**UNIVERSIDADE ESTADUAL DO TOCANTINS**

**CÂMPUS DE PALMAS**

**CURSO DE SISTEMAS DE INFORMAÇÃO**

**PROJETO E DESENVOLVIMENTO DE UM WEB SERVICE PARA ALOCAÇÃO DE CURSOS, SALAS E PROFESSORES**

**Autor: SÁVIO MARTINS VALENTIM**

Palmas – TO

2017

**UNIVERSIDADE ESTADUAL DO TOCANTINS**

**CÂMPUS DE PALMAS**

**CURSO DE SISTEMAS DE INFORMAÇÃO**

**PROJETO E DESENVOLVIMENTO DE UM WEB SERVICE PARA ALOCAÇÃO DE CURSOS, SALAS E PROFESSORES**

**Autor: SÁVIO MARTINS VALENTIM**

Projeto de Estágio submetido à Universidade Estadual do Tocantins para obtenção de aprovação na disciplina de Estágio Supervisionado do curso de Sistemas de Informação (Bacharelado).

Orientador: Prof. Me. Silvano Maneck Malfatti

Palmas, junho de 2017

**RESUMO:** Este trabalho objetiva apresentar o desenvolvimento de um sistema web para alocação de salas e horários semestrais, utilizando tecnologias livres e populares do mercado, utilizando paradigmas de desenvolvimento de software e documentando os resultados obtidos ao final do trabalho.

**Palavras-chave:** Desenvolvimento Web, Web Service, Front-end, Back-end

**ABSTRACT:** This work aims to present the development of a web system for the allocation of semesters and timetables, using free and popular technologies of the market, using paradigms of software development and documenting the results obtained at the end of the work.

**Key word:** Web Development, Web Service, Front-end, Back-end **LISTA DE FIGURAS**

**Figura 01:** Ciclo de vida do PHP

**Figura 02:** Elementos do modelo espiral e suas iterações

**Figura 03:** Objetos do MVC e suas interações.

**Figura 04:** Arquitetura do Sistema

**Figura 05:** Casos de uso do projeto e seus atores

**Figura 06:** Banco de dados e seus relacionamentos

**Figura 07:** Protótipo de Tela

**Figura 08:** Protótipo de Tela

**Figura 09:** Protótipo de Tela

**Figura 10:** Protótipo de Tela

**Figura 11:** Protótipo de Tela

**Figura 12:** Protótipo de Tela

**Figura 13:** Protótipo de Tela

**Figura 14:** Arquitetura do Projeto

**Figura 15:** Layout final do projeto

**Figura 16:** Layout final do projeto

**Figura 17:** Layout final do projeto

**Figura 18:** Layout final do projeto

**Figura 19:** Layout final do projeto

**Figura 20:** Layout final do projeto

**Figura 21:** Layout final do projeto

**Figura 22:** Layout final do projeto

**Figura 23:** Layout final do projeto

**Figura 24:** Layout final do projeto

**LISTA DE QUADROS**

**Quadro 1:** Cronograma de Fevereiro

**Quadro 2:** Cronograma de Março

**Quadro 3:** Cronograma de Abril

**Quadro 4:** Cronograma de Maio

**Quadro 5:** Cronograma de Junho

**Quadro 6:** Cronograma de Julho

**Quadro 7:** Requisitos Funcionais

**Quadro 8:** Requisitos Não Funcionais

**LISTA DE PALAVRAS ABREVIADAS**

**PHP:** Hypertext Preprocessor

**ORM:** Object-Relational Mapping

**HTML:** Hypertext Markup Language

**HTTP:** Hypertext Transfer Protocol

**FPM:** FastCGI Process Manager

**MVC:** Model View Controller

**SGBD:** Sistema de Gerenciamento de Banco de Dados

**SUMÁRIO**

[1. INTRODUÇÃO 8](#_Toc486299144)

[1.1. JUSTIFICATIVA 8](#_Toc486299145)

[1.2. OBJETIVOS 9](#_Toc486299146)

[1.2.1. OBJETIVO GERAL 9](#_Toc486299147)

[1.2.2. OBJETIVOS ESPECIFICOS 10](#_Toc486299148)

[1.3. PROBLEMA 10](#_Toc486299149)

[2. FUNDAMENTAÇÃO TEÓRICA 11](#_Toc486299150)

[2.1. PHP 11](#_Toc486299151)

[2.2. Bootstrap 12](#_Toc486299152)

[2.3. JQuery 12](#_Toc486299153)

[2.4. Doctrine 2 12](#_Toc486299154)

[2.5. NGinX 13](#_Toc486299155)

[2.6. PHP-FPM 13](#_Toc486299156)

[2.7. Microsoft® SQL Server™ 2008 R2 13](#_Toc486299157)

[2.8. Modelo Espiral 14](#_Toc486299158)

[2.9. Scrum 15](#_Toc486299159)

[2.10. Padrão Arquitetura MVC 15](#_Toc486299160)

[3. METODOLOGIA 17](#_Toc486299161)

[3.1. FASE 1: PROJETO 17](#_Toc486299162)

[3.2. FASE 2: DESENVOLVIMENTO 17](#_Toc486299163)

[3.3. FASE 3: IMPLANTAÇÃO 18](#_Toc486299164)

[3.4. LIMITAÇÕES 18](#_Toc486299165)

[4. CRONOGRAMA 19](#_Toc486299166)

[5. DESENVOLVIMENTO 22](#_Toc486299167)

[5.1. PROJETO 22](#_Toc486299168)

[5.1.1. Diagrama de Arquitetura do Sistema 22](#_Toc486299169)

[5.1.2. Casos de Uso 23](#_Toc486299170)

[5.1.3. Requisitos Funcionais 23](#_Toc486299171)

[5.1.4. Requisitos Não Funcionais 24](#_Toc486299172)

[5.1.5. Schema Banco de Dados 25](#_Toc486299173)

[5.1.6. Protótipo de Telas 26](#_Toc486299174)

[5.2. DESENVOLVIMENTO 29](#_Toc486299175)

[5.2.1. Arquitetura do Projeto 29](#_Toc486299176)

[5.2.2. Arquivos do Projeto 30](#_Toc486299177)

[6. CONCLUSÃO 36](#_Toc486299178)

[7. REFERÊNCIAS 37](#_Toc486299179)

[8. ANEXOS 39](#_Toc486299180)

# INTRODUÇÃO

A tecnologia tem evoluído e junto com ela, os processos e os métodos utilizados nos ambientes de trabalho, sejam eles escritórios, escolas e até mesmo em restaurantes, os sistemas aonde a informação está disponível de maneira rápida.

Até mesmo o perfil profissional das pessoas sofreu impacto, tanto direto quanto indireto, com o advento da tecnologia: novas profissões surgiram enquanto outras foram obrigadas a se adaptar, abandonando os modelos centralizadores e manuais para meios digitais de acesso rápido e fácil.

Os analistas de sistemas são um dos inúmeros profissionais responsáveis pelo desenvolvimento desses sistemas inovadores que tem facilitado tanto a vida de milhões de usuários.

Sendo o Analista de Sistemas um profissional capaz de atuar no gerenciamento da execução de sistemas, cuidar das etapas que vão desde o planejamento, perpassando por todas as fases de implantação e organização de sistemas, esse relatório pretende relatar as atividades desenvolvidas no estágio supervisionado.

O estágio está sendo cumprido na UNITINS – Campus Graciosa, em um período que vai de 04/02 a 24/06/2016, onde estão sendo desenvolvidas atividades em conjunto com a Administração do Campus e a Coordenação de T.I.

As atividades têm por objetivo propiciar o aperfeiçoamento prático do conhecimento teórico adquirido durante o curso e o desenvolvimento de atividades de cunho profissional, bem como colaborar com o crescimento da instituição de ensino.

## JUSTIFICATIVA

O que motivou este trabalho foi observar a necessidade que os servidores administrativos possuem em relação aos recursos que lhe são fornecidos e compreender como um sistema pode ser construído para facilitar suas atividades.

Portais eletrônicos e sistemas web são cada vez mais atrativos para o usuário, uma vez que a informação pode ser acessada em tempo real, permitindo assim aos estudantes e professores acesso e monitoramento a qualquer mudança na grade de horários e mapeamentos das salas utilizadas.

A cena descrita a seguir é muito comum na rotina da universidade, principalmente no começo de todo semestre: acadêmicos e professores perdidos procurando informações a respeito de qual a sala e disciplina será ministrada no dia.

A UNITINS, seguindo a exemplo de outras universidades, possuí o Portal do Aluno (chamado de educa), que é um serviço web disponibilizado para que alunos e professores tenham acesso a diversas informações, como os conteúdos das disciplinas, o boletim acadêmico, frequência, entre outras informações relevantes do meio acadêmico.

Um exemplo de serviço utilizando este sistema é a rematrícula, é feita online através do educa, o que provê agilidade e conforto para os acadêmicos, que não precisam se locomover até a universidade para faze-la. O mapa semestral de salas é disponibilizado no sistema, mas ele é feito de maneira estática, na forma de um documento em PDF, não se diferenciando muito de um mural convencional.

Seguindo o exemplo da rematrícula, a automatização desse processo significa um grande avanço, pois além da rapidez, permitirá uma economia de recursos com impressões desnecessárias, uma vez que a cada mudança todas os quadros de horário precisam ser reimpressos, o que não ocorreria em um sistema acessível a todos, representando a economia de recursos tanto pessoais e financeiros.

Advindo a automação será possível obter informações concisas e em tempo real do mapeamento das salas e laboratórios do campus, permitindo aos usuários saber qual sala está alocada para o seu curso e período, obtendo apenas as informações de que precisa, sem necessidade de perder tempo procurando em uma tabela convencional.

## OBJETIVOS

Foram agrupados o objetivo geral e os objetivos específicos, ambos têm em comum colocar em exercício as competências adquiridas durante o curso, demonstrando a capacidade de desenvolver, de planejar, estruturar e executar um sistema de dados.

## OBJETIVO GERAL

Desenvolver um sistema web com suporte a serviços web para a alocação semestral de cursos, salas e professores da universidade, visando reduzir gastos, simplificar e otimizar a administração dos recursos físicos e operacionais do campus.

## OBJETIVOS ESPECIFICOS

* Definir uma metodologia de trabalho para o projeto proposto;
* Melhorar o acesso à informação de forma mais rápida e dinâmica utilizando os meios eletrônicos, agilizando os serviços realizados pelos usuários;
* Incentivar novos projetos semelhantes.

## PROBLEMA

Toda a parte de estrutura e gerenciamento de alocação de salas é feita manualmente, o que gera atrasos e outros problemas a cada mudança de horários realizados, dificultando a edição desses dados.

Além disso, a dificuldade de acesso a essas informações gera transtornos no uso dos recursos físicos disponíveis no campus, acarretando confusão entre todos os envolvidos no processo de uso.

# 

# FUNDAMENTAÇÃO TEÓRICA

## PHP

No site do PHP (2017) (que é um acrônimo recursivo para *PHP: Hypertext Preprocessor*) ele é definido como uma linguagem de script open source de uso geral, muito utilizada, e especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do HTML.

No PHP o código é executado no servidor, gerando um HTML que então é enviado de volta para o navegador do cliente. O cliente por sua vez, recebe os resultados como se fosse um código HTML normal, sem saber como nem em qual linguagem o script foi executado.

