

**Instituto Federal de Educação, Ciência e Tecnologia   
do Norte de Minas Gerais - Campus Januária**

**Proposta de Especificação de Software**

**Sistema para Gerenciamento do Site do TADS**

**SisTADS**

**Acadêmicos: André Fellype Matos, Erika Maria dos Santos, Vinícius Dias de Souza e Mauro.**

**Januária/MG**

**Abril 2015**



**Instituto Federal de Educação, Ciência e Tecnologia   
do Norte de Minas Gerais - Campus Januária**

**Proposta de Especificação de Software**

**Sistema para Gerenciamento do Site do TADS**

**SisTADS**

Este documento apresenta a proposta do Sistema para Gerenciamento do Site do TADS - SisTADS, que tem como principal função controlar e gerenciar atividades do Curso de Tecnologia em Análise de Desenvolvimento de Sistemas. Para a construção do software foi feita a análise de requisitos, aplicação de técnicas de modelagem do esquema e das classes.

**Januária/MG**

**Abril 2015**

**Índice**

**1 – Introdução** 1.1 Introdução à situação problema  
 1.2 Mini-Mundo  
 1.3 Requisitos  
 1.3.1 Requisitos Funcionais e Não Funcionais  
 1.3.2 Requisitos Suplementares  
 1.4 Objetivos 1.4.1 Objetivos Gerais  
 1.4.2 Objetivos Específicos **2 – Análise** 2.1 Modelo Conceitual  
 2.2 Modelo Lógico do Banco de Dados  
 2.3 Diagrama de Pacotes  
 2.4 Diagrama de Classes  
 2.5 Diagrama de Caso de Uso  
 2.6 Especificação de Caso de Uso  
 2.7 Diagrama de Estados  
 2.8 Diagrama de Sequência  
 2.9 Diagrama de Implantação  
 2.10 Diagrama de Componente

**3 – Desenvolvimento** 3.1 Ambiente ALM  
 3.1.1 ALM  
 3.1.2 Linguagem  
 3.1.3 IDE  
 3.1.4 Controle de Versão  
 3.1.5 SGBD  
 3.1.6 Issue/Bug Tracking  
 3.1.7 Criação de Diagramas UML  
3.2 Arquitetura de Software  
3.3 API'S  
3.4 Descrição da Arquitetura  
 3.4.1 Tecnologias  
3.5 Protótipos de Telas  
  
**4 - Referências**

**1 – Introdução**

**1.1 Introdução à situação problema**

**1.2 Mini-Mundo**

O Instituto Federal do Norte de Minas Gerais - Campus Januária, possui alunos matriculados no curso de Tecnologia em Análise e Desenvolvimento de Sistemas (TADS). Deseja-se controlar e gerenciar algumas atividades, além de publicar informações importantes. Para isto é necessário a implementação de um site que atenda os seguintes requisitos.

Para todos os alunos é necessário guardar as seguintes informações: matrícula, nome, endereço completo, sexo, data de nascimento, cidade, estado, RG, CPF, telefones, e-mail, período referente ao curso e disciplinas cursadas. E para o curso: nome, código e carga horária. Durante todo o curso, é realizado eventos (nome, data), mobilizando alunos e professores a prestigiarem palestras, trabalhos acadêmicos, maratonas de programação, entre outros. Devemos gerenciar quais as disciplinas (código, nome, carga horária) em que o professor (nome, sexo, CPF, currículo lattes) leciona.

É preciso registrar o perfil socioeconômico (estrutura familiar, renda), de cada aluno e cada professor. O registro da escola anterior (nome, ano de conclusão) do aluno é registrado para guardar informações importantes e após a conclusão, é feito o registro da instituição futura (nome, ano de admissão, cargo), permitindo o acompanhamento dos administradores. É permitido ainda que cada aluno, gerencie seus tipos de produções (artigo, TCC) e horas de AACC, ao longe de todo o curso. Realizações de viagens (data, cidade) só poderá acontecer, se houver um professor responsável.

**1.3 Requisitos**

**1.3.1 Requisitos Funcionais e Não-Funcionais**

**Responsáveis**

|  |  |
| --- | --- |
| **Nome** | **Assinatura** |
| André Fellype Matos | André |
| Erika Maria dos Santos | Erika |
| Vinícius Dias de Souza | Vinícius |
| Mauro | Mauro |

