**UNIVERSIDADE PAULISTA**

**BRUNA PESSINI BIUDES SILVEIRA MOLINA**

**CARLOS AUGUSTO SPECCHI**

**MATHEUS CRUZ PEREIRA**

**PATRICK ELIAS DA SILVA**

**ROSANA SANTOS DE ABREU**

**LOCALIZUNIP**

**Sorocaba/SP  
2018**

**BRUNA PESSINI BIUDES SILVEIRA MOLINA**

**CARLOS AUGUSTO SPECCHI**

**MATHEUS CRUZ PEREIRA**

**PATRICK ELIAS DA SILVA**

**ROSANA SANTOS DE ABREU**

**LOCALIZUNIP**

Trabalho de Conclusão do Projeto Interdisciplinar apresentado à Universidade Paulista em Sorocaba/SP.

Orientadores: Prof. Esp. Richardson kennedy Luz, Prof. Esp. Eduardo Martins Simi, Prof.Esp. Marcos Domingos da Cruz, Prof. Esp. Randal Gasparini e Prof. Esp. Reverdan Almeida Sparinger.

**Sorocaba/SP**

**2018**

**BRUNA PESSINI BIUDES SILVEIRA MOLINA**

**CARLOS AUGUSTO SPECCHI**

**MATHEUS CRUZ PEREIRA**

**PATRICK ELIAS DA SILVA**

**ROSANA SANTOS DE ABREU**

**LOCALIZUNIP**

Trabalho de Conclusão do Projeto Interdisciplinar apresentado à Universidade Paulista em Sorocaba/SP.

Orientadores: Prof. Esp. Richardson kennedy Luz, Prof. Esp. Eduardo Martins Simi, Prof.Esp. Marcos Domingos da Cruz, Prof. Esp. Randal Gasparini e Prof. Esp. Reverdan Almeida Sparinger.

Aprovado em:

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Richardson kennedy Luz

Universidade Paulista – UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Eduardo Martins Simi

Universidade Paulista – UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof.Esp. Marcos Domingos da Cruz

Universidade Paulista UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Randal Gasparini

Universidade Paulista UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Reverdan Almeida Sparinger

Universidade Paulista UNIP

**DEDICATÓRIA**

Dedicamos este trabalho primeiramente a Deus por ter nos concedido a sabedoria de poder concluir este projeto, ao corpo docente da Universidade Paulista em Sorocaba/SP.

**AGRADECIMENTOS**

Agradecemos este, bеm como todas às nossas demais conquistas, аоs nossos pais, pelo apoio e incentivo.

Obrigado.

**RESUMO**

O presente trabalho apresenta um aplicativo cuja finalidade é promover a localização dos alunos dentro do Campus da Universidade Paulista em Sorocaba, a necessidade surgiu de auxiliar os novos alunos no inicio do semestre, a dirigirem-se aos seus respectivos prédios, salas de aulas.

Palavras-chave: Aplicativo, Sistema, Android, Plataforma, Web.

**ABSTRACT**

The present work aims to present an application that will serve to promote the location of students and visitors within the Campus of the Paulista University in Sorocaba, which arose from the need to help new students to go to their respective buildings, classrooms, laboratories, library, auditorium, especially at the beginning of the semester. Upon accessing the application, students will have access to the period, course, semester, block and room that they will have to go to and when they access the auditoriums, they will have access to the agenda for checking the lectures and events scheduled to take place. The application must be installed on mobile devices, equipped with Android system, updates of the rooms, as well as the agenda of the auditoriums will be updated by the administrator through access to the Web platform.

Keywords: Application, System, Android, Platform, Web.