Um dos melhores pontos do PHP é a sua simplicidade e seu alto nível de funcionalidades, o que não afasta os iniciantes na linguem por conta de dificuldades e nem deixa os profissionais mais experientes sem recursos a disposição. A figura 01 exemplifica o ciclo de vida do PHP, mostrando como ele é processado pelo servidor e enviado ao cliente;



Figura 01 – Ciclo de vida do PHP

Fonte: https://pt.slideshare.net/cloudson/php-eu-escolho-voc-ufjf

Essas e outras informações a respeito de métodos, funções e versões podem ser encontradas no site [www.php.net](http://www.php.net).

## Bootstrap

É um framework front-end de código aberto desenvolvido pela equipe do *Twitter* (Bootstrap, 2017). Esse framework facilita a vida dos desenvolvedores web a criar sites com tecnologia mobile (responsivo) sem ter que se preocupar com o desenvolvimento do CSS.

O Bootstrap possui uma diversidade de componentes em *JavaScript* que auxiliam o desenvolvedor a implementar o design. Componentes como *tootlips*, menus-dropdown, modals, formulários e outros podem ser implementados sem a menor dificuldade, apenas acrescentando algumas configurações no código, sem a necessidade de criar scripts.

Este framework é muito utilizado e amplamente aceito no meio comercial devido as suas facilidades de uso e gama de recursos e componentes visuais que facilitam a vida dos desenvolvedores.

## JQuery

É uma biblioteca de *JavaScript* leve, fácil de usar e com uma curva de aprendizagem relativamente curta. É descrito como *“uma biblioteca de JavaScript rápida, pequena e rica em recursos. Faz coisas como travessia e manipulação de documentos HTML, manipulação de eventos, animação e Ajax muito mais simples com uma API fácil de usar que funciona em uma infinidade de navegadores.* ” (JQuery, 2017).

Ou seja. Esse framework substitui as maiores e mais complicadas tarefas do *JavaScript* por funções mais diretas, rápidas e compatíveis com a generalidade dos browsers, facilitando a vida do desenvolvedor que necessita trabalhar com *JavaScript*.

## Doctrine 2

O Doctrine 2 é descrito em seu projeto como um framework e uma biblioteca para mapeamento de banco de dados relacionais (Mapeamento Objeto-Relacional, da sigla em inglês, ORM) (Doctrine 2, 2017).

Na pratica, com o ORM as tabelas no banco de dados são representadas através de classes e seu uso ocorre como na orientação à objetos, sem a necessidade de se construir todas as consultas ao banco na mão, representando assim maior simplicidade em obter e mapear os valores de um banco de dados. Para comparação, esse framework tem a mesma ideia do Hibernate, que é um framework ORM para a linguagem JAVA.

## NGinX

O Nginx (chamado também de engine x) é um servidor proxy HTTP escrito por Igor Sysoev desde 2005. É caracterizado pelos usuários como um servidor web rápido, leve, e com inúmeras possibilidades de configuração para melhor performance, a mais notável sendo a divisão por containers (NGinx, 2017).

Tecnicamente, o Nginx consome menos memória que o Apache, pois lida com requisições Web através do conceito de “*event-based web server*” enquanto o Apache é baseado no conceito “*process-based server*”, ou seja, o Nginx responde de acordo com os eventos que ocorrem enquanto o Apache responde de acordo com os processos em execução. Apesar disso eles não são necessariamente “concorrentes”, Apache e Nginx podem trabalhar juntos.

As fontes e documentação são distribuídos sob cláusula 2 licenças BSD-like.A.

## PHP-FPM

O PHP-FPM (*FastCGI Process Manager*) é apontado pelo site do PHP como uma alternativa para a implementação FPM com alguns recursos adicionais (e principalmente) usado em sites pesados (PHP, 2017). Entre alguns desses recursos listados estão inclusos:

* Gerenciamento de processos avançado com stop/start;
* Habilidade para começar a trabalhar com uid/gid/chroot/environment diferentes, ouvindo em diferentes portas e últilizando-se de php.ini destintos (troca de safe\_mode);
* Reinício de emergência em caso de destruição acidental de opcode cache;
* Aceleração do processo de upload;
* *fastcgi\_finish\_request()* - função especial para finalizar pedido e liberar todos os dados ao mesmo tempo continuando a fazer algo demorado (conversão de vídeo, processamento de estatísticas, etc);
* php.ini-baseado em arquivos de configurações.

## Microsoft® SQL Server™ 2008 R2

A Microsoft (2014) descreve o Microsoft® SQL Server™ como um sistema de gerenciamento e análise de banco de dados para soluções de comércio eletrônico, linha de negócios e data warehouse. É um dos SGBDs (Sistema de Gerenciamento de Banco de Dados) mais utilizados no mundo, tendo como competidores outros sistemas como o da da Oracle (MySQL), Postgres SQL e Maria DB.

A versão utilizada pela instituição é o SQL Server 2008 R2, portanto sendo o mesmo banco utilizado neste projeto.

## Modelo Espiral

O modelo de desenvolvimento em espiral é um modelo incremental criado por Barry Boehm em seu artigo de 1988 *A Spiral Model of Software Development and Enhancement*. Apesar do modelo não ser o primeiro a discutir o Desenvolvimento iterativo e incremental, foi o primeiro modelo a explicar os seus motivos.

Leite (2007) o classifica como um *metamodelo* (um modelo sob o outro) que pode acomodar diversos processos específicos. Na prática, significa que se pode encaixar nele as principais características de outros modelos, adaptando as necessidades do desenvolvimento. Este modelo prevê prototipação, desenvolvimento evolutivo e cíclico, e as principais atividades da modelo cascata, como mostra a figura 02.

O modelo espiral descreve um fluxo de atividades cíclico e evolutivo constituído de quatro estágios (Leite, 2007).

1. Determinação de objetivos, soluções alternativas e restrições;
2. Analise de riscos das decisões do estágio anterior. Opcionalmente, a construção de protótipos e simulação do software;
3. Atividades da fase de desenvolvimento, como design, especificação, codificação e verificação. A cada ciclo pode surgir uma nova especificação, devendo esta ser verificadas após seu surgimento;
4. Revisão das etapas anteriores e o planejamento da próxima fase. Dependendo dos resultados obtidos nos estágios anteriores, pode-se optar por seguir o desenvolvimento num modelo cascata (linear), evolutivo ou transformação.

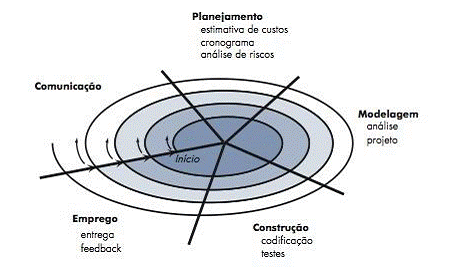


Figura 02 – Elementos do modelo espiral e suas iterações

Fonte: http://www.itnerante.com.br/profiles/blogs/resumo-sobre-o-modelo-em-espiral

## Scrum

O Desenvolvimento Ágil (2014) classifica o Scrum como uma metodologia ágil para gestão e planejamento de projetos de software. Ele também é um proesso que permite serem empregadas diversas ferramentas para o desenvolvimento iterativo e incremental utilizado no gerenciamento de projetos diversos e no desenvolvimento de software ágil.

Diferente de outras metodologias, o Scrum não é um conjunto de regras, e sim um conjunto de valores, princípios e práticas que fornecem a base para desenvolver o próprio método de desenvolvimento.

O Scrum possuí os sprints (onde os projetos são divididos em ciclos (sendo mais comumente semanais, quinzenais ou mensais), representando uma divisão do trabalho em iterações, aonde um checkpoint definirá o tempo aonde as tarefas devem ser executadas.

A cada Sprint, podem ser checadas se as tarefas estão completas e precisam de revisão ou o que for necessário, sendo uma vantagem uma participação maior do cliente no desenvolvimento do projeto.

## Padrão Arquitetura MVC

O MVC (Model-View-Controller) é um padrão arquitetural de software bastante utilizado e difundido no meio de, sendo um dos padrões mais conhecidos, principalmente o nas aplicações web e mobile, sendo o padrão de projeto escolhido para a implementação deste projeto.

Segundo o DevMedia (2012) o MVC inicialmente foi desenvolvido no intuito de mapear o método tradicional de entrada, processamento, e saída que os diversos programas baseados em GUI utilizavam. No padrão MVC, teríamos então o mapeamento de cada uma dessas três partes para o padrão MVC. Essas três partes podem ser definidas como:

* *Model*: representa a informação e a lógica de negócio do sistema;
* *View*: recebe entradas do sistema e exibe os resultados para os clientes. A View normalmente é algum tipo de GUI, mas não possui necessariamente uma interface gráfica;
* *Controller*: aceita entradas da View e requisições de informações do Model. Além disso, determina as ações.

Uma das maiores vantagens deste padrão arquitetural é que a View é separada do Model e assim ela pode ser alterada sem impactar no mesmo. Esse padrão é extremamente benéfico nas aplicações distribuídas atuais, como as mobiles e web, que podem ser conectadas ao mesmo Model.

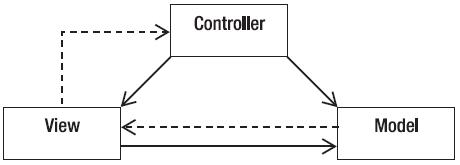


Figura 03 – Objetos do MVC e suas interações.

Fonte: www.oficinadanet.com.br

# METODOLOGIA

É a descrição dos passos que foram e serão realizados durante todo esse trabalho, ou seja, a etapa a qual cada parte corresponde. De forma geral, a metodologia segue um método hibrido entre os modelos espiral e a metodologia Scrum, descritas anteriormente. O trabalho compreende três fases como um todo, sendo essas fases a de Projeto, Desenvolvimento e Implantação.

Maiores detalhes a respeito do que foi feito nessas etapas estarão descritas no capitulo de desenvolvimento.

## FASE 1: PROJETO

Nessa etapa foi executado o desenvolvimento teórico do trabalho, ou seja, todos os alicerces do projeto foram definidos nessa primeira fase. Esses alicerces correspondem na elaboração e gestão de toda fundamentação teórica, estrutural e tecnológica presentes no escopo do projeto.

É nessa etapa que foram levantadas as funcionalidades, requisitos funcionais e não funcionais, casos de uso, entre outros. O principal método de trabalho utilizado foram as entrevistas e reuniões com o setor administrativo da UNIITINS, responsáveis pela alocação, e com o pessoal da DTIC, que é responsável pela infraestrutura de serviços.

Com o pessoal da administração foram feitas reuniões para o levantamento de funcionalidades a respeito do trabalho realizado, realizando das reuniões e das entrevistas os requisitos, *schema* de banco de dados, prototipação de telas do sistema, entre outros.

Com o pessoal da DTIC foram realizadas reuniões para definir as tecnologias utilizadas no projeto, uma vez que foi preciso ter conhecimento a respeito das tecnologias que já eram usadas por eles, além do suporte para as mesmas.

Em todos os dois casos foram feitas as validações das informações abstraídas por profissionais das áreas relacionadas, com reuniões de retorno para verificação com as partes interessadas.

## FASE 2: DESENVOLVIMENTO

Nessa etapa será executado o desenvolvimento prático do trabalho, ou seja, tudo o que foi levantado na fase anterior será utilizado no desenvolvimento do software. Cada parte será desenvolvida em separado, com os módulos com menor dependência de informações estrangeiras sendo desenvolvidos primeiro.