**Requisitos Funcionais e Não-Funcionais**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F1 – Controle de Acesso** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de Login do usuário. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF1.1 – Campo de Nome | O campo de nome terá o nome de usuário. |  | ( ) | (x) |
| NF1.2 – Campo de Senha | O campo de senha terá a senha do usuário. |  | ( ) | (x) |
| NF1.3 – Campo de Erro | O campo de erro mostrará o erro se o usuário for inválido. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F2 – Cadastro de Turma** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de turma, onde somente o administrador terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF2.1 – Campo Período | O campo período deve com o período do ingresso da turma no IFNMG. |  | ( ) | (x) |
| NF2.2 – Campo Ano | O campo ano deve conter o ano do ingresso da turma no IFNMG. |  | ( ) | (x) |
| NF2.3 – Campo Ingresso | Modificar. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F3 – Cadastro de Professor** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de professor, onde somente o professor terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF3.1 – Campo Nome | O campo nome deve conter o nome do professor com 150 caracteres. |  | ( ) | (x) |
| NF3.2 – Campo Sexo | O campo sexo de conter o sexo do professor (F ou M). |  | ( ) | (x) |
| NF3.3 – Campo Cpf | O campo cpf deve conter uma máscara (###. ###. ###-##) que contem 14 caracteres. |  | ( ) | (x) |
| NF3.4 – Campo Currículo Lattes | O campo Currículo Lattes deve conter o currículo do professor com 150 caracteres. |  | ( ) | (x) |
| NF3.5 – Campo Estado Civil | O campo Estado Civil deve conter o estado civil do professor (solteiro, casado, divorciado, viúvo) que contem 25 caracteres. |  | ( ) | (x) |
| NF3.6 – Campo Renda | O campo renda deve conter a renda do professor do tipo numérico. |  | ( ) | (x) |
| NF3.7 – Campo Nome Pai | O campo nome pai deve com o nome do pai do professor que contém 150 caracteres. |  | ( ) | (x) |
| NF3.8 – Campo Nome Mãe | O campo nome pai deve com o nome da mãe do professor que contém 150 caracteres. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F4 – Cadastro de Disciplina** | | | | |
| **Descrição**: O Sistema deverá possuir uma tela de cadastro da disciplina, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF4.1 – Campo Nome | O campo nome deve conter o nome da disciplina com 150 caracteres. |  | ( ) | (x) |
| NF4.2 – Campo Carga Horaria | O campo carga horária deve conter a quantidade de hora da disciplina do tipo int. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F5 – Cadastro de Cidade** | | | | |
| **Descrição**: O Sistema deverá possuir uma tela de cadastro de cidade, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF5.1 – Campo do nome | O campo nome deve possuir o nome da cidade e conter 150 caracteres. |  | ( ) | (x) |
| NF5.2 – Campo do estado | O campo do estado deve possui o estado da cidade conter 2 caracteres. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F6 – Cadastro de Tipo de Evento** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de tipo evento, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF6.1 – Campo de Viagem | O campo de viajem deve possuir a descrição da viajem e deve conter 150 caracteres. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F7 – Cadastro de Evento** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de evento, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF7.1 – Campo de nome | O campo de nome deve conter o nome do evento e conter 150 caracteres. |  | ( ) | (x) |
| NF7.2 – Campo de data | O campo de data deve conter a data que o evento será realizado. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F8 – Cadastro de Curso** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de curso, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF8.1 – Campo de Curso | O campo de curso o terá o nome do curso e vai conter 150 caracteres. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F9 – Cadastro de Aluno** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de aluno, onde somente o aluno terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF9.1 – Campo Nome | O campo de nome conterá o nome do aluno e 150. |  | ( ) | (x) |
| NF9.2 – Campo de Sexo | O campo de sexo conterá o sexo (F ou M) e 1 caractere. |  | ( ) | (x) |
| NF9.3 – Campo de Data de Nascimento | O campo de nascimento terá a data de nascimento do aluno. |  | ( ) | (x) |
| NF9.4 – Campo de Matricula | O campo de matricula conterá o número de matricula do aluno. |  | ( ) | (x) |
| NF9.5 – Campo de Telefone | O campo de telefone terá o telefone do aluno e usara uma máscara ((##) ####-####). |  | ( ) | (x) |
| NF9.6 – Campo de RG | O campo de Rg terá o número da identidade do aluno e usara uma máscara (AA. ###.###). |  | ( ) | (x) |
| NF9.7 – Campo de CPF | O campo de Cpf terá o número do cpf do aluno e terá a sub-máscara (###. ###.###-##). |  | ( ) | (x) |
| NF9.8 – Campo de Quantidade de Irmãos. | O campo quantidade de irmão terá o número de irmãos que o aluno tem. |  | ( ) | (x) |
| NF9.9 – Campo de Renda | O campo de renda vai ter a renda bruta do aluno |  | ( ) | (x) |
| NF9.10 – Campo do Nome Mãe | O campo do nome da mãe terá o nome da mãe do aluno. |  | ( ) | (x) |
| NF9.11 – Campo do Nome do Pai | O campo do nome do pai terá o nome do pai do aluno. |  | ( ) | (x) |
| NF9.12 – Campo de Escola Anterior | O campo de escola anterior terá a escola que o aluno irá. |  | ( ) | (x) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **F10 – Cadastro de Produção** | | | | |
| **Descrição:** O Sistema deverá possuir uma tela de cadastro de produção, onde somente a escola terá acesso. | | | | |
| **Requisitos Não Funcionais** | | | | |
| Nome | Restrição | Categoria | Desejável | Permanente |
| NF10.1 – Campo Nome da Produção | O campo de nome da produção conterá o título da produção com 150 caracteres. |  | ( ) | (x) |
| NF10.2 – Campo Nome do Orientador | O campo do orientador tem o nome do orientador e terá 150 caracteres. |  | ( ) | (x) |
| NF10.3 – Campo data | O campo data terá a data da produção. |  | ( ) | (x) |
| NF10.4 – Campo Tipo de Produção | O campo tipo de produção terá o tipo de produção (TCC ou Artigo) que o aluno realizará. |  | ( ) | (x) |

**1.3.2 Requisitos Suplementares**

|  |  |  |  |
| --- | --- | --- | --- |
| **S1 – Tipo de interface** | | | |
| Restrição | Categoria | Desejável | Permanente |
| As interfaces do sistema devem ser implementadas como formulários que permitem maior usabilidade de fácil absorção por parte de quem o usa. |  | (X) | ( ) |

|  |  |  |  |
| --- | --- | --- | --- |
| **S2 – Persistência de dados** | | | |
| Restrição | Categoria | Desejável | Permanente |
| A camada de persistência deve ser implementada de forma que diferentes tecnologias de banco de dados possam vir a ser utilizadas no futuro. |  | (X) | ( ) |

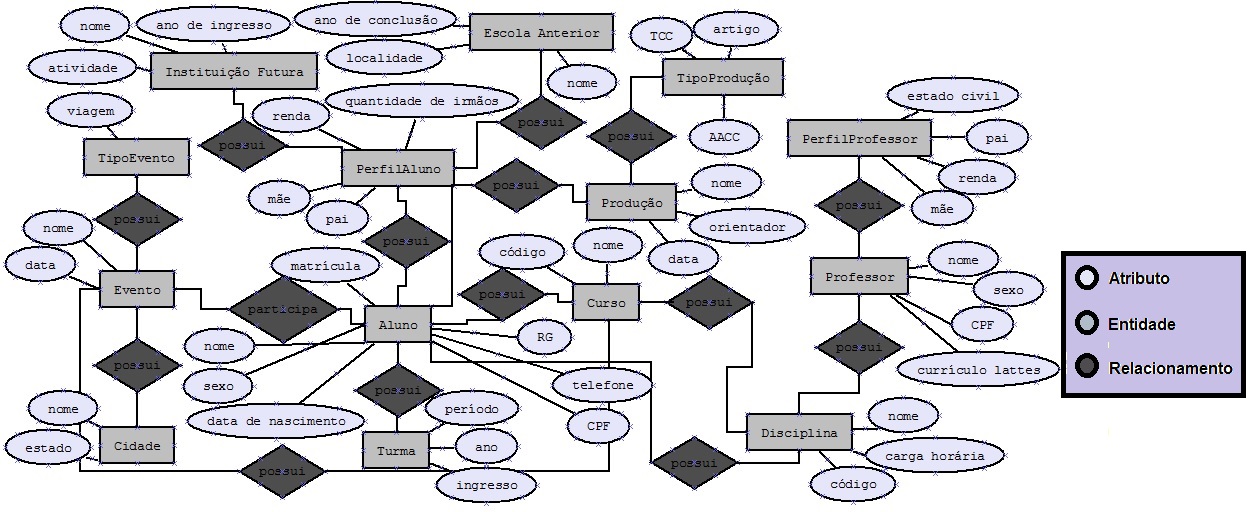
|  |  |  |  |
| --- | --- | --- | --- |
| **S3 – Portabilidade** | | | |
| Restrição | Categoria | Desejável | Permanente |
| A camada de apresentação deve ser implementada de forma que possa ser visualizada em dispositivos móveis. |  | (X) | ( ) |