SUMÁRIO

[1 INTRODUÇÃO 10](#_Toc516014410)

[2 Modelagem de Software 11](#_Toc516014411)

[2.1 Linguagem de Modelagem Unificada - UML 11](#_Toc516014412)

[2.2 Diagramas da UML 12](#_Toc516014413)

[2.3 Atividades típicas de um processo de desenvolvimento 13](#_Toc516014414)

[2.3.1 Requisitos 13](#_Toc516014415)

[2.3.2 Requisitos Funcionais 15](#_Toc516014416)

[2.3.3 Requisitos Não Funcionais 15](#_Toc516014417)

[2.4 Levantamento e analise dos requisitos 16](#_Toc516014418)

[2.5 Validação dos requisitos 16](#_Toc516014419)

[2.6 Revisão dos requisitos 17](#_Toc516014420)

[2.7 Participantes do processo 17](#_Toc516014421)

[2.7.1 Gerente de Projetos 17](#_Toc516014422)

[2.7.2 Analistas 17](#_Toc516014423)

[2.7.3 Projetista 18](#_Toc516014424)

[2.7.4 Arqueiteto de software 18](#_Toc516014425)

[2.7.5 Programadores 18](#_Toc516014426)

[2.8 Modelo de Processo 18](#_Toc516014427)

[2.8.1 Modelo Cascata 19](#_Toc516014428)

[2.9 Desenvolvimento Revolucionario 20](#_Toc516014429)

[2.10 Desenvolvimento Incremental 21](#_Toc516014430)

[2.11 Modelo Espiral 22](#_Toc516014431)

[3 Modelagem de casos de uso 23](#_Toc516014432)

[3.1 Casos de uso 24](#_Toc516014433)

[3.1.1 Cenários 24](#_Toc516014434)

[3.1.2 Atores 25](#_Toc516014435)

[3.1.3 Relacionamentos 25](#_Toc516014436)

[3.2 Diagrama de Caso de Uso 26](#_Toc516014437)

[3.3 Diagrama de classes 26](#_Toc516014438)

[3.4 Diagrama de Sequencia 27](#_Toc516014439)

[3.5 Diagrama de Atividade 28](#_Toc516014440)

[3.5.1 Fluxos de controle sequenciais 28](#_Toc516014441)

[3.5.2 Fluxos de controle paralelos 28](#_Toc516014442)

[4 TOPOLOGIA DE REDES 29](#_Toc516014443)

[4.1 Tipos de conexão 29](#_Toc516014444)

[4.1.1 Ponto a Ponto 29](#_Toc516014445)

[4.1.2 Multiponto 30](#_Toc516014446)

[4.2 Tipos de Topologia 30](#_Toc516014447)

[4.2.1 Topologia de Barramento 30](#_Toc516014448)

[4.2.2 Topologia Estrela 31](#_Toc516014449)

[4.2.3 Topologia Anel 32](#_Toc516014450)

[4.2.4 Topologia Hibrida ou Mista 33](#_Toc516014451)

[4.2.5 Topologia em Malha 34](#_Toc516014452)

[4.2.6 Topologia Ponto-a-Multiponto 35](#_Toc516014453)

[4.2.7 Topologia do PIM 36](#_Toc516014454)

[5 Ferramentas 36](#_Toc516014455)

[5.1 HTML5 36](#_Toc516014456)

[5.2 Bootstrap 37](#_Toc516014457)

[5.3 CSS3 38](#_Toc516014458)

[5.4 JavaScript 39](#_Toc516014459)

[5.4.1 Camadas de desenvolvimento 40](#_Toc516014460)

[5.5 Microsoft Visio 40](#_Toc516014461)

[5.5.1 Interface e funções do programa 40](#_Toc516014462)

[5.5.2 Vantagens do Visio para sua organização 41](#_Toc516014463)

[5.6 PhoneGAP 41](#_Toc516014464)

[5.7 Sublime Text 42](#_Toc516014465)

[5.8 Xampp 43](#_Toc516014466)

[5.9 Pencil Wireframe 43](#_Toc516014467)

[5.10 MER 43](#_Toc516014468)

[5.11 DER 44](#_Toc516014469)

[6 PSEUDOCODIGO 46](#_Toc516014470)

[7 FLUXOGRAMA LOCALIZA-SE 48](#_Toc516014471)

[8 WIREFRAME 49](#_Toc516014472)

[9 MAPA DO SITE 51](#_Toc516014473)

[10 MANUAL DO APLICATIVO 60](#_Toc516014474)

[11 CONSIDERAÇÕES FINAIS 62](#_Toc516014475)

[12 REFERÊNCIAS BIBLIOGRÁFICAS 63](#_Toc516014476)

[13 APÊNDICE A – DOCUMENTO DE REQUISITO DO SISTEMA 64](#_Toc516014477)

[14 APÊNDICE B – CODIGO FONTE 67](#_Toc516014478)

# INTRODUÇÃO

A proposta deste trabalho surgiu da necessidade de auxiliar os novos alunos a dirigirem-se aos seus respectivos prédios, salas de aulas, aos laboratórios, biblioteca, auditório, principalmente no início do semestre.