A cada módulo e etapa concluída, o mesmo será testado e revisado no Sprint, que está programado para ser quinzenal. Tanto o web service quanto o cliente serão desenvolvidos em simultâneo.

## FASE 3: IMPLANTAÇÃO

Nessa etapa será executada a implantação do servidor no ambiente de trabalho da UNITINS, com testes e verificações para adaptação e adequação do sistema conforme as necessidades que forem demandadas.

Na prática é uma extensão da fase de desenvolvimento, mais focada a correções de problemas e adaptações.

## LIMITAÇÕES

A limitação encontrada nesse projeto foi o fator tempo, o mesmo é muito curto para a execução, com outras atividades relacionadas do curso entrando em conflito com as atividades do estágio. Os prazos são bem curtos e é possível que a execução não fique 100% concluída por conta desses choques de horário.

# CRONOGRAMA

**Quadro 1 – Cronograma de Fevereiro**

|  |  |
| --- | --- |
| **Semana** | **Atividade Programada** |
| 1 | * Criação da Proposta |
| 2 | * Definição do Orientador e do Tema; * Providenciar Documentação; |
| 3 | * Providenciar Documentação; * Analise da Proposta e das tecnologias utilizadas |
| 4 | * Analise da Proposta e das tecnologias utilizadas; * Definição final do projeto |
| 5 | * Analise da estrutura do Sistema (recursos e tecnologias) * Revisão do cronograma e do sumário |

**Quadro 2 – Cronograma de Março**

|  |  |
| --- | --- |
| **Semana** | **Atividade Programada** |
| 1 | * Desenvolvimento   + Classificação das necessidades do projeto * Escrita - Documento Parcial   + Sumário Preliminar |
| 2 | * Desenvolvimento   + Levantamento de requisitos do sistema * Escrita - Documento Parcial   + Sumário Final   + Introdução |
| 3 | * Desenvolvimento   + Levantamento de Requisitos do sistema   + Projeto do Banco * Escrita - Documento Parcial   + Introdução   + Objetivos   + Justificativa |
| 4 | * Desenvolvimento   + Projeto do Banco * Escrita - Documento Parcial   + Fundamentação Teórica   + Projeto (Descrição) |
| 5 | * Desenvolvimento   + Diagrama de Classes * Escrita - Documento Parcial   + Projeto (Descrição) |

**Quadro 3 – Cronograma de Abril**

|  |  |
| --- | --- |
| **Semana** | **Atividade Programada** |
| 1 | * Desenvolvimento   + Desenho de Telas * Escrita - Documento Parcial   + Projeto (Descrição) |
| 2 | * Apresentação 1 * Desenvolvimento   + Desenho de Telas * Escrita – Documento Final   + Desenvolvimento |
| 3 | * Desenvolvimento   + Implementação das Regras de Negócio – Servidor (Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
| 4 | * Desenvolvimento   + Implementação das Regras de Negócio – Cliente (Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
| 5 | * Desenvolvimento   + Implementação das Regras de Negócio – Servidor (Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) * Amostra para a Administração do que foi feito |

**Quadro 4 – Cronograma de Maio**

|  |  |
| --- | --- |
| **Semana** | **Atividade Programada** |
| 1 | * Desenvolvimento   + Implementação das Regras de Negócio – Cliente (Telas, Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
| 2 | * Desenvolvimento   + Implementação das Regras de Negócio – Servidor (Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
| 3 | * Desenvolvimento   + Implementação das Regras de Negócio – Cliente (Telas, Interfaces e Métodos) * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
| 4 | * Desenvolvimento   + Implementação das Regras de Negócio – Servidor (Interfaces e Métodos)   + Início dos Testes na parte do servidor * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) |
|  | * Desenvolvimento   + Implementação das Regras de Negócio – Cliente (Telas, Interfaces e Métodos)   + Início dos Testes na parte do Cliente * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento) * Revisão do Desenvolvimento e Escrita (Sprint) * Amostra para a Administração do que foi feito |

**Quadro 5 – Cronograma de Junho**

|  |  |
| --- | --- |
| **Semana** | **Atividade Programada** |
| 1 | * Desenvolvimento   + Testes finais de implantação   + Correção de bugs * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento)   + Revisão dos Tópicos * Revisão do Desenvolvimento e Escrita (Sprint) |
| 2 | * Desenvolvimento   + Testes e implantação   + Correção de bugs * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento)   + Revisão dos Tópicos * Revisão do Desenvolvimento e Escrita (Sprint) |
| 3 | * Desenvolvimento   + Testes finais de implantação   + Correção de bugs * Escrita – Documento Final   + Desenvolvimento Prático do projeto (De acordo com o Desenvolvimento)   + Revisão dos Tópicos * Revisão do Desenvolvimento e Escrita (Sprint) |
| 4 | * Apresentação 2 do Projeto * Entrega Final do projeto |
| 5 | * Correção do que for solicitado |

# DESENVOLVIMENTO

Todo o desenvolvimento foi dividido em duas frentes: o Projeto, que contempla a fase 1 e o Desenvolvimento, que contempla as fases 2 e 3 da metodologia.

## PROJETO

É referente ao desenvolvimento teórico de todo o projeto que foi descrito na fase 1 da metodologia, ou seja, levantamento de requisitos, criação de diagramas de casos de uso e schema de banco de dados.

### Diagrama de Arquitetura do Sistema

Este diagrama tem por objetivo demonstrar o sistema como um todo, permitindo uma visualização geral dos componentes presentes no mesmo. A figura 04 demonstra os componentes gerais do sistema, como o servidor de hospedagem que contém o sistema web, web *service* e o banco de dados, bem como os usuários do sistema administrativo e dos consumidores do serviço.

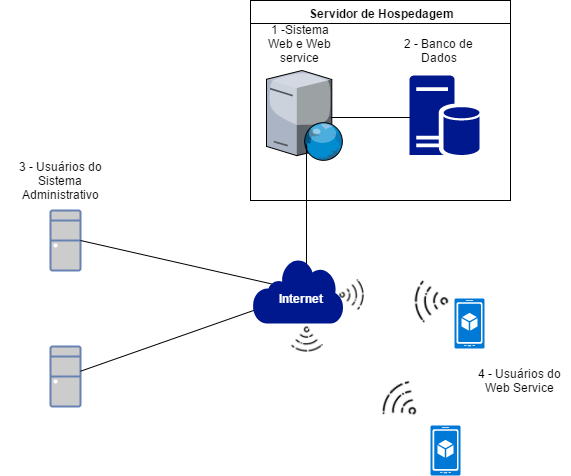


Figura 04 – Arquitetura do Sistema.

Fonte: elaborado pelo autor pelo web site www.draw.io.

### Casos de Uso

É um diagrama comportamental dos atores (usuários) do sistema. Ventura (2016) afirma que o Diagrama de Caso de Uso serve para representar como os casos de uso interagem entre si no sistema e com os usuários (atores), ou seja, como as funcionalidades se relacionarão umas com as outras e como serão utilizadas pelo usuário, durante o uso do sistema.

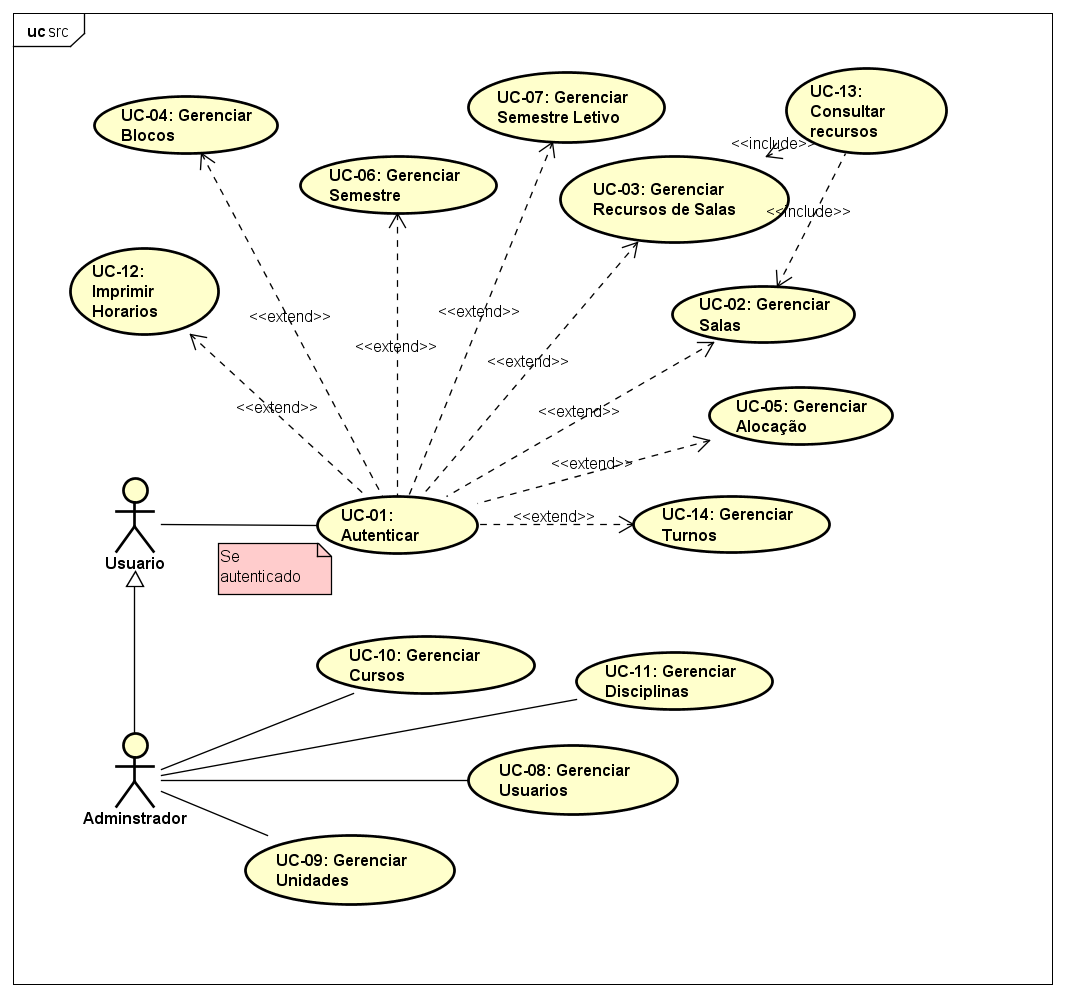


Figura 05 – Casos de uso do projeto e seus atores.

Fonte: elaborado pelo autor utilizando o software Astah.

### Requisitos Funcionais

Segundo Ventura (2016), um requisito funcional pode ser descrito como a requisição de uma função que um software deverá atender/realizar. Ou seja, é a exigência e a necessidade que um software deverá materializar para o seu usuário. O quadro 6 demonstra os requisitos funcionais levantados para o sistema.