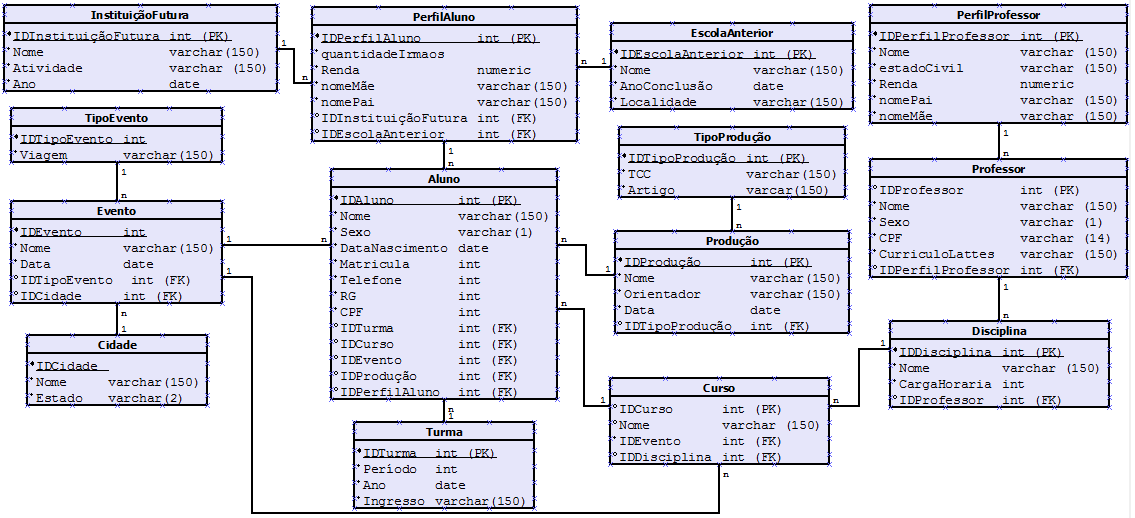
**1.4 Objetivos**

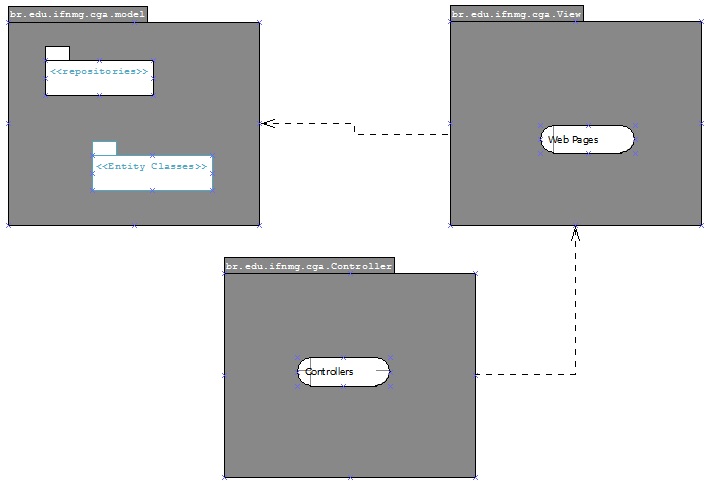
**1.4.1 Objetivos Gerais**

**1.4.2 Objetivos Específicos**

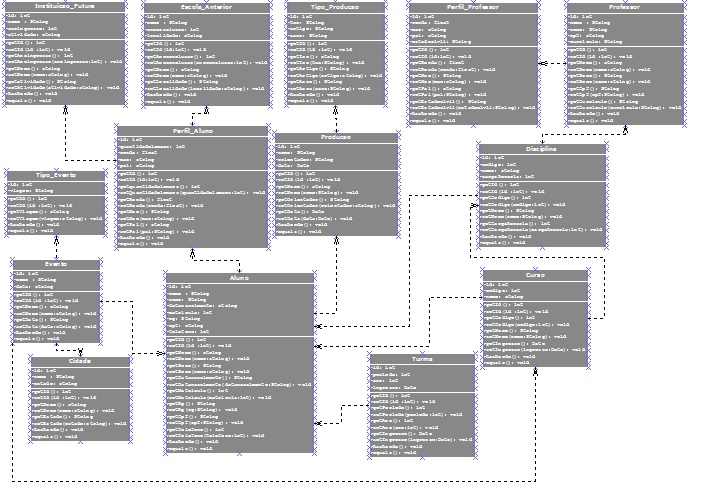
**2 - Análise**

**2.1 Diagrama Conceitual**

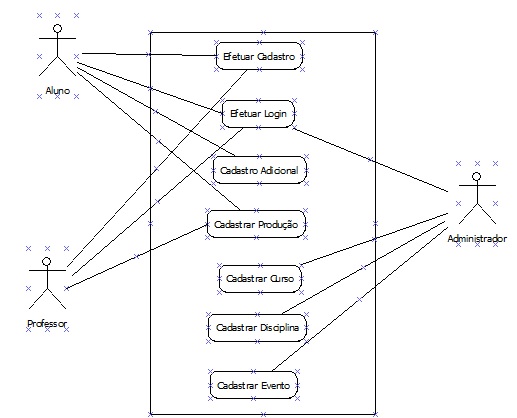
**2.2 Diagrama Lógico do Banco de Dados**

**2.3 Diagrama de Pacotes**

**2.4 Diagrama de Classes**

****

**2.5 Diagrama de Casos de Uso**

****

**2.6 Especificação do Caso de Uso**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Ator(es)** | **Descrição** | **Referência Cruzada** |
| CDU1:  Controle de Acesso | Alunos, Professores, Administrador | Os usuários informam seu login e sua senha. O sistema libera o acesso. | F2, F8, F10 |
| CDU2: | Administrador | O Administrador o registro de um novo curso, informando nome, descrição. | F1, F3, F7 |
| CDU3:  Registar Produção | Alunos,  Professores | Os usuários iniciam o registro de uma nova produção informando nome, tipo e a data. O sistema informa que foi registrado com sucesso. | F5, F9 |
| CDU4:  Registrar Eventos | Administrador | O Administrador inicia o registro de um novo evento informando tipo, nome, data e a cidade. O sistema informa que foi registrado com sucesso. | F4, F6 |

**Especificação**

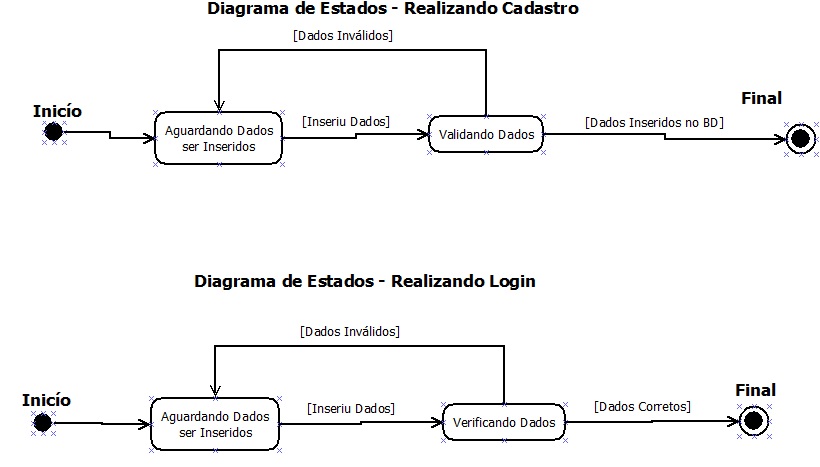
|  |  |
| --- | --- |
| **CDU 1** | **Controle de Acesso** |
| **Atores** | Alunos, Professores, Administrador |
| **Pré-Condições** | É necessário ter um usuário e senha previamente cadastrados no sistema. |
| **Fluxo Principal** | 1. O usuário preenche os campos login e senha;  2. O sistema libera o acesso;  3. O sistema apresenta a tela inicial respectiva;  4. O caso de uso termina; |
| **Pós-Condições** | O usuário estará logado; |
| **Fluxo Alternativo** |  |
| **Fluxo de Exceção** | Ocorre quando o login e/ou senha inseridos estão incorretos.  1.1. O sistema exibe mensagem “Login ou senha inválidos! ”.  1.2. O caso de uso retorna ao passo 1. |
| **Regras de Negócio** | 1. O campo login e o campo senha não devem ficar em branco.  2. O campo login deve ter no mínimo 3 caracteres.  3. O campo senha deve ter no mínimo 5 caracteres. |