O uso da tecnologia tornou se a ferramenta para o desenvolvimento de um aplicativo com a função de auxiliar os alunos, especialmente os novos e os visitantes, uma vez que normalmente os mesmos encontram dificuldade para chegarem as suas respectivas salas, considerando que a Universidade trata se de um ambiente ainda desconhecido.

A justificativa é que tanto os novos alunos como os visitantes, conhecem apenas uma parte do local, ficam atrelados a secretaria e o prédio principal, ainda que haja outros prédios próximos, há uma dificuldade de localização e a necessidade do desenvolvimento do aplicativo vem da falta de uma ferramenta que auxilie indicando o local para onde deve dirigir-se.

O presente trabalho também tem como objetivo o desenvolvimento de um software que visa a informatização dos trabalhos manuais desempenhados pela chefia de Campus, em relação as salas de aula, tais como: mudança de sala e localização da melhor sala para uma determinada quantidade de alunos, a disponibilização da localização para os profissionais e alunos em meio a troca de sala e redução de gastos com papeis e profissionais.

Surgiu a necessidade de gerenciar as informações de uma forma adequada eficiente, com o objetivo de dar suporte e melhorar as informações que nela fluem, com o objetivo de melhorar a produtividade nos processos.

O trabalho está estruturado em forma de capítulos, começa tratando de questões específicas, aborda questões voltadas a Engenharia de Software, metodologias, introdução da UML, descrevemos as principais atividades de um processo de desenvolvimento de software, os principais profissionais envolvidos e suas respectivas atribuições, passando para análise de requisitos e modelos de processo.

Em seguida, há o modelo de casos de uso e os diversos elementos para construção do diagrama, o modelo de classes e os principais elementos de notação definidos pela UML para a construção desse diagram e por fim o diagramas de atividades.

Segue para a definição da topologia, características dos hardwares necessários para a instalação e o funcionamento do aplicativo, e logo após, faz referência das tecnologias utilizadas para o desenvolvimento do aplicativo, seguindo, vem o MER e DER, o Fluxograma, Pseudocódigo, a seguir temos o Wireframe, onde é mostrado o protótipo das telas e após, o manual para o usuário do site e do aplicativo.

# Modelagem de Software

O presente capítulo tem como objetivo motrar a complexidade do desenvolvimento de sistemas de software, uma vez que compreende todas as atividades necessárias para defiir, desenvolver, testar e manter um produto de software.

Segundo Bezerra (2007, p. 2) “[...] a complexidade de seu desenvolvimento aumenta à medida que cresce o tamanho do sistema, sendo assim, há uma necessidade de um planejamento inicial, essa necessidade leva o conceito de modelo, que pode ser visto como uma representação idealizada de um sistema a ser construído.

Bezerra afima que “[..] os principais motivos para utilizarem os modelos é pelo fato de que há uma limitação do ser humano em lidar com a complexidade [...]” e através dessa maneira, os envolvidos podem fazer estudos e prever comportamentos do sistema que está sendo desenvolvido.

Neste momento, os detalhes irrelevantes que podem gerar dúvidas, são ignorados por um momento e estuda-se separado, serve também para promover a difusão de informações entre os individuo envolvidos.

Um dos objetivos do uso dos modelos é o de promover a redução dos custos, tendo em vista que neste momento a correção dos erros é menos cara e sendo assim, erros detectados neste momento, possuem um impacto menos desastroso.

Pode-se prever o comportamento do sistema através da análise dos modelos, para Bezerra, “[...] Os modelos servem como um “laboratório”, em que diferentes soluções para um problema relacionado à construção do sistema podem ser experimentadas [...]” (BEZERRA, 2007, p.3).