Quadro 6 – Requisitos Funcionais do Sistema

|  |  |
| --- | --- |
| **Requisito** | **Caso de Uso Vinculado** |
| RF01: O sistema deve permitir o cadastro e gerenciamento de usuários e administradores no sistema | UC-08 |
| RF02: O sistema deve permitir ao administrador logado cadastrar novos campus no sistema; | UC-09 |
| RF03: O sistema deve permitir ao usuário cadastrar novos blocos no sistema; | UC-04 |
| RF04: O sistema deve permitir ao usuário cadastrar novas salas, laboratórios e os recursos nele presentes; | UC-02 |
| RF05: O sistema deve permitir ao usuário cadastrar novos semestres letivos; | UC-07 |
| RF06: O sistema deve permitir ao administrador cadastrar novos cursos; | UC-10 |
| RF07: O sistema deve permitir ao usuário cadastrar novos turnos de aulas; | UC-14 |
| RF08: O sistema deve permitir ao usuário cadastrar tipos de recursos presentes nas salas; | UC-03 |
| RF09: O sistema deve permitir a alocação de uma sala ou laboratório por turnos e horários diferentes; | UC-05 |
| RF10: O sistema deve permitir ao usuário a consulta de informações especificas registradas; | - |
| RF11: O sistema deve permitir ao administrador cadastrar novas disciplinas vigentes | UC-11 |
| RF12: O sistema deve permitir ao usuário gerar relatórios dos quadros de horários das salas e laboratórios; | UC-12 |
| RF13: O sistema deve emitir avisos de mudanças nos horários | - |

Fonte: Levantamento de Dados das entrevistas (2017)

### Requisitos Não Funcionais

São as “não funcionalidades” do sistema, ou seja, trata-se de algo que não é uma funcionalidade, mas que precisa ser realizado para que o software atenda seu propósito.

Ventura (2016) eles atendem aos requisitos do sistema que não se referem a funcionalidades do negócio), mas que fazem parte do escopo do sistema, podendo ou não estar associado a um Requisito Funcional.

O quadro 7 demonstra os requisitos não funcionais levantados para o sistema.

Quadro 7 – Requisitos Não Funcionais do Sistema

|  |
| --- |
| **Requisito** |
| RNF01: O sistema deve ser executado e um servidor que disponha de recursos como NginX® e SQLServer instalados e configurados. |
| RNF02: O sistema deve possuir acesso a redes TCP/IP. |
| RNF03: O sistema deve possuir suporte a PHP e JavaScript. |
| RNF04: Integração com o sistema de aplicativos da Unitis. |
| RNF05: A integridade dos dados deve ser garantida. |

Fonte: Levantamento de Dados das entrevistas (2017)

### Schema Banco de Dados

A figura 06 mostra o banco de dados e seus relacionamentos. O mesmo foi verificado e validado por Carlos Diego, DBA (Data Base Administrator) da UNITINS.

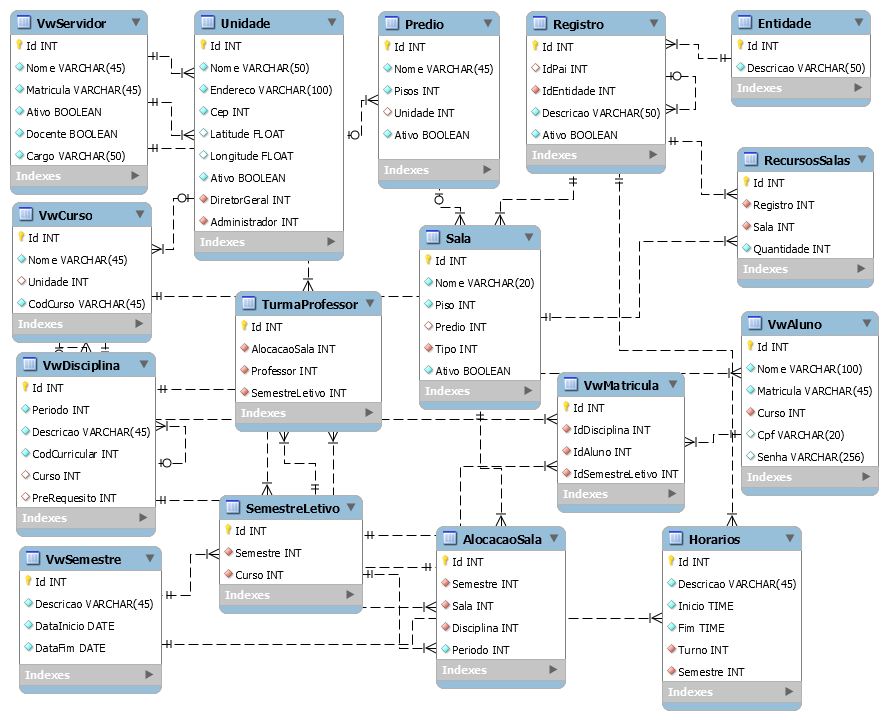


Figura 06 – Banco de dados e seus relacionamentos

Fonte: elaborado pelo autor utilizando o software MySql Work Bench.

### Protótipo de Telas

As telas foram desenhadas utilizando o software Balsamiq*™* Mockup*®*, com as mesmas sendo validadas pela administradora do campus Graciosa, Tais Bogo. Essas telas são o conceito de interface do software.

A figura 07 mostra o menu principal, com sua barra de tarefas

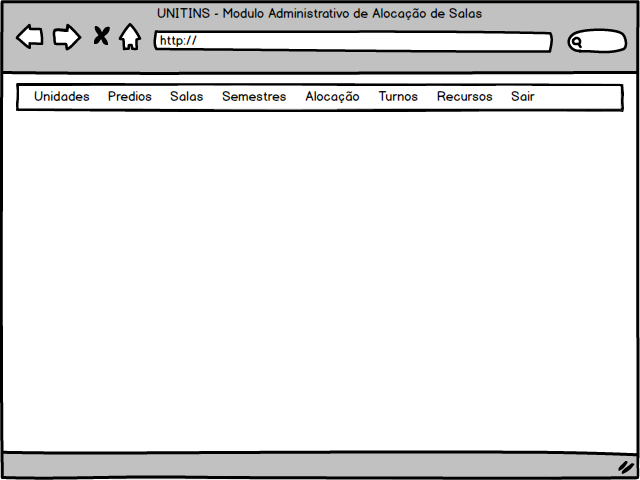


Figura 07 – Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 08 mostra o menu de salas, com suas opções e sua listagem principal.

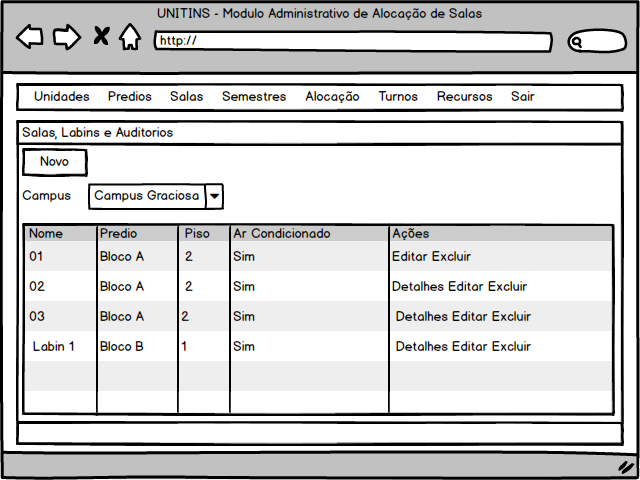


Figura 08 – Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 09 mostra o detalhamento de salas para visualização, mostrando seus recursos em uma listagem principal, bem como as informações da sala selecionada.

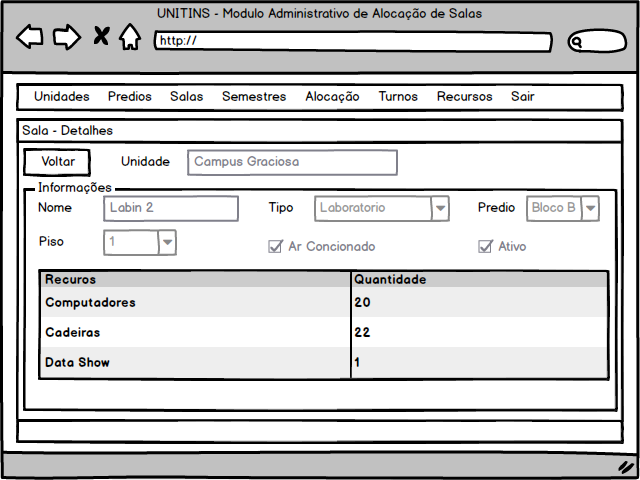


Figura 09 – Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 10 mostra o detalhamento de salas para edição e novo registro (os campos estão habilitados), mostrando seus recursos em uma listagem principal, bem como as informações da sala selecionada.

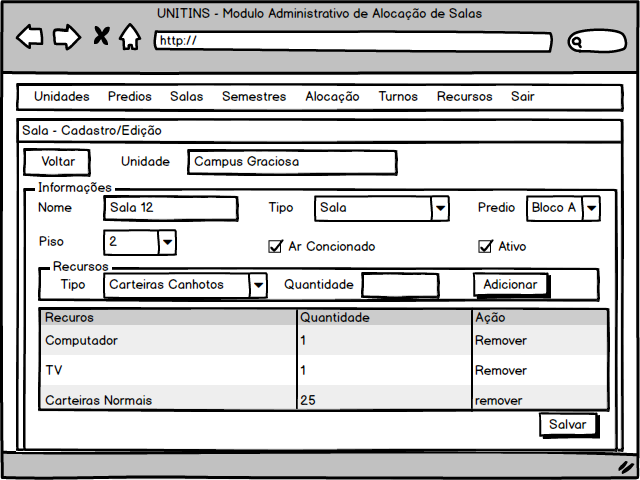


Figura 10 – Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 11 mostra os detalhes da alocação de um determinado curso, mostrando as alocações registradas no sistema e dando opções ao usuário (Ações).

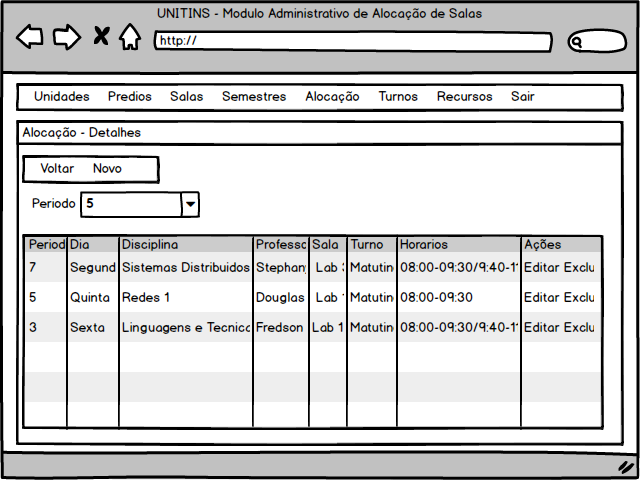


Figura 11 - Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 12 mostra a listagem geral de alocações por curso, dando opções ao usuário (Ações) de obter os detalhamentos mostrados na figura anterior.

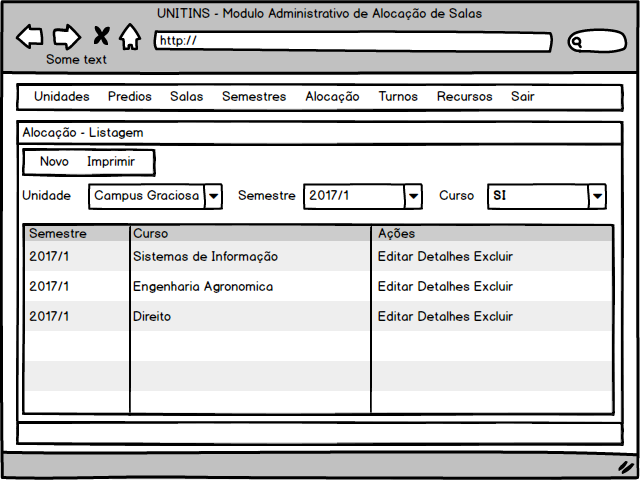


Figura 12 - Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

A figura 13 mostra o cadastro de uma nova alocação a partir de um semestre cadastrado.