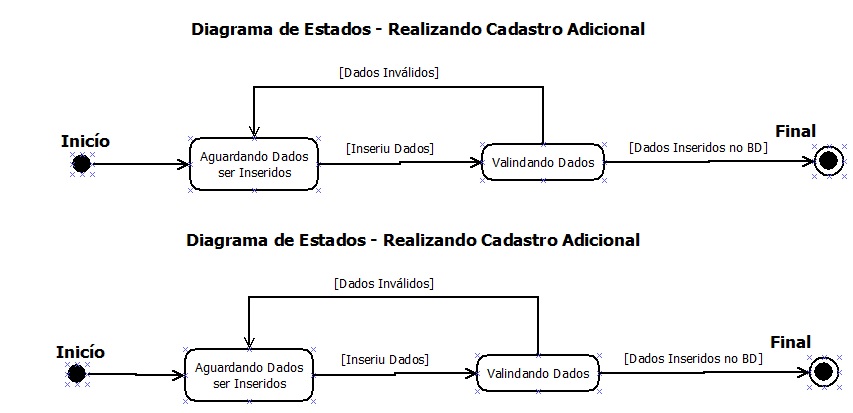
|  |  |
| --- | --- |
| **CDU 2** | **Registrar Curso** |
| **Atores** | Administrador |
| **Pré-Condições** | É preciso estar logado; |
| **Fluxo Principal** | 1. O administrador seleciona a opção “novo” no menu Curso;  2. O sistema exibe um formulário de cadastro de um novo curso;  3. O administrador preenche os campos do formulário;  4. O administrador confirma a operação;  6. O sistema retorna a seguinte mensagem: “Curso cadastrado com  Sucesso! “.  7. O caso de uso termina; |
| **Pós-Condições** | Um novo curso é cadastrado. |
| **Fluxo de Exceção** | 1. Caso o usuário opte por “cancelar” a operação o caso de uso será finalizado. |
| **Regras de Negócio** |  |

|  |  |
| --- | --- |
| **CDU 3** | **Registrar Produção** |
| **Atores** | Alunos, Professores |
| **Pré-Condições** | É preciso estar logado; |
| **Fluxo Principal** | 1. O usuário seleciona a opção “novo” no menu Produção;  2. O sistema exibe um formulário de cadastro de uma nova produção;  3. O usuário preenche os campos do formulário;  4. O usuário seleciona o tipo de produção;  5. O usuário confirma a operação;  6. O sistema retorna a seguinte mensagem: “Produção cadastrada com  Sucesso! “.  7. O caso de uso termina; |
| **Pós-Condições** | Uma nova produção é cadastrada. |
| **Fluxo de Exceção** | 1. Caso o usuário opte por “cancelar” a operação o caso de uso será finalizado. |
| **Regras de Negocio** | 1. O tipo de produção deve ser selecionado no seu respectivo combobox. |

|  |  |
| --- | --- |
| **CDU 4** | **Registrar Eventos** |
| **Atores** | Administrador |
| **Pré-Condições** | É preciso estar logado; |
| **Fluxo Principal** | 1. O administrador seleciona a opção “novo” no menu Eventos;  2. O sistema exibe um formulário de cadastro de um novo evento;  3. O administrador preenche os campos do formulário;  4. O administrador seleciona a cidade de realização do evento;  5. O administrador confirma a operação;  6. O sistema retorna a seguinte mensagem: “Evento cadastrado com  Sucesso! “.  7. O caso de uso termina; |
| **Pós-Condições** | Um novo evento é cadastrado. |
| **Fluxo de Exceção** | 1. Caso o administrador opte por “cancelar” a operação o caso de uso será finalizado. |
| **Regras de Negocio** | 1. A cidade de realização do evento deve ser selecionada no seu respectivo combobox. |