Há diversos modelos cujos componentes são desenhos gráficos que seguem algum padrão lógico, e esses desenhos são denominados diagramas, sendo assim, o diagrama é apresentação de uma coleção de elementos gráficos que possuem um significado predefinido.

## Linguagem de Modelagem Unificada - UML

Segundo Bezerra (2007, p. 15) “[...] UML é uma lingugem visual utilizada para modelar sistemas orientados a objetos [...]”, defini elementos gráficos que podem ser utilizados na modelagem de sistemas e a partir disto, construir diagramas que representam diversas perspectivas de um sistema. O UML pode ser utilizado para modelagem do sistema, não importa a linguagem a ser utilizada na implementação do sistema.

Para Bezerra “O desenvolvimento de um sistema de software complexo demanda que seus desenvolvedores tenham a possibilidade de examinar e estudar esse sistema a partir de diversas perspectivas [...]”, cada um dos diagramas da UML fornece uma perspectiva parcial do sistema sendo modelado consistente com as demais perspectivas, sendo assim, “[...] o sistema pode ser descrito em cinco visões [...]” (BEZERRA, 2007, p. 16).

**Visão Casos de Uso:** descreve a interação entre o sistema e os agentes externos ao sistema.

**Visão de Projeto:** enfatiza as características que dão suporte tanto estrutural quanto comportamental.

**Visão de Implementação:** gerenciamento de versões do sistema construídas pelo agrupamento de módulos.

**Visão de Implantação:** corresponde a distribuição física do sistema em subsistemas e a conexão entre as partes.

**Visão de Processos:** enfatiza a visão de concorrência, sincronização e desempenho do sistema.

Bezerra (2007, p.17) afirma que dependendo das características e da complexidade do istema, nem todas as visões precisam ser construídas [...]”.

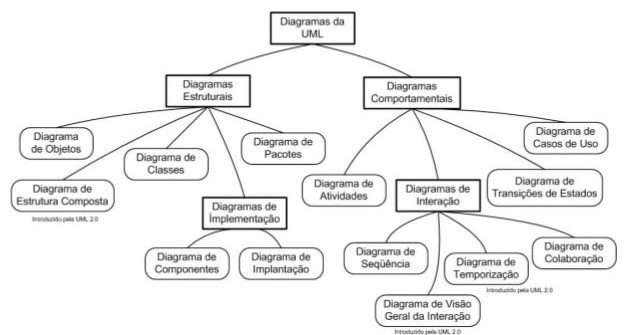
## Diagramas da UML

Segndo Bezerra (2007, p. 17): “um processo de desenvolvimento que utilize UML como linguagem de suporte à modelagem envolve a criação de diversos documentos [...]”, os documentos por sua vez podem ser textuais ou gráficos, esses documentos são denominados artefatos e compõe a visão do sistema.

Os artefatos gráficos podem ser definidos pela utilização dos diagramas de UML, ao todo temos 13 diagramas. Bezerra (2007, p. 18) afirma: “[...] os diagramas são desenhos bidimensionais, e “[...] para compensar essa dimensão a menos, utilizam-se diversos diagramas para construir modelos de várias perspectivas do sistema. ”

Os diferentes diagramas fornecem uma perspectiva parcial do sistema e deve ser consistente com as demais perspectivas.

**Figura 1 – Diagramas definidos pela UML**



Fonte: Bezerra (2007).

Para Bezerra (2007, p.17) “[...] os retângulos com os cantos retos representam agrupamentos (tipos) de diagramas da UML e os retângulos com os cantos boleados representam os diagramas propriamente ditos. ”

## Atividades típicas de um processo de desenvolvimento

Durante a construção de um sistema, as tarefas realizadas são classificadas em tarefas, segundo Bezerra (2007), “não existe o mrlhor processo de desenvolvimento, aquele que melhor se aplica a todas as situações de desenvolvimento.

### **Requisitos**

Summerville (2004, p.26) “[...] a atividade de definição dos requisitos se destina a levantar os requisitos do sistema como um todo, envolve consultas com os clientes e usuários finais do sistema [...]”.

Para Filho (2005, p. 87) “[...] os requisitos devem ser levantados pela equipe do projeto em conjunto com representantes do cliente, usuários chaves e outros especialistas da área de aplicação [...]”.