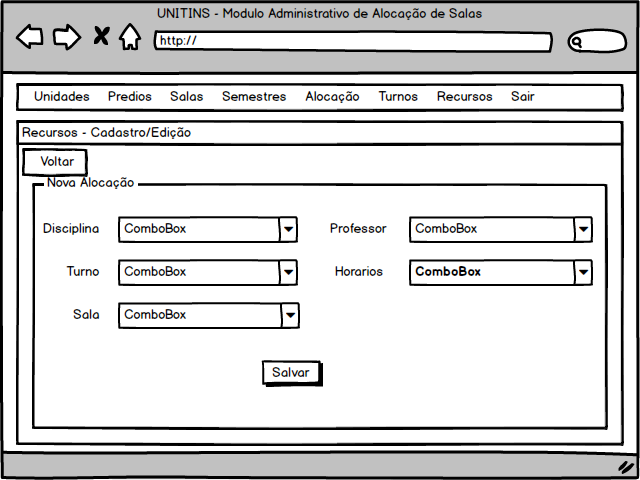


Figura 13 - Protótipo de tela

Fonte: elaborado pelo autor utilizando o software Balsamiq Mockup.

## DESENVOLVIMENTO

É referente ao desenvolvimento prático de todo o projeto que foi descrito na fase 2 e 3 da metodologia, ou seja, desde do desenvolvimento até os testes de produção.

### Arquitetura do Projeto

Toda a organização dos arquivos do projeto foi pensada a fim de obedecer aos paradigmas da arquitetura MVC citada no referencial teórico. A hierarquia de diretórios ficou organizada conforme a figura 14.

Os arquivos fonte do Bootstrap ficam nas pastas css (layout) e js (arquivos *javascript* para execução de componentes visuais), enquanto os arquivos fonte do JQuery ficam apenas na pasta js.

O diretório app é um dos diretórios mais importantes, pois nele ficam os arquivos de *controllers*, modelos e *views*. A pasta *controller* é aonde estão outros diretórios com o nome dos módulos do sistema (substantivos), dentro desses diretórios estão os arquivos php que executam as ações do modulo, com designações como listar, inserir, carregar combo, deletar e detalhar (verbos).

A pasta model estão os modelos gerados pelo Doctrine, esses modelos são instanciados e manipulados nos *controllers* citados anteriormente, e na pasta *view* estão os arquivos *JavaScript* utilizados pelo JQuery, com cada arquivo sendo correspondente a um modulo do sistema.

Os arquivos de configuração do Doctrine e suas dependências ficam na pasta principal, na pasta vendor e na pasta bin.

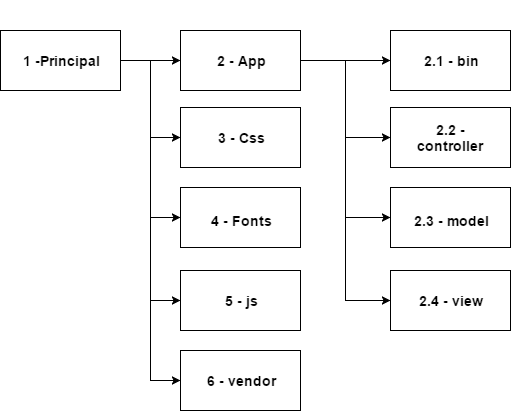


Figura 14 – Arquitetura do Projeto

Fonte: elaborado pelo autor pelo web site www.draw.io.

### Arquivos do Projeto

Dentro da hierarquia de diretórios (que funcionam como pacotes nesse projeto), existe a hierarquia de arquivos do projeto. Dois arquivos em especial são necessários para todas as páginas, sendo eles o *top.php* e o *bottom.php*.

Todos os arquivos restantes de layout, como *index.php*, *unidades.php* e p*redio.php* herdam esses dois arquivos em seus conteúdos (*header* e *bottom*, respectivamente), fazendo assim com que cada mudança no menu e no rodapé da página sejam feitas apenas em seus arquivos fonte, e não em cada página que as herda, conforme mostra a figura 15.

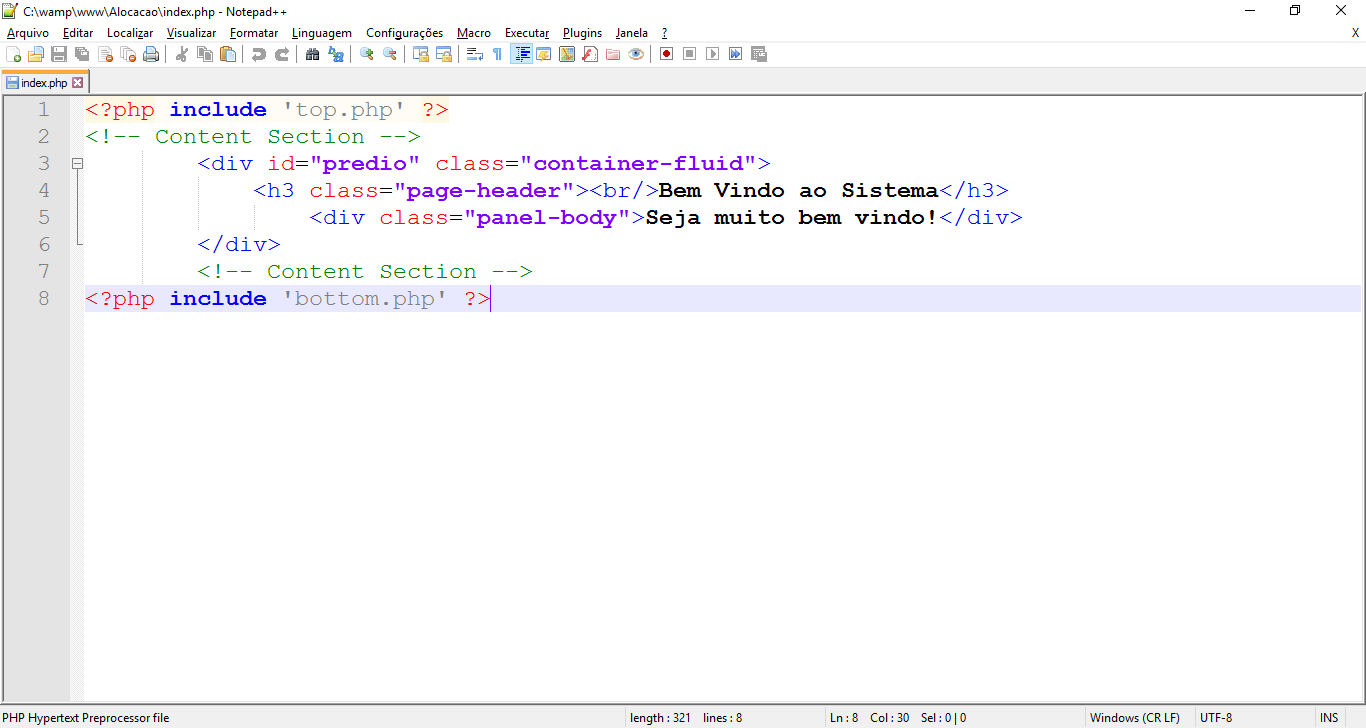


Figura 15 – Arquivo com herança

Fonte: elaborado pelo autor.

A figura 16 mostra o layout final do sistema na tela de salas, demonstrando como funcionou na prática todos os elementos reunidos.

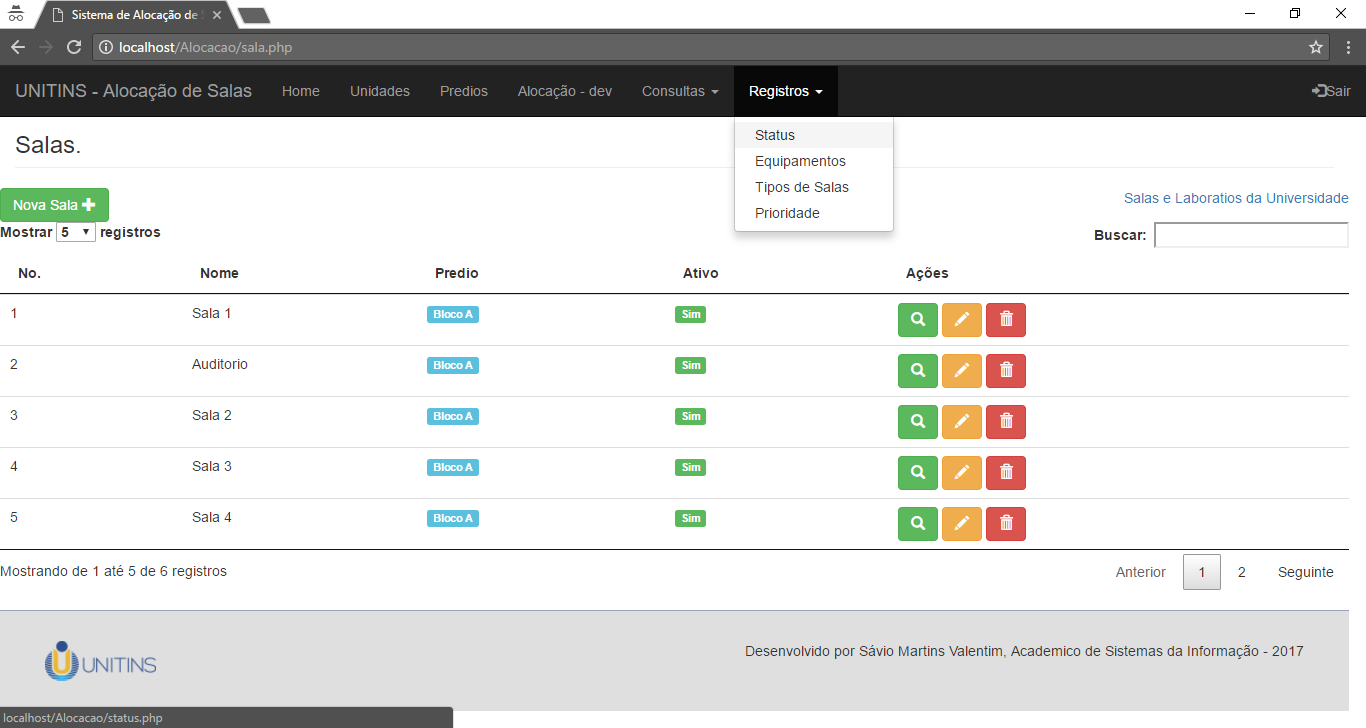


Figura 16 – Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 17 mostra o layout final do sistema na tela de Unidades (ou campus), demonstrando a listagem e as opões de ações, mostrando os ícones correspondentes a cada ação (detalhar, editar e excluir), de busca da listagem, a paginação e o menu, que é padrão para todas as telas.

