe um arquivo PDF e seleciona a opção Salvar. 4. O sistema valida o formato e tamanho do arquivo. 5. O sistema anexa o documento ao projeto, exibe uma mensagem de sucesso e redireciona o estudante à listagem de projetos. 6. O sistema envia um email ao orientador, informando do envio de uma nova proposta por um de seus orientandos. 7. O orientador seleciona a opção Visualiza Proposta, do projeto em questão. 8. O sistema exibe a proposta e solicita ao orientador para que ele selecione uma das opções disponíveis (Aprovar/desaprovar) 9. O orientador seleciona a opção aprovar. 10. O sistema envia um email à comissão, informando do envio de uma nova proposta e atualiza o status da proposta para "Aprovada pelo orientador". 11. A comissão seleciona a opção Visualizar Proposta, do projeto em questão. 12. O sistema exibe a proposta e um campo de comentários e solicita à comissão para que ela selecione uma das opções disponíveis (Aprovar/desaprovar) 13. A comissão comenta (opcionalmente) na proposta e a aprova. 14. O sistema envia um email ao orientador e ao estudante, informando de que a proposta foi aprovada e atualiza o status da proposta para "Aprovada pela comissão". 15. O caso de uso se encerra.

Fluxos alternativos	<p>Formato e/ou tamanho inválidos</p> <ol style="list-style-type: none">1. O subfluxo se inicia no passo 4 do fluxo básico, quando o sistema detecta que o formato e/ou o tamanho do arquivo são inválidos.2. O sistema informa ao usuário dos dados inválidos e solicita-o que os corrija.3. O subfluxo segue para o passo 2 do fluxo básico. <p>Desaprovação da proposta</p> <ol style="list-style-type: none">1. O subfluxo se inicia no passo 9 do fluxo básico, quando o usuário for o orientador, ou no passo 13 do fluxo básico, quando o usuário for da comissão. O usuário selecionou a reprovar.2. O sistema envia um email ao estudante (e ao orientador, caso a proposta tenha sido reprovada pela comissão), informando que sua proposta foi reprovada.3. O sistema atualiza o status da proposta para "Reprovada pelo orientador" ou "Reprovada pela comissão", dependendo de qual usuário tenha reprovado a proposta.4. O caso de uso se encerra.
---------------------	---

6.1.5 Manter Defesas

Atores	Estudante, Orientador, Comissão
Pré-condições	<p>O estudante deve estar logado no sistema e ter selecionado um projeto.</p> <p>O orientador deve estar logado no sistema.</p> <p>A comissão deve estar logada no sistema.</p>
Fluxo básico	<ol style="list-style-type: none"> 1. O caso de uso se inicia quando o estudante seleciona a opção Solicitar Defesa para o projeto selecionado. 2. O sistema exibe os seguintes campos: <ol style="list-style-type: none"> a) Quantidade de páginas da monografia b) Data sugerida para realização da defesa c) Cópia (arquivo PDF) 3. O estudante preenche os campos e seleciona a opção Salvar. 4. O sistema valida os campos. 5. O sistema anexa o cópia ao projeto, exibe uma mensagem de sucesso e redireciona o estudante à listagem de projetos. 6. O sistema envia um email ao orientador, informando do envio da solicitação de defesa por um de seus orientandos. 7. O orientador seleciona a opção Solicitação de Defesa, do projeto em questão. 8. O sistema exibe o cópia e solicita ao orientador para que ele selecione uma das opções disponíveis (Aprovar/desaprovar) 9. O orientador seleciona a opção aprovar. 10. O sistema envia um email à comissão, informando do envio de uma nova solicitação de defesa e atualiza o status do projeto para "Defesa aprovada pelo orientador". 11. A comissão seleciona a opção Solicitação de Defesa, do projeto em questão. 12. O sistema exibe os seguintes campos: <ol style="list-style-type: none"> a) Cópia (arquivo para download) b) Comentários c) Data autorizada para realização da defesa 13. O sistema solicita à comissão para que ela selecione uma das opções disponíveis (Aprovar/Desaprovar) 14. A comissão comenta (opcionalmente) na solicitação, preenche a data de realização da defesa e a aprova. 15. O sistema envia um email ao orientador e ao estudante, informando de que a solicitação foi aprovada e atualiza o status do projeto para "Defesa aprovada pela comissão". 16. O caso de uso se encerra.

Fluxos alternativos	<p>Formato e/ou tamanho inválidos</p> <ol style="list-style-type: none">1. O subfluxo se inicia no passo 4 do fluxo básico, quando o sistema detecta que o formato e/ou o tamanho do arquivo são inválidos.2. O sistema informa ao usuário dos dados inválidos e solicita-o que os corrija.3. O subfluxo segue para o passo 2 do fluxo básico. <p>Desaprovação da proposta</p> <ol style="list-style-type: none">1. O subfluxo se inicia no passo 9 do fluxo básico, quando o usuário for o orientador, ou no passo 13 do fluxo básico, quando o usuário for da comissão. O usuário selecionou a reprovar.2. O sistema envia um email ao estudante (e ao orientador, caso a proposta tenha sido reprovada pela comissão), informando que sua proposta foi reprovada.3. O sistema atualiza o status da proposta para "Reprovada pelo orientador" ou "Reprovada pela comissão", dependendo de qual usuário tenha reprovado a proposta.4. O caso de uso se encerra.
---------------------	---

7 ANEXOS

7.1 Formulário de proposta de projeto final



UNIVERSIDADE ESTADUAL DO CEARÁ
CENTRO DE CIÊNCIAS E TECNOLOGIA
CURSO BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

PROPOSTA DE PROJETO FINAL

Nome : _____
 Matrícula: _____
 Tel: (____) _____ E-mail: _____
 Orientador: _____
 Título do Trabalho: _____

 Local e Data

 Assinatura do Aluno

 Assinatura do Orientador

Parecer da Comissão de Projeto Final

☐ Proposta Aprovada ☐ Proposta Reprovada

Comentários: _____

 Local e Data

 Nome:

 Nome:

 Nome:

IMPORTANTE: Anexar Proposta de Projeto Final conforme Modelo

7.2 Formulário de solicitação de defesa e banca



UNIVERSIDADE ESTADUAL DO CEARÁ
CENTRO DE CIÊNCIAS E TECNOLOGIA
CURSO BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

**SOLICITAÇÃO DE DEFESA E INDICAÇÃO DA COMISSÃO
EXAMINADORA DE PROJETO FINAL**

Nome : _____

Matrícula: _____

Tel: (____) _____ E-mail: _____

Orientador: _____

Título do Trabalho: _____

Quantidade de páginas da monografia: _____

Indicação da Comissão Examinadora:

1)Nome: _____

Instituição: _____

Titulação: _____ Experiência: _____ anos

2)Nome: _____

Instituição: _____

Titulação: _____ Experiência: _____ anos

Data sugerida para realização da defesa: ____ / ____ / ____ : ____ horas

Local e Data

Assinatura do Aluno

Assinatura do Orientador

IMPORTANTE: Anexar cópião do texto de monografia.