A Engenharia de Software é o conjunto de técnicas empregadas para levantar, detalhar, documentar e validar os requisitos de um produto.

O levantamento de requisitos para novos produtos é mais complexo, uma vez que é mais difícil para os usuários identificarem quais as características de maior valor, diferentemente do desenvolvimento de uma nova versão de um produto existente, pois a experiencia dos usuários com as versões anteriores, permite identificar de forma rápida e clara as necessidades prioritárias.

Para Summerville, a fase de definição de requisitos normalmente se concentra em derivar três tipos de requisitos, sendo:

1. Requisitos funcionais abstratos: são as funções básicas que o sistema deve oferecer, são definidas em um nível abstrato.
2. Propriedades do Sistema: são propriedades emergentes de sistemas não funcionais que afetam os requisitos de todos os subsistemas, podem incluir propriedades como disponibilidade, desempenho, segurança.
3. Características que o sistema não deve exibir: é tão importante quanto detalhar o que o sistema deve fazer é especificar o que ele não deve fazer.

Devem estabelecer um conjunto de objetivos gerais que devem ser cumpridos, não devendo ser apenas expresso em termos de funcionalidade, mais também o motivo pelo qual o sistema está sendo obtido.

Os requisitos são formados pelas descrições das funções e das restrições e alguns dos possíveis problemas que surgem durante o processo, são resultado da falta de separação dos níveis de descrição.

Os requisitos podem ser classificados da seguinte maneira:

Requisitos Funcionais: são as funções que o sistema deve fornecer, como reagir a determinadas entradas e situações e pode também informar o que o sistema não deve fazer.

Requisitos não funcionais: são as restrições sobre determinados serviços oferecidos pelo sistema.

### **Requisitos Funcionais**

Requisitos Funcionais descrevem a funcionalidade que se espera do sistema forneça, sendo descrito de maneira detalhada as entradas, saídas e exceções, possibilitando a definição dos recursos específicos que devem ser apresentados pelo sistema.

A imprecisão na especificação dos requisitos tende a gerar muitos problemas, através da interpretação ambígua dos requisitos, o qual se apresenta diferente do que o cliente almeja, sendo necessário estabelecer novos requisitos e promover mudanças no desenvolvimento do sistema, causando atraso na entrega e aumento nos custos.

Segundo Summerville (2004, p. 84) “[...] a especificação de requisitos funcionais de um sistema deve ser completa e consistentes. As funções devem requeridas devem ser definidas e não devem ter definições contraditórias.”

### Requisitos Não Funcionais

São aqueles que não dizem respeito diretamente as funções específicas fornecidas pelo sistema podem estar relacionadas a confiabilidade, segurança. Os requisitos não funcionais podem definir as restrições do sistema, o que deve e não deve fazer, uma vez que diz respeito como um todo e não a características individuais. Surgem da necessidade do usuário em razão de restrições de orçamento, hardware ou a fatores externos, segurança, legislação.

Summerville classifica os requisitos não funcionais em 3 tipos:

Requisitos de produtos: especificam o comportamento do processo. Desempenho, espaço de memória que irá utilizar, confiabilidade.

Requisito Organizacionais: Relacionados a políticas e procedimentos internos das empresas, como requisitos, linguagem escolhida, padrões e normas a serem seguidas.

Requisitos externos: Abrange todos os requisitos procedentes de fatores externos aos sistemas e ao seu desenvolvimento.

Os requisitos não funcionais são de difícil verificação, deixando o enfoque em

aberto e com margem para interpretações quando o sistema é entregue, não podendo ser objetivamente verificados.

## Levantamento e analise dos requisitos

O levantamento dos requisitos envolve diferentes profissionais de uma empresa, que tenham algum tipo de influência sobre o levantamento dos requisitos.

Para Summerville (2004, p. 105) o levantamento e analise compõem um processo difícil, uma vez que os Stakeholders não sabem o que querem do sistema, fazem pedidos não realistas e não possuem noção do custo das solicitações.

Segundo Bezerra (2007, p. 22): “[...] a etapa de levantamento de requisitos corresponde a etapa de compreensão do problema aplicada ao desenvolvimento do software. [...]”