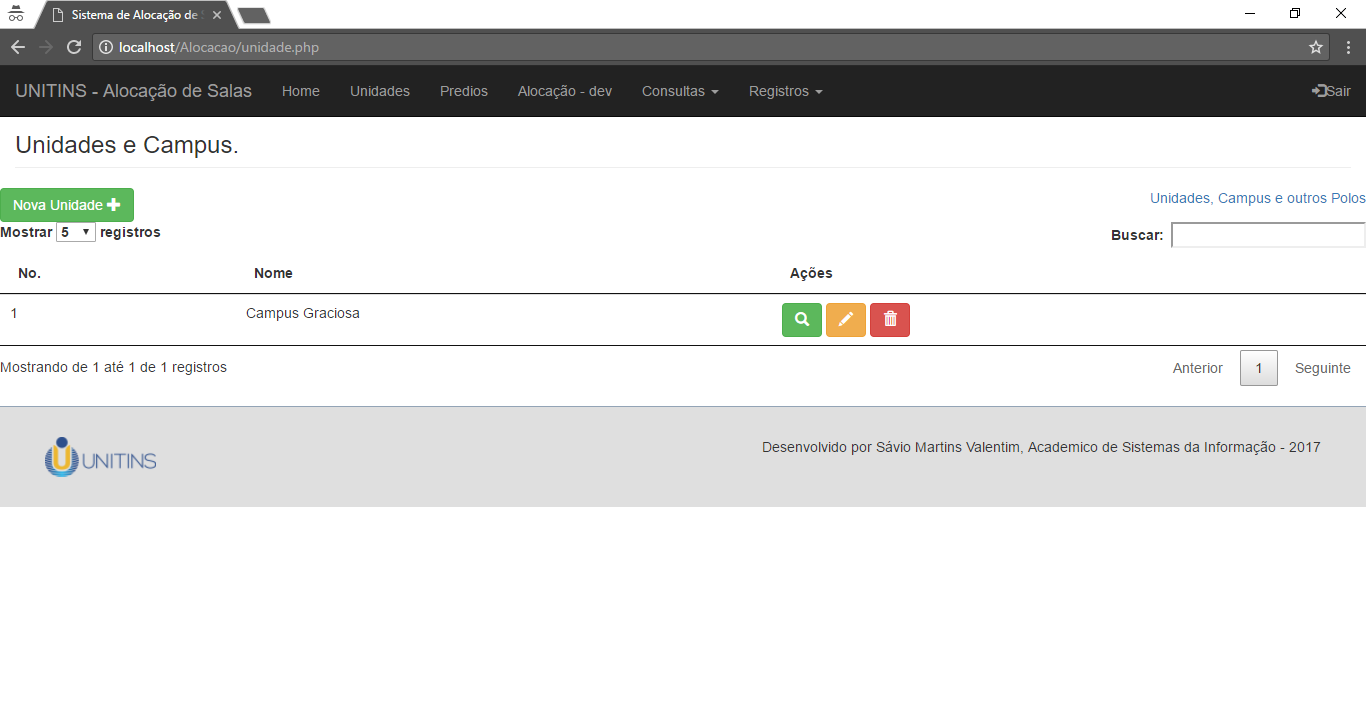


Figura 17 – Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 18 mostra o modal utilizado para inserir e editar registros, todas as telas seguem o mesmo padrão, mudando os parâmetros utilizados conforme cada tela.

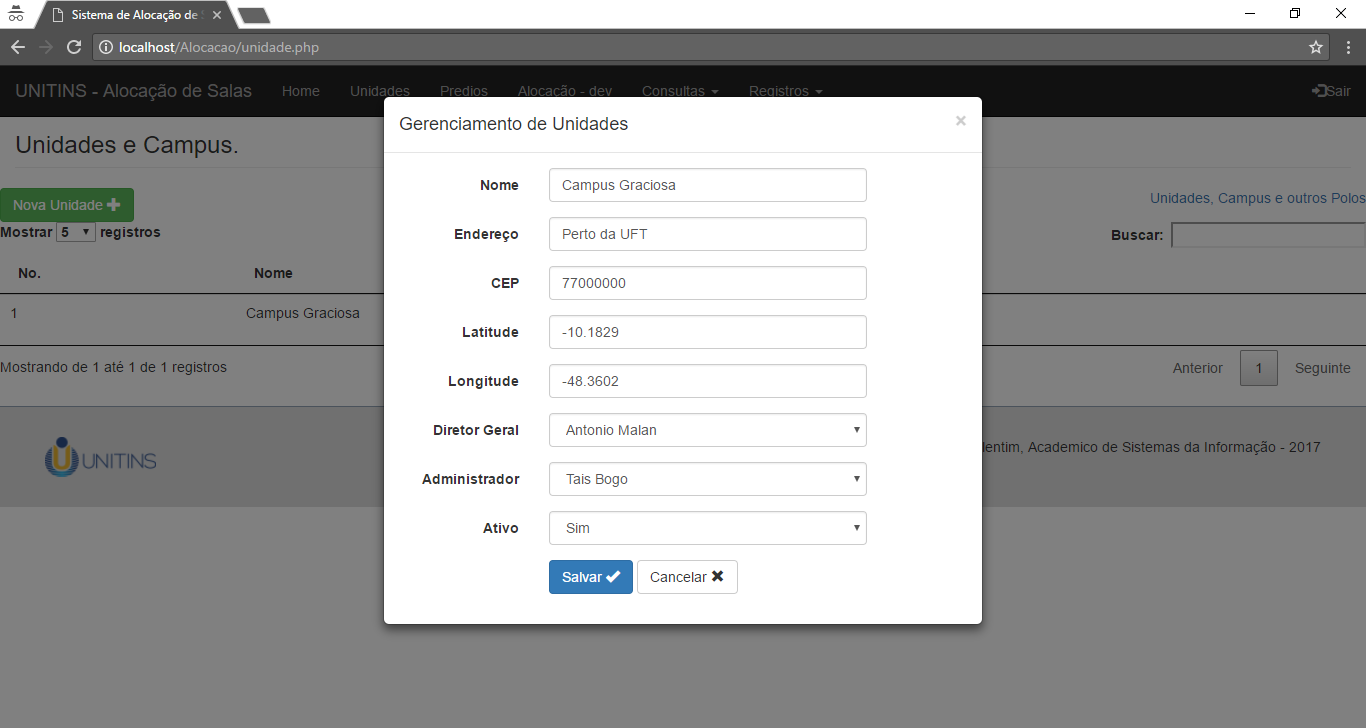


Figura 18– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 19 mostra as caixas de diálogo pedindo confirmação antes de ações críticas de exclusão e desvinculação de registros. Essas caixas de diálogo também são utilizadas para informar mensagens de status, como sucesso, erro ou falha em alguma operação.

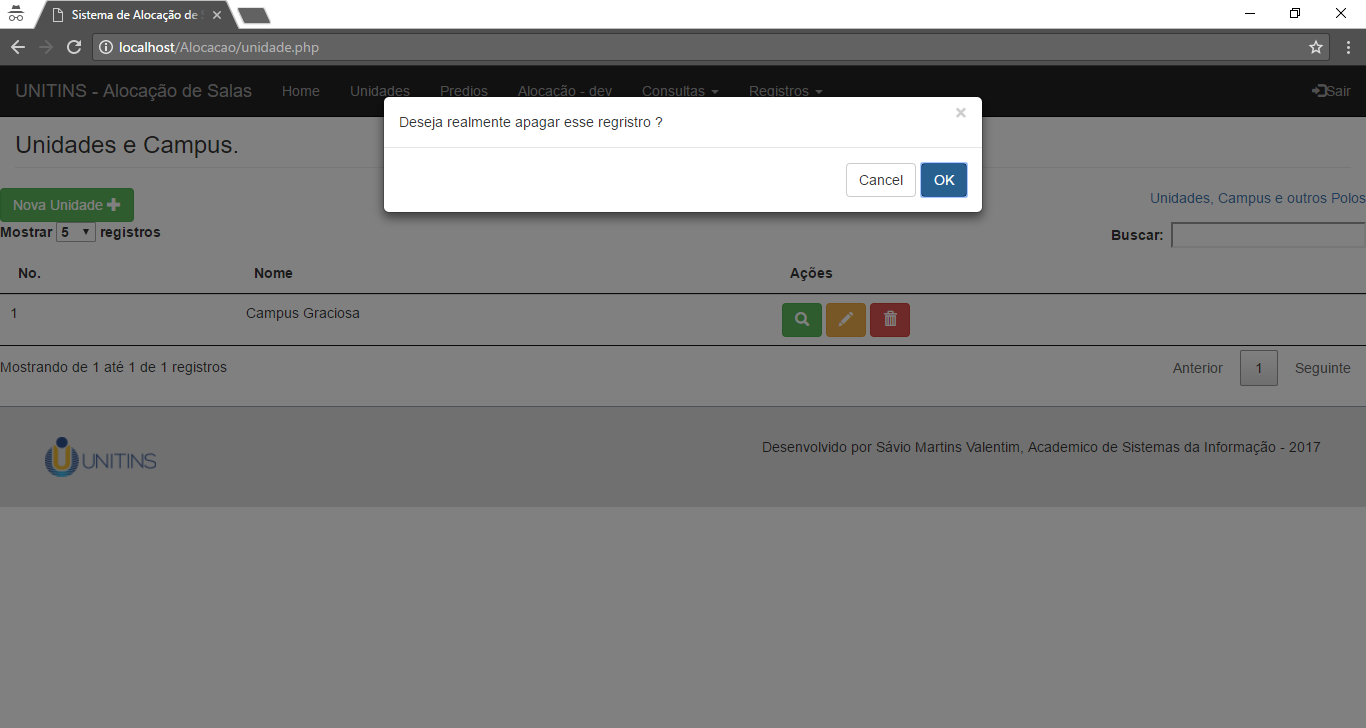


Figura 19– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 20 mostra uma tela de detalhamento de uma entidade (no exemplo, uma Unidade). Esse detalhamento mostra os registros vinculados ao detalhamento, com a categoria da entidade vinculada separada por abas.

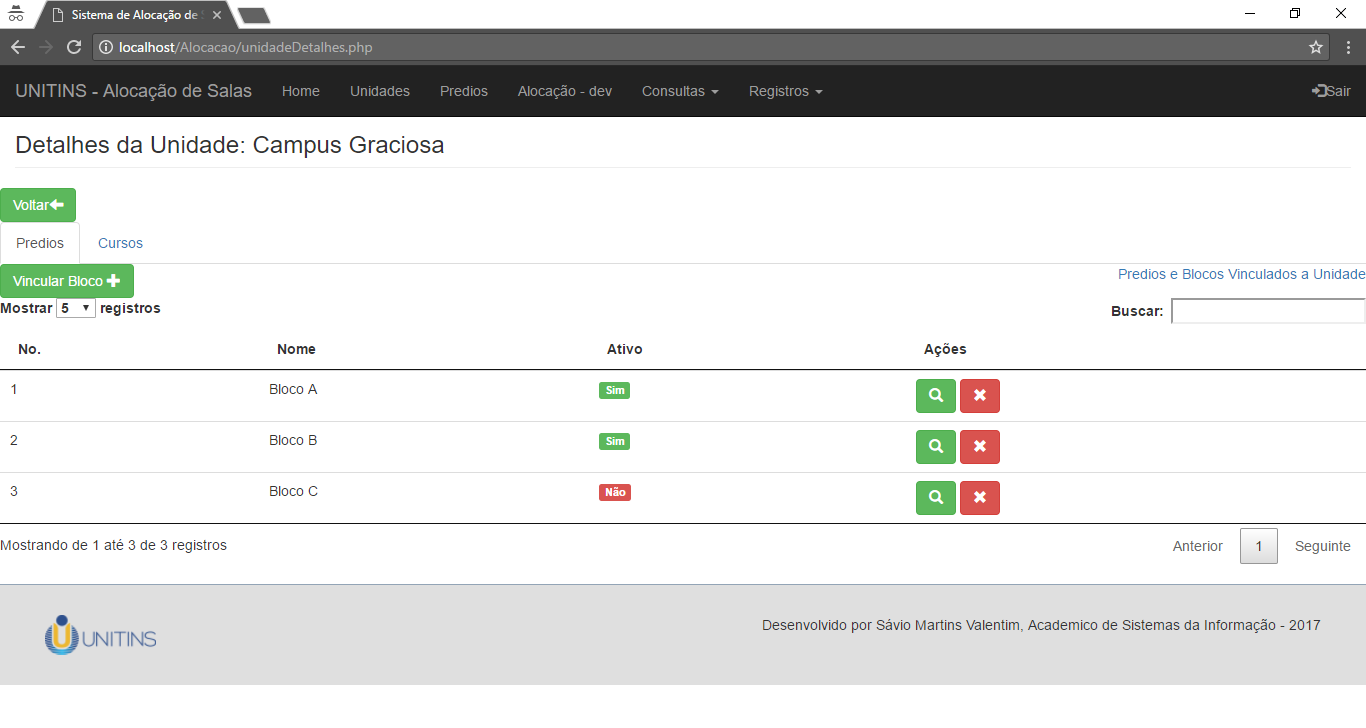


Figura 20– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 21 mostra o funcionamento dos menus *drop-down,* além do rodapé que se posiciona de acordo com o conteúdo da página, podendo ficar mais recolhido ou mais estendido, dependendo do espaço ocupado pelas tabelas..