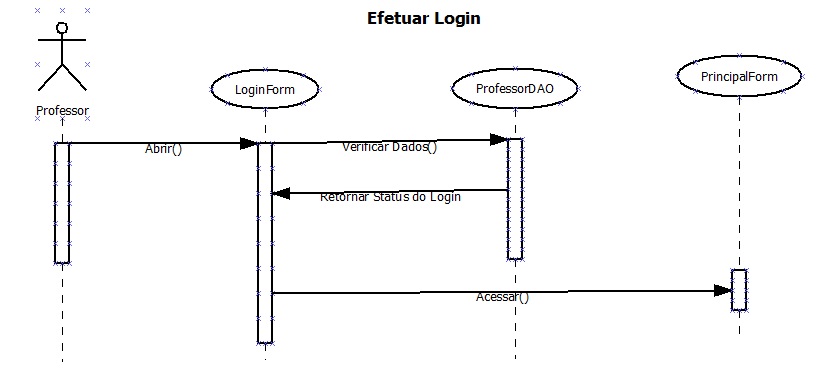
**2.7 Diagrama de Estados**

****

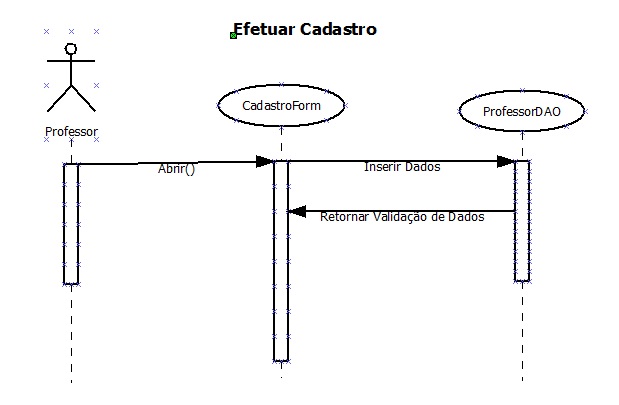
****

**2.8 Diagrama de Sequência**

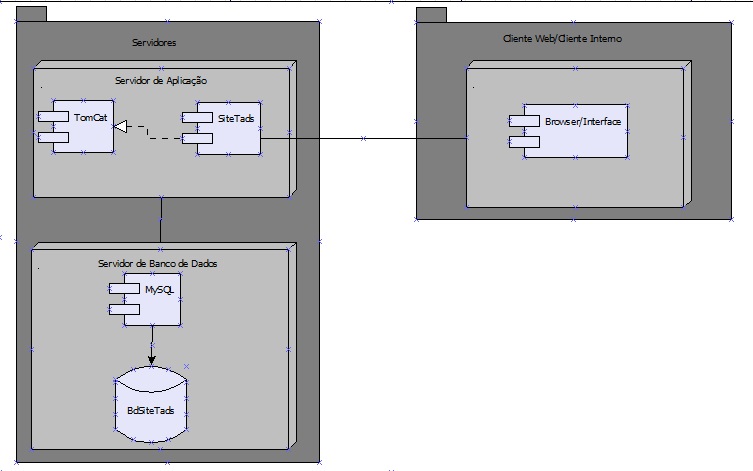
Caso de Uso – Autenticação de Usuários



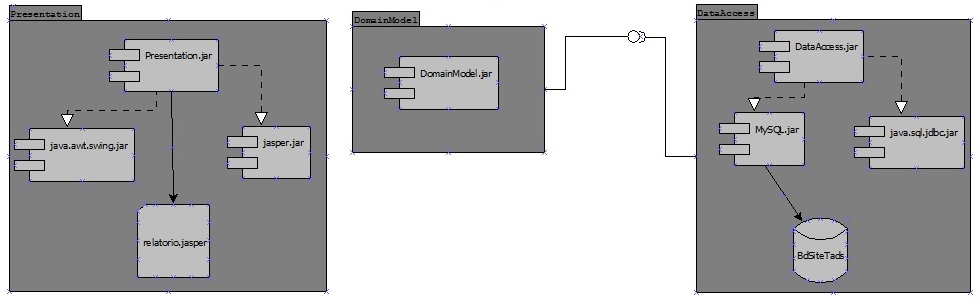
Caso de uso – Autenticação de Cadastro



**2.9 Diagrama de Implantação**

****

**2.10 Diagrama de Componente**

****

**3 – Desenvolvimento**

**3.1 Ambiente ALM**

**3.1.2 ALM**: Application Lifecycle Management (ALM), é todo o processo que guia a vida útil de uma aplicação desde a sua concepção, passando pela construção, operação e evolução. A ALM tem como objetivos: possibilitar maior controle ao projeto, previsibilidade no processo de desenvolvimento, garantia do acompanhamento das tarefas do projeto, disponibilizando um controle de versões mais eficiente, dentre outras [1].

**3.1.2 Linguagem:** Para a programação, será utilizada a linguagem Java, por ser uma linguagem de programação orientada a objetos estática e fortemente tipada. O Java, é também uma plataforma de desenvolvimento enriquecida de vantagens assim, será possível desenvolver aplicações para desktop, celular, web, etc. Arquivos podem ser compilados em Java, gerando um bytecode que é interpretado pelo JMV. Assim, será é permitido executa-lo em outros sistemas operacionais, desde que o mesmo possua uma JMV [2].

**3.1.3 IDE:** A IDE a ser usada no desenvolvimento do trabalho, será o NetBeans IDE 8.0, pois segundo [3] oferece suporte abrangente e de primeira classe para as tecnologias e melhorias de especificação Java mais recentes, antes de outros IDEs, além de funcionalidade avançadas. É o primeiro IDE gratuito a oferecer suporte a JDK 8, JDK 7, Java EE 7, inclusive a seus aprimoramentos de HTML5 relacionados, e a JavaFX2. O NetBeans IDE oferece suporte para as linguagens C / C ++ e desenvolvedores PHP, fornecendo editores e ferramentas abrangentes para os seus quadros e tecnologias relacionadas. Sua característica marcante é a instalação em todos os sistemas operacionais que suportam Java.

**3.1.4 Controle de Versão**: A ferramenta para controle de versão será o Git. Segundo [4], o Git é um controle de versão distribuída e é utilizado principalmente para gerenciar versões de software, desenvolvidos por um ou mais desenvolvedores, podendo ser implementado novas funcionalidades ficando registrado tudo em um histórico, o qual é permitido retroceder sempre que necessário, além dos integrantes do projeto, participar enviando correções, atualizações, etc. Alterações não comprometem o projeto principal, pois o dono do projeto sempre optará por incluir ou não as alterações realizadas. Utilizando-se o Git, é evidente algumas vantagens marcantes, tais como: consistência, velocidade, espaço e simplicidade.

**3.1.5 SGDB**: O Sistema Gerenciador de Banco de Dados (SGBD) a ser implementado será o MySQL, por ser conhecido pela facilidade de uso e capacidade de rodar em vários sistemas operacionais. Conforme [5], o banco possui vantagens de ser: gratuito, código fonte aberto, leve, rápido, seguro e prático.

**3.1.6 Ferramenta para Issue/Bug Tracking**: O GitHub é o maior anfitrião de código do planeta, com mais de **21,3 milhões** de repositórios. Grande ou pequeno, cada repositório vem com as mesmas ferramentas. A ferramenta issue tracker é flexível e por isso permite que você fique em cima dos erros e se concentre nos recursos [6].

**3.1.7 Criação de Diagramas UML**: Para a criação de diagramas UML, o programa utilizado será o Dia. O Dia é um programa baseado em gtk+ para criação do diagrama liberado sob a licença do GPL. Ele é inspirado no programa do Windows comercial "Visio", embora seja voltada para diagramas informais para uso casual. O Dia pode ser utilizado para desenhar tipos diferentes de diagramas, possuindo diversos objetos especiais para ajudar a desenhar diagramas de entidade relacionamento, diagramas UML, fluxogramas, diagramas de rede, e muitos outros; além da possibilidade de adicionar suporte para novos formatos de gravação de arquivos em XML, usando um subconjunto de SVG para desenhar a forma [7].

**3.2 Arquitetura de Software**

A arquitetura a ser utilizada para o desenvolvimento do software será o padrão dearquitetura Model View Controller (MVC). Pois, segundo [8] é dividida em 3 camadas distintas: Modelo (Model), Visão (View) e Controlador (Controller) e visa separar dados ou lógica de negócios (Model) da interface do usuário (View) e do fluxo da aplicação (Controller), permitindo que uma mesma lógica de negócios possa ser acessada e visualizada através de várias interfaces.

**3.3 API's**

As tecnologias a serem utilizadas para o desenvolvimento do software serão:

**Java**: O Java é uma tecnologia usada para desenvolver aplicações que tornam a Web mais divertida e útil, além de ser a base para praticamente todos os tipos de aplicações em rede; e o padrão global para o desenvolvimento e distribuição de aplicações móveis e incorporadas. O Java foi projetado para permitir o desenvolvimento de aplicações portáteis de alto desempenho para a mais ampla variedade possível de plataformas de computação. De forma eficiente, o Java permite que você desenvolva, implante e use aplicações e serviços estimulantes.   
  
O Java tornou-se inestimável para os desenvolvedores, permitindo que eles:

* Gravem software em uma plataforma e o executem virtualmente em qualquer outra plataforma.
* Criem programas que podem ser executados dentro de uma web browser e acessem web services disponíveis.
* Desenvolvam aplicações do servidor para fóruns on-line, armazenamentos, pesquisas, processamento de forms HTML e mais.
* Combinem aplicações ou serviços usando a linguagem Java para criar aplicações ou serviços altamente personalizáveis.
* Crie aplicações potentes e eficientes para telefones celulares, processadores remotos, microcontroladores, módulos sem fio, sensores, gateways, produtos de consumo e praticamente qualquer outro dispositivo eletrônico [9].

**Java EE**: Segundo [10], o **Java EE** (Java Enterprise Edition) consiste de uma série de especificações bem detalhadas, dando uma receita de como deve ser implementado um software que faz cada um desses serviços de infraestrutura, tais como: persistência em banco de dados, transações, acesso remoto, web services, gerenciamento de threads, gerenciamento de conexões HTTP, cache de objetos, gerenciamento de sessão web, balanceamento de carga, dentre outros.   
O principal objetivo do Java EE é fornecer um conjunto de recursos para o desenvolvimento de sistemas baseados no modelo multicamadas, denominados de aplicações corporativas. Assim, aplicações que seguem a solução Java EE são beneficiadas pelo modelo de componentes de software [11].   
Algumas APIs disponibilizadas pelo Java Enterprise:

* JavaServer Pages (JSP), Java Servlets, Java Server Faces (JSF) (trabalhar para a Web, onde é focado este curso).
* Enterprise Javabeans Components (EJB) e Java Persistence API (JPA). (objetos distribuídos, clusters, acesso remoto a objetos etc).
* Java API for XML Web Services (JAX-WS), Java API for XML Binding (JAX-B) (trabalhar com arquivos xml e webservices).
* Java Autenthication and Authorization Service (JAAS) (API padrão do Java para segurança).
* Java Transaction API (JTA) (controle de transação no contêiner).
* Java Message Service (JMS) (troca de mensagens assíncronas).
* Java Naming and Directory Interface (JNDI) (espaço de nomes e objetos).
* Java Management Extensions (JMX) (administração da sua aplicação e estatísticas sobre a mesma).