Os engenheiros de software têm a difícil missão de compreender os requisitos e descobrir todas as fontes, os pontos comuns e os de conflitos que devem atender, segundo Bezerra (2007), nesta estapa, os desenvolvedores juntamente com os clientes tentam levantar e definir as necessidades dis futuros usuários.

Os requisitos são influenciados também por fatores políticos, os stakeholders definem os requisitos para poderem exercer maior influência sobre a empresa ou por influência da economia, que se modifica durante o processo de análise.

O levantamento e a análise dos requisitos é um processo continuo e interativo entre os engenheiros e a empresa, uma vez que o feedback deve acontecer de uma atividade para a outra.

## Validação dos requisitos

A validação dos requisitos se preocupa em verificar se os requisitos levantados são de fato o que o cliente deseja.

Os requisitos trabalham com os dados incompletos e a validação se ocupa em elaborar um documento completo, afim de evitar o retrabalho, que leva ao aumento dos gastos inicialmente previstos.

O custo de reparação de um erro de levantamento de requisitos é maior do que a reparação no início, considerando que as mudanças dos requisitos, é a mudança do projeto até a implementação.

## Revisão dos requisitos

A revisão dos requisitos é a verificação dos documentos de requisitos e tem como objetivo, detectar omissões ou excessos. Essas revisões podem ser formais ou informais e podem contar com muitos participantes envolvidos em diferentes partes do documento.

As revisões informais envolvem os desenvolvedores e os stakeholders, que discutem os requisitos levantados e as revisões formais, os desenvolvedores conduzem os clientes pelos requisitos levantados, com clareza, deve explicar o funcionamento de cada um dos requisitos.

“Conflitos, contradições, erros e omissões nos requisitos devem ser destacados durante a revisão e formalmente registrados”. (SUMMERVILLE 2004, pg. 117).

## Participantes do processo

O desenvolvimento de software é uma tarefa que envolve a cooperação de toda a equipe, envolvendo vários especialistas, sendo gerente, analistas, projetistas, programadores, clientes, e grupos que fazem a avaliação da qualidade.

### Gerente de Projetos

Para Bezerra (2007, p. 31): “[...] é o responsável pela coordenação das atividades necessárias para a construção do sistema[...]”, inclui fazer os orçamentos, tempo necessário para o desenvolvimento, definição do processo a ser utilizado, os recursos de hardware e software necessários.

Deve acompanhar as atividades para saber se os recursos estão sendo alocados conforme determinado e caso contrário, sanar as irregularidades.

### Analistas

É responsável por entender as necessidades do cliente, tem por objetivo entender os problemas do domínio do negócio e definir os requisitos do sistema a ser desenvolvido.

Para Bezerra (2007, p. 31): “[...] analistas devem estar aptos a se comunicar com especialistas do domínio para obter conhecimento acerca dos problemas e das necessidades envolvidas [...].”

Por ser uma ponte entre o profissional da computação e o profissional do negócio, o mesmo deve ter conhecimento suficiente da área na qual o sistema será implantado, para poder se comunicar de maneira clara com o profissional do negócio.

### Projetista

Tem como função avaliar as alternativas do problema e gerar a especificação de uma solução computacional. Existem os projetistas de interface, de redes, de bancos de dados, etc...

O ponto comum a todos os tipos é que trabalham nos modelos resultantes da análise para adicionar aspectos tecnológicos a tais modelos (Bezerra, 2007).

### Arqueiteto de software

O objetivo desse profissional é elaborar a arquitetura do sistema como um todo, através da tomada de decisões sobre quais serão os subsistemas que irão compor o sistema como um todo e quais serão as interfaces entre esses subsistemas.

Segundo Bezerra (2007, p. 33): “um profissional encontrato em grandes equipes reunidas para desenvolver sistemas complexos [..]. “

### Programadores

Esse profissional é o responsável pela implantaçõ do sistema. (Bezerra, 2007)

São os responsáveis pela implementação do sistema, o programador pode ser proficiente em uma ou mais linguagens de programação, além do conhecimento sobre o banco de dados, este participa unicamente das fases.