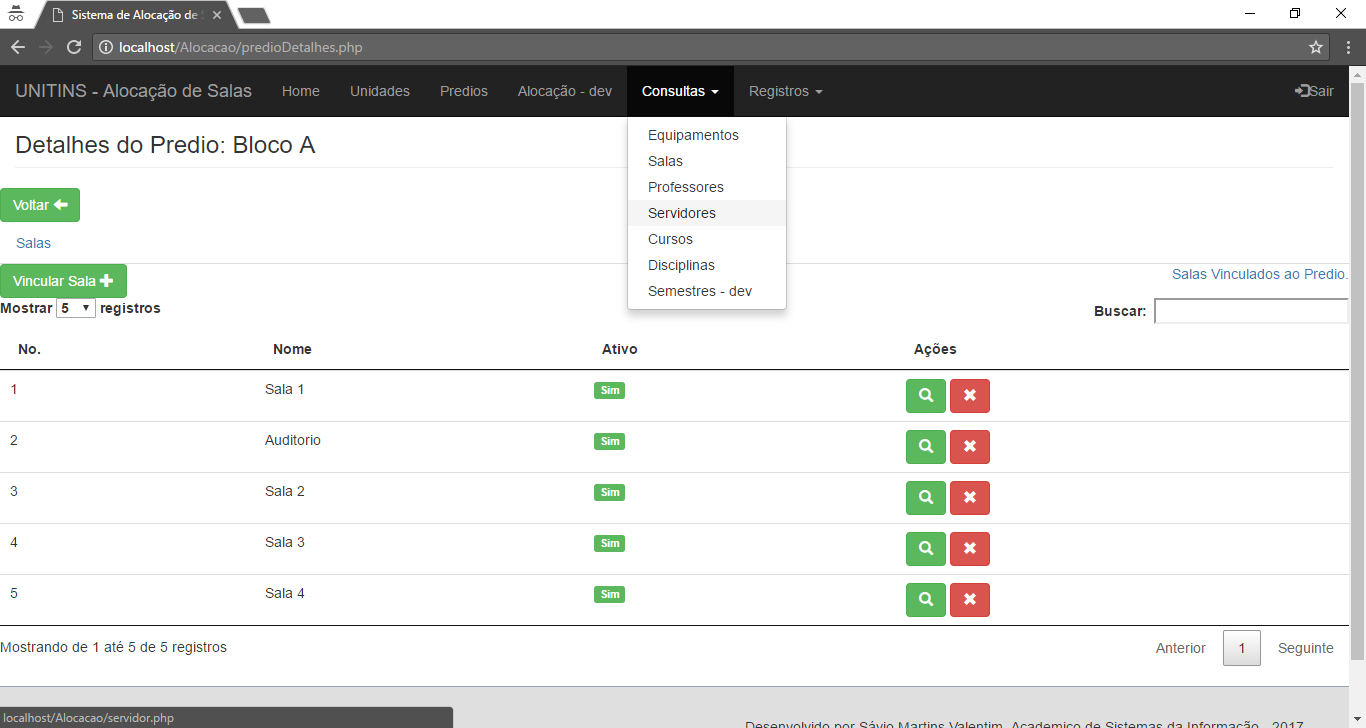


Figura 21– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 22 mostra o menu de ações da tabela, em alguns módulos ele possuí apenas uma opção, enquanto alguns possuem duas opções (como na figura), ou três opções como na figura 17, dependendo da necessidade do modulo em específico.

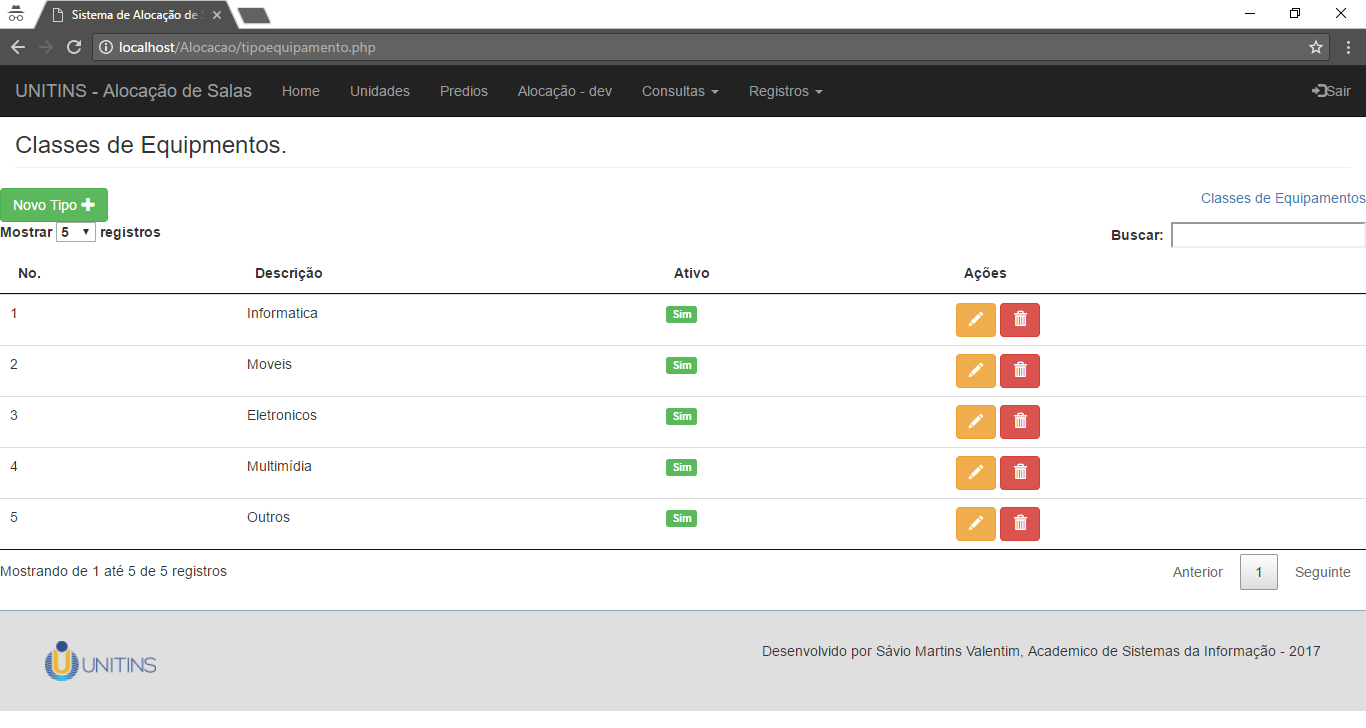


Figura 22– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 23 a inserção de registros que possuem dependência de outros registros no sistema. Sempre que houver a necessidade de inserir registros dessa forma, o registro virá através de combo-box, que é uma caixa com opções pré-carregadas.

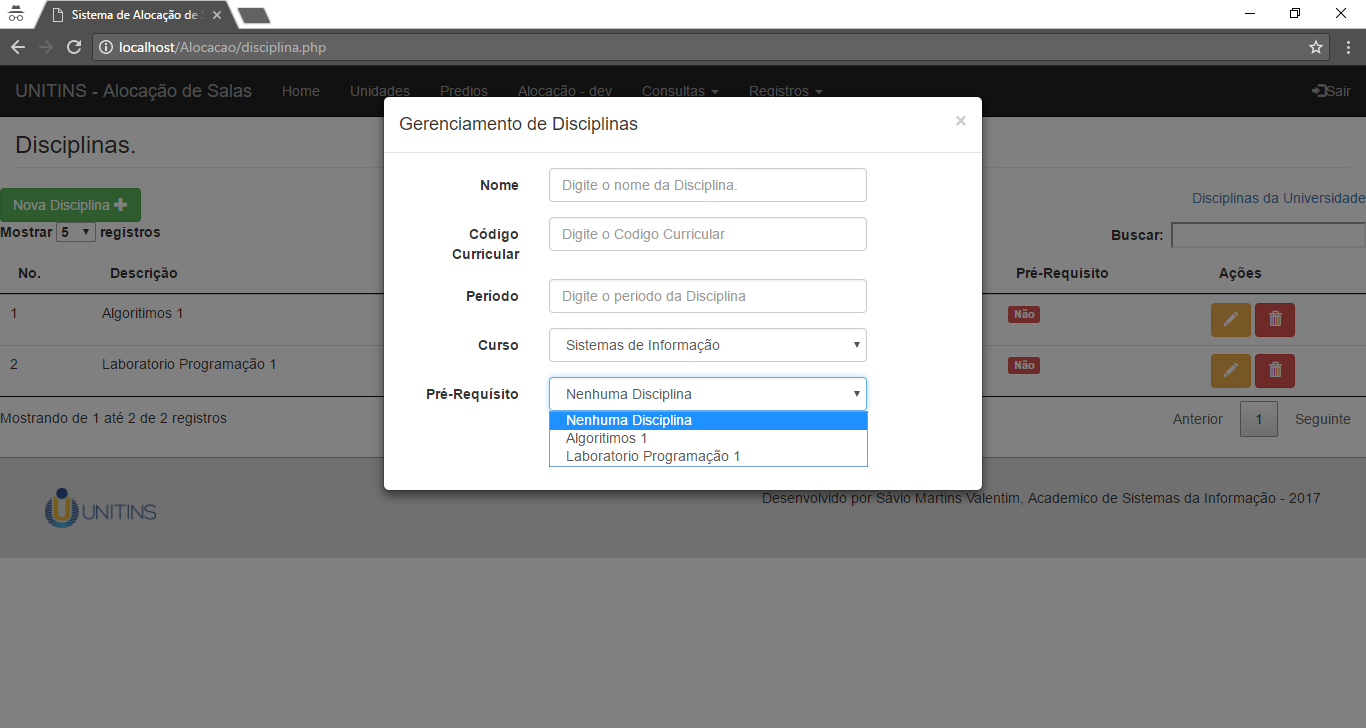


Figura 23– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

A figura 24 mostra a listagem registros ativos e inativos. Essa opção teve por objetivo manter a integridade de dados do sistema, pois um registro pode ser desativado e mantido no sistema, sem afetar assim os registros que possuem dependência do mesmo.