**EJB3**: O Enterprise Java Beans (EJB) 3.0 é uma plataforma para criação de aplicações de negócio portáveis, robustas e reutilizáveis usando a linguagem de programação Java. A vantagem de utilização do EJB, consiste no fato do desenvolvedor utilizar mais tempo desenvolvendo lógica de negócio assim, preocupando-se menos com transação, persistência de dados, serviços de rede e outros.

Existe três tipos de EJB, cada um com sua finalidade:

* Session Beans – Utilizado para guardar a lógica de negócio da aplicação.
* Message-Driven Bean – Utilizado para troca de mensagens.
* Entity Bean (até versão EJB 2.x) – Utilizado para representar as tabelas do banco de dados [12].

**JPA 2.0:** Java Persistence API é uma especificação para camada de persistência dos dados. É utilizada nas aplicações Java para permitir a gravação/leitura de objetos em bancos de dados relacionais de forma transparente, técnica conhecida como ORM (Object Relational Mapping).

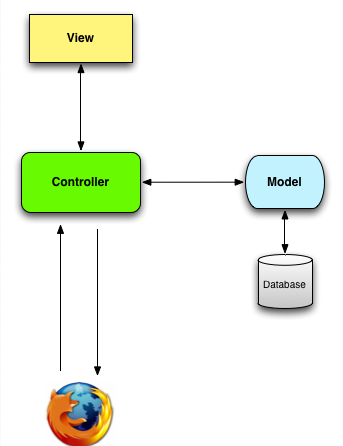
O JPA 2.0 é uma versão atualizada, que incluiu características que não estavam na primeira versão. O JPA 2.0, foi especificado pela JSR 317 do Java EE 6 e sua implementação de referência é o EclipseLink. A principal característica da versão do JPA 2.0, foi em relação ao mapeamento objeto-relacional, as quais mais sofreram melhorias, porque além de melhorar alguns mapeamentos que eram limitados na versão anterior, criou-se outros recursos de mapeamento com base no que já existia nos frameworks de persistência proprietários como Hibernate, visando atender às necessidades da maioria dos usuários, dentre outras [13].

**JSF 2.0:** O JavaServer Faces (JSF), é uma tecnologia que permite criar aplicações Java para Web utilizando componentes visuais pré-prontos, de forma que o desenvolvedor não se preocupe com Javascript e HTML. Basta adicionarmos os componentes (calendários, tabelas, formulários) e eles serão renderizados e exibidos em formato html. Outra característica é a separação entre as camadas de apresentação e de aplicação. Pensando no modelo MVC, o JSF possui uma camada de visualização bem separada do conjunto de classes de modelo. O JSF possui ainda a vantagem de ser uma especificação do Java EE, isto é, todo servidor de aplicações Java tem que vir com uma implementação dela e há diversas outras disponíveis [14].

O JSF 2.0 é uma versão que traz uma série de melhorias e mudanças úteis para o desenvolvedor de software, como o AJAX nativo, anotações, navegação implícita e condicional, suporte a requisição via GET, View Scope, Resources, Project Stage, dentre outras [15].

**3.4 Descrição da Arquitetura**

Definiu-se construir a arquitetura em três camadas: Controle, Negócio e Modelo. A utilização dessas camadas é uma adaptação do padrão de projeto MVC (Model-View-Controller) que permite o isolamento de implementação e reduz o impacto de possíveis mudanças em cada uma. A camada Controle agrupa a lógica necessária para receber a interação do usuário e interagir com a camada Negócio, que possui os métodos implementados para a realização da lógica de negócios em si. A camada Modelo fornece os métodos para a realização das operações CRUD (Criar, Buscar, Atualizar e Remover) nas entidades referenciadas no sistema em questão. A separação das camadas visa isolar possíveis mudanças na arquitetura através de interfaces bem definidas entre elas.



**3.4.1 Tecnologias**

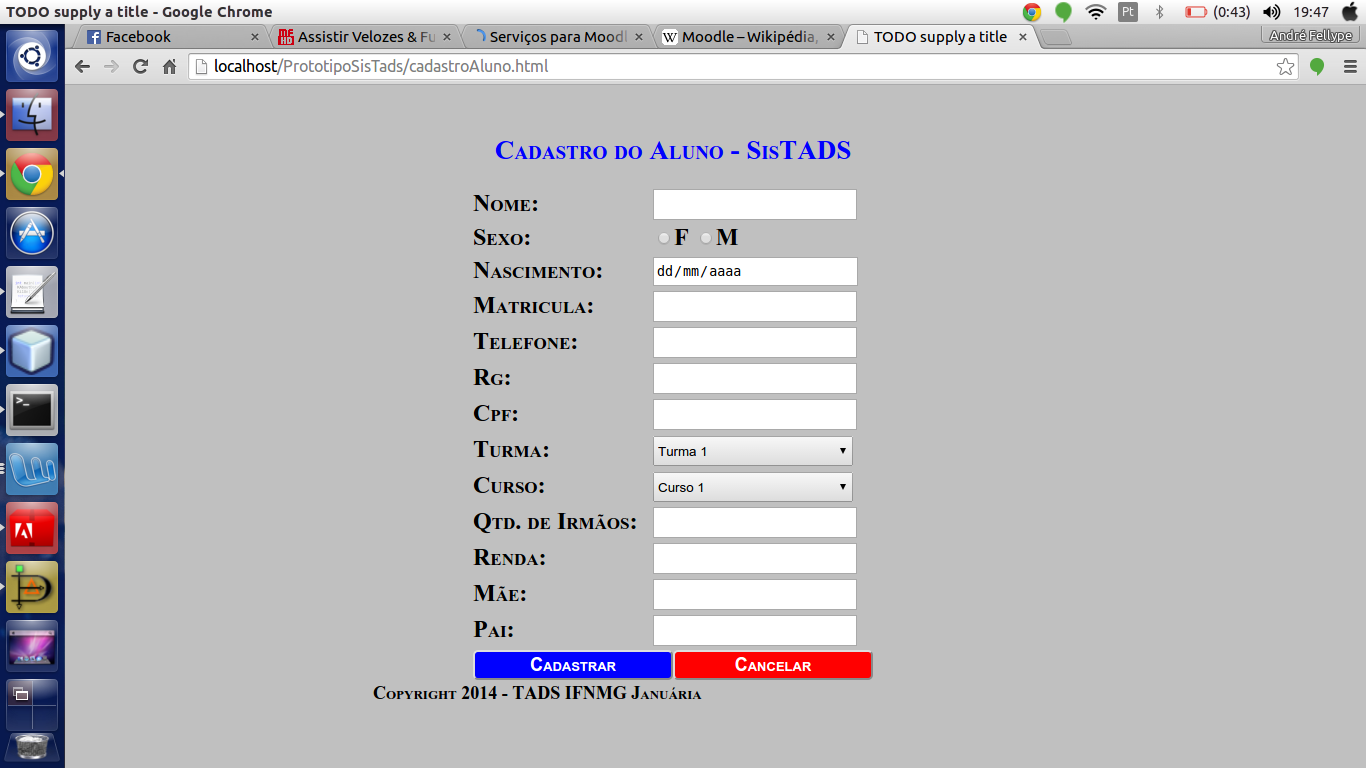
A camada Negócio da aplicação é implementada com o uso da tecnologia Enterprise Java Beans (EJB). Através dessa tecnologia, é possível distribuir os objetos responsáveis pela lógica de negócio em diferentes servidores, permitindo uma solução escalável e distribuída. Além desses benefícios, conforme a quantidade de aplicações se expande, é possível formar uma Arquitetura Orientada a Serviços (SOA), facilitando a interação com ferramentas atuais construídas com esse foco. A camada da interface gráfica, em contato direto com o usuário final, é desenvolvida através da tecnologia JSP (Java Server Pages), em conjunto com tecnologias como: HTML (HyperText Markup Language), Javascript, CSS (Cascading StyleSheet), Ajax.

**3.5 Protótipos de Telas**

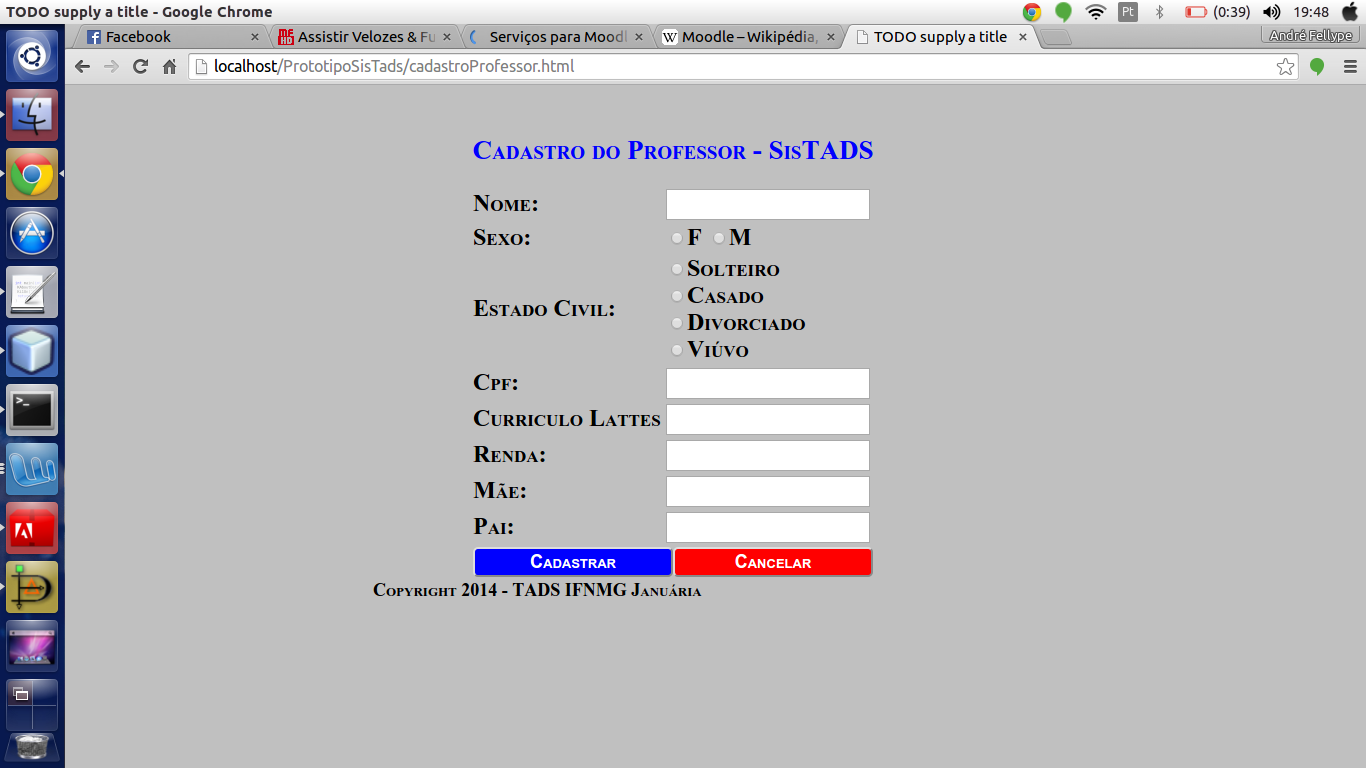
Controle de Acesso

****

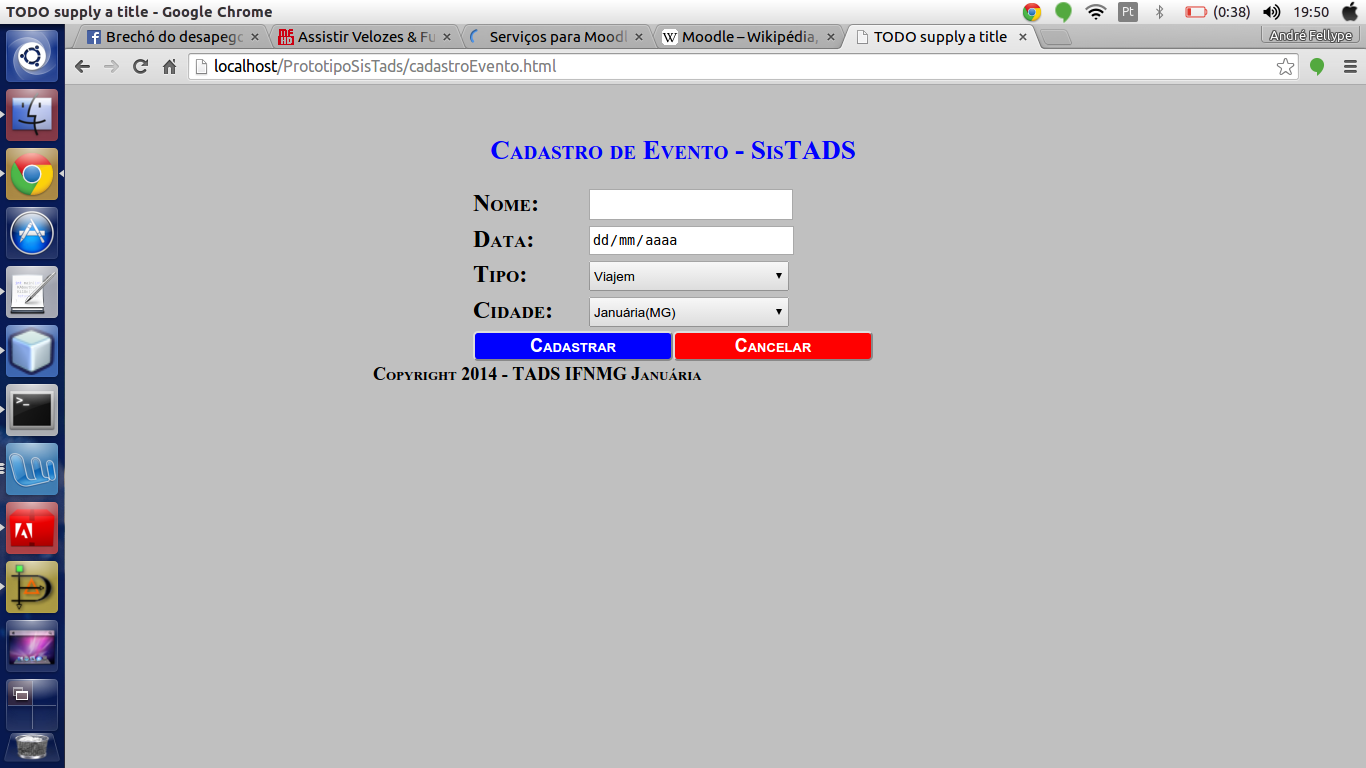
Cadastro de Aluno

****

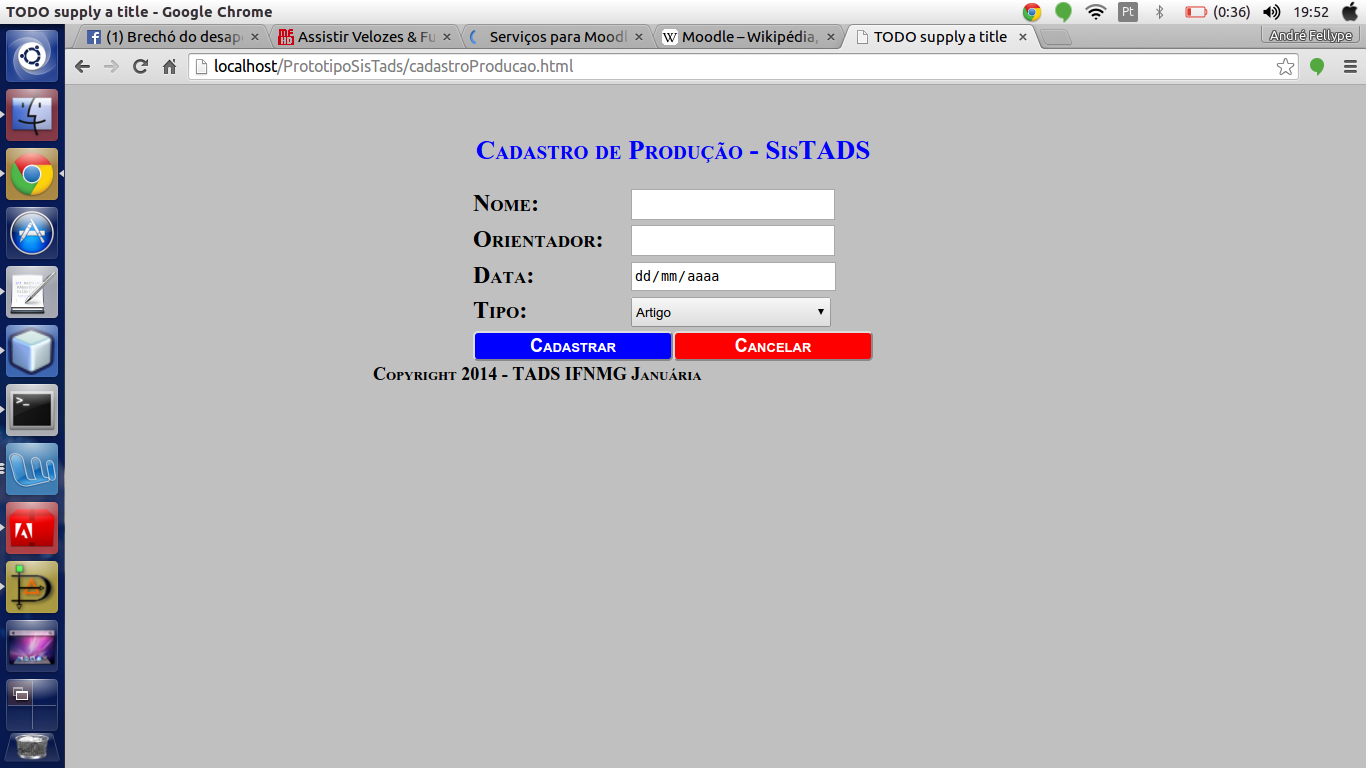
Cadastro de Professor

****

Cadastro de Evento

****

Cadastro de Produção

****

**4 – Referências**

[1] **Introdução ao Application Lifecycle Management (ALM)**. Disponível em: <https://msdn.microsoft.com/pt-br/library/ee156630.aspx >. Acesso em: 04/04/2015.

[2] DE SANTANA, Otávio Gonçalves. **Por que Java?**. Disponível em: <<http://www.devmedia.com.br/por-que-java/20384>>. Acesso: 04/04/2015.

[3] **NetBeans IDE**. Disponível em: < https://netbeans.org/features/index.html>. Acesso em: 04/04/2015.

[4] SILVESTRE, Henrique. **O que é** **Git?**. Disponível em: <<http://pt.slideshare.net/HenriqueSilvestre1/git-e-github-conceitos-bsicos>>. Acesso em: 05/04/2015.

[5] SANTOS, Adriano. **MySQL: Quem é você?**. Disponível em: <<http://www.devmedia.com.br/mysql-quem-e-voce/1752>>. Acesso em: 05/04/2015.

[6] **GitHub – Features**. Disponível em: <<https://github.com/features>>. Acesso em: 05/04/2015.

[7] WILLIAM, Jon Mc Cann. **Dia**. Disponível em: <<https://wiki.gnome.org/action/show/Apps/Dia?action=show&redirect=Dia>>. Acesso em: 05/04/2015.