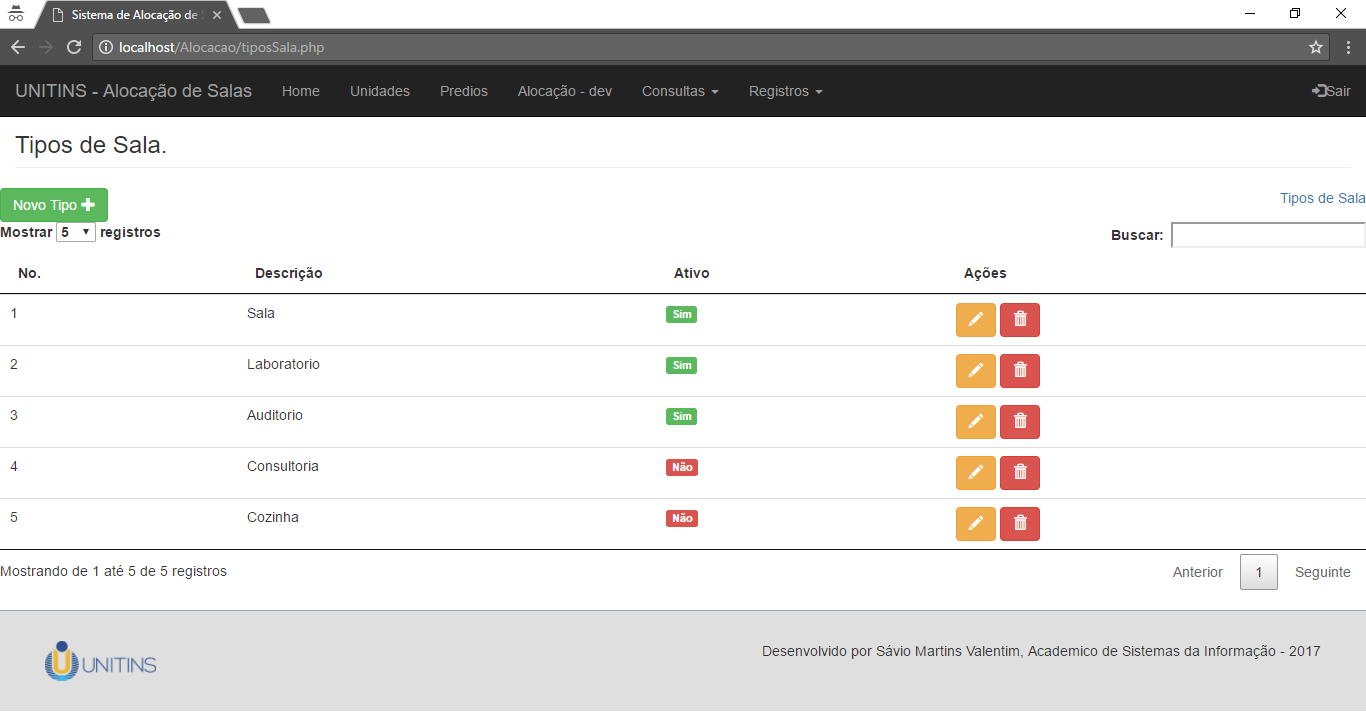


Figura 24– Layout final do projeto em execução.

Fonte: elaborado pelo autor.

# CONCLUSÃO

Desde o começo esse projeto gerou boas expectativas em todos os envolvidos. O primeiro ponto é que o campus, principalmente o pessoal da administração, necessita de novos métodos e ferramentas para agilizar e organizar seus trabalhos, que atualmente ficam muito centralizados nas mãos de uma única pessoa.

O segundo ponto é a questão do aprendizado, essas demandas podem ser supridas por acadêmicos do próprio curso, o que geraria um bom portfolio para os novos profissionais e ajudaria a solucionar alguns problemas da instituição.

O terceiro ponto é o incentivo para a fábrica de software no curso de Sistemas de Informação, uma ideia que vem sendo trabalha pelo menos desde 2016, mas que até esse presente momento (2017) não saiu do papel.

O sistema foi todo projetado para atender a uma necessidade do campus, visando criar um sistema de mapa de salas para facilitar o serviço do administrativo em realizar o mapeamento e informar aos usuários, em um modulo separado, as informações a respeito da alocação.

Devido ao atraso no projeto por conta da troca de uma das ferramentas, não foi possível finalizar o sistema (pulando assim a fase de implantação), porem toda a experiência e aprendizado não podem ser deixados de lado.

A experiência de projetar um sistema do zero, escolhendo as ferramentas e fazendo todos os requisitos, telas e banco de dados foi uma grande experiência de como é o desenvolvimento de software no mundo real, trabalhando com tecnologias muito utilizadas comercialmente, como o PHP, o *JavaScript*, o JQuery e o Bootstraps.

O atraso no projeto também serviu para demonstrar o quão impactante são os problemas e as dificuldades encontradas, como as mudanças de ferramentas e métodos causam atrasos e a maneira como ele impacta no cronograma do projeto, por conta disso o mesmo não foi 100% concluído, faltando o modulo de alocação e geração de relatórios.

Todos os resultados ao final somaram muito ao aprendizado, demonstrando na prática como é o desenvolvimento de um sistema e todas suas variáveis, sendo elas positivas (execução, levantamento de requisitos, etc) ou negativas (atrasos, mudanças de ferramentas, etc).

# REFERÊNCIAS

Bootstrap. **Documentação e Site oficial.** Disponível em: <http:// http://getbootstrap.com//>. Acessado em: 30 de mai. 2017.

Desenvolvimento Ágil de Software. **Scrum.** Disponível em: < http://www.desenvolvimentoagil.com.br/scrum/>. Acessado em: 28 de mar. 2017.

DevMedia. **Introdução ao Padrão MVC.** Disponível em: < http://www.**devmedia**.com.br/introducao-ao-padrao-mvc/29308>. Acessado em: 25 de mar. 2017.

Doctrine 2. **Documentação e Site oficial.** Disponível em: <http://www.doctrine-project.org/>. Acessado em: 25 de mar. 2017.

JQuery. **Documentação e Site oficial.** Disponível em: <https://jquery.com//>. Acessado em: 30 de mai. 2017.

LEITE, Jair. **Engenharia de Software: O Modelo Espiral**, 2007. Disponível em:< http://engenhariadesoftware.blogspot.com.br/2007/03/o-modelo-espiral.html/>. Acessado em: 25 de mar. 2017.

Microsoft. **Microsoft SQL Server.** Disponível em: < https://msdn.microsoft.com/pt-br/library/bb545450.aspx/>. Acessado em: 25 de mar. 2017.

NGinx. **Documentação e Site oficial**. Disponível em: <http://nginx.org/>. Acessado em: 25 de mar. 2017.

PHP. **Documentação e Site oficial**. Disponível em: <https://secure.php.net/>. Acessado em: 19 de mar. 2017.

VENTURA, Plinio. **Requisitos Funcionais, Casos de Uso em Engenharia de Software**, 2016. Disponível em:< http://www.ateomomento.com.br//>. Acessado em: 10 de mar. 2017.

# ANEXOS

Anexo 1 – Ata de Reunião com o Administrativo do Campus

|  |  |  |  |
| --- | --- | --- | --- |
| **Sistema:** Sistema de Alocação Semestral de Salas | | | |
| **ATA DE REUNIÃO** | | | |
| **Código:** 1 | **Data:** 14/03/2017 | **Hora início:** 10:40 | **Hora término:** 11:30 |
| **Entrevistador (es):** Sávio Martins Valentim | | | |
| **Entrevistado (s):** Tais Bogo – Administração da UNITINS Campus Graciosa | | | |
| **Assunto:** Definição dos requisitos do sistema | | | |
| **Descrição:**   * A UNITINS – Universidade Estadual do Tocantins é uma universidade que necessita de inovações no gerenciamento de seus recursos, sendo um deles um sistema para alocação semestral de salas, laboratórios e horários. Atualmente o sistema é completamente manual e gerado a partir de uma tabela do Excel compartilhada via google drive. * O sistema web deve permitir o gerenciamento de salas (e seus recursos), laboratórios (e seus recursos), horários, cursos, bloco, campus e semestre letivo, de modo a manter todas as informações armazenadas em um banco de dados e, assim, agilizar a alocação e a consulta das salas alocadas. * O sistema Web deve ser baseado em um banco de dados gratuito, de modo que não haja despesas extras para a universidade. Neste banco de dados serão cadastradas todas as salas, laboratórios, recursos, horários, cursos, bloco, campus e semestre letivo. * Para as salas e laboratórios devem ser mantidas os seguintes dados: número, bloco, piso, número de carteiras e recursos disponíveis (TV’s, computadores, etc.). Para os blocos deverá ser cadastrado o número de salas e laboratórios, bem como o campus a qual pertencem e seu número. Para o campus é importante que sejam armazenadas informações como nome, número de blocos, endereço e coordenadas geográficas para geolocalização futura. Para os semestres é necessário a nomenclatura, data de começo, data de fim, * Para realizar operações de gerenciamento de dados no sistema o administrador deverá estar cadastrado e fornecer o seu login e senha para entrar no sistema. * A senha de acesso ao sistema permite ao funcionário cadastrar, editar e excluir salas, laboratórios e blocos de seu respectivo campus, bem como gerenciar a entrada semestres e alocação das salas para os mesmos. Além disso, um funcionário logado poderá realizar consultas dos dados e emitir históricos e registros de alocação. * Ao realizar a entrada de uma alocação, o administrador deverá especificar o semestre, a sala, o curso e o turno que está sendo alocado. Não pode haver choque entre horários alocados por turmas diferentes. | | | |

Anexo 2 – Ata de Reunião com a DTIC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sistema:** Sistema de Alocação Semestral de Salas | | | | **Versão:** 1.0 |
| **ATA DE REUNIÃO** | | | | |
| **Código:** 1 | **Data:** 22/03/2017 | **Hora início:** 16:45 | **Hora término:** 17:30 | |
| **Entrevistador (es):** Sávio Martins Valentim, Yzack | | | | |
| **Entrevistado (s):** Leandra, Marcio, Carlos – DTIC UNITINS Sede | | | | |
| **Assunto:** Definição da Infraestrutura do Projeto | | | | |
| **Descrição:**   * Foi marcada uma reunião na sede da UNITINS no dia 22/03/2017 na DTIC, para serem acertados os detalhes de infraestrutura, como o servidor de hospedagem, tipo de banco e comunicação com a base dos dados da própria instituição. * Decidiu-se utilizar um servidor que suporta a linguagem PHP, o NGinX, com o web service sendo hospedado por virtualização. Os motivos da escolha foram:   + É um serviço que já é utilizado pela instituição, o que facilita a alocação de um container;   + Em caso de problemas de execução do serviço, os profissionais do setor área poderão verifica-lo e corrigi-lo; * O banco de dados utilizado será o SQL Server 2008 R2, o mesmo usado pela instituição, com o banco sendo armazenado no mesmo servidor da instituição; * Haverá suporte técnico (alocação de recursos para o servidor, etc.) por parte da DTIC, com a mesma concedendo permissões para acesso ao sistema via o sistema da instituição; * Algumas informações utilizadas pelo sistema serão consumidas a partir de informações já existentes na base de dados da instituição; * Espera-se que esse projeto sirva de exemplo para os objetivos da futura “Fábrica de Software”, idealizada pela instituição | | | | |