[8] DE ALMEIDA, Rodrigo Rebouças. **Model – View - Controller (MVC)**. Disponível em: <<http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/arqu/mvc/mvc.htm>>. Acesso em: 05/04/2015.

[9] **Java**. Disponível em: <<https://www.java.com/pt_BR/>>. Acesso em: 04/04/2015.

[10] **O que é Java EE?** . Disponível em: <<http://www.caelum.com.br/apostila-java-web/o-que-e-java-ee/>>. Acesso em: 04/04/2015.

[11] MARTINS, Cláudio. **Aplicações corporativas multicamadas**. Disponível em: <<http://www.devmedia.com.br/aplicacoes-corporativas-multicamadas-parte-2-revista-java-magazine-112/27083>>. Acesso em: 04/04/2015.

[12] **Introdução ao Enterprise JavaBeans 3.0**. Disponível em: <http://www.universidadejava.com.br/docs/introducaoaoenterprisejavabeans30>. Acesso em: 04/04/2015.

[13] PAGANINI, Silvio. **JPA 2.0 – Persistência e toda prova**. Disponível em:< <http://www.devmedia.com.br/jpa-2-0-persistencia-a-toda-prova-java-magazine-81/17437>>. Acesso em: 04/04/2015.

[14] **Introdução ao JSF e Primefaces**. Disponível em: <<http://www.caelum.com.br/apostila-java-testes-jsf-web-services-design-patterns/introducao-ao-jsf-e-primefaces/#7-2-caracteristicas-do-jsf>>. Acesso em: 04/04/2015.

[15] **As grandes novidades do JSF 2.0**. Disponível em: <<http://pt.slideshare.net/4linuxbr/as-grandes-novidades-do-jsf-20-10668720>>. Acesso em: 04/04/2015.