Segundo Bezerra (2007, p. 33): “[..] a maioria das equipes de desenvolvimento possui analistas que realizam programação e programadores que realizam alguma análise. “

## Modelo de Processo

Segundo Summerville (2004, pg. 37) “[...] um modelo de processo é uma representação abstrata de um processo de software, [...]” proporciona apenas informações parciais sobre o processo.

Os modelos de processo são utilizados para visualizar a estrutura do desenvolvimento, porém não vemos os detalhes dessa construção.

As descrições não são definições de processos de software, trata-se de abstrações úteis que podem ser utilizadas para explicar diferentes abordagens do desenvolvimento do software, diferentes processos são utilizados para desenvolver diferentes partes do sistema. (SUMMERVILLE 2004, p. 37).

### Modelo Cascata

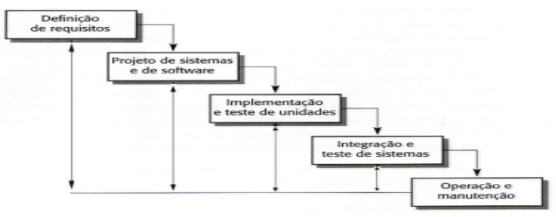
O modelo cascata foi o primeiro modelo publicado, o nome é atribuído a sequencia em cascata de uma fase para outra. Para Summerville (2004, p.37) os principais estágios do modelo retratam as atividades de desenvolvimento fundamentais:

1. Análise e definição de requisitos: as funções, restrições e os objetivos do sistema são estabelecidos por meio de consultas ao usuário.
2. Projeto de sistema de software: Envolve a identificação e a descrição das abstrações fundamentais do sistema e suas relações.
3. Implementação e testes de unidades: verifica se cada unidade atende ao especificado.
4. Integração e testes de sistemas: as unidades individuais são integradas ao sistema e testado como um sistema completo, afim de verificar se os requisitos foram atendidos.
5. Operação e manutenção: envolve a correção de erros não detectados em estágios anteriores, melhora a implementação e aumenta as funções a medida que novos requisitos são descobertos.

Os resultados de cada nova fase são documentados, passam por aprovação, que deve ser assinada, nesse modelo, os estágios que se sobrepõe e trocam informações ente si, sendo assim, enquanto não houver a conclusão de uma fase, uma nova fase não deve ser iniciada.

Esse modelo tem como característica a inflexível divisão do projeto nesses estágios distintos, sendo assim, é indicado o uso quando todos os requisitos forem bem compreendidos, porém é um modelo de simples gerenciamento, a separação do projeto e implementação levam a um sistema robusto e de suscetível a mudanças.

**Fluxograma 01 – Modelo Cascata**

  
Fonte: Elaborado pelo(a) autor(a) [[1]](#footnote-1)\*

## Desenvolvimento Revolucionario

A ideia desse modelo é a desenvolver um projeto inicial, expor o resultado a análise do usuário e a partir disso, fazer o ajuste por meio de versões, até que o sistema tenha sido totalmente desenvolvido com base nas reais necessidades do cliente.

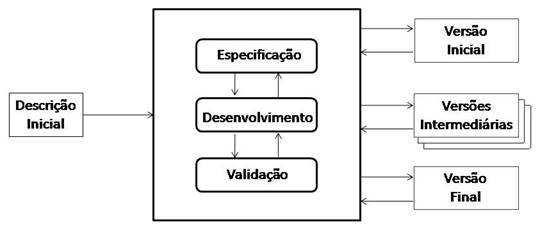
Para Summerville, existem dois tipos de desenvolvimento evolucionário:

- Exploratório: tem como objetivo trabalhar com o cliente visando o levantamento de requisitos. Inicialmente o sistema é desenvolvido com as compreensões das necessidades e após, evolui de acordo com as solicitações do cliente.

- Protótipos descartáveis: tem a finalidade de fazer experimentos com partes dos requisitos que estejam mal compreendidos.

Segundo Summerville (2004, p. 39): “[...] é mais eficaz do que a abordagem em cascata, no sentido de produzir sistemas que atendam às necessidades imediatas dos clientes, [...]” a especificação pode ser desenvolvida gradativamente.

**Fluxograma 02 - Desenvolvimento Revolucionario**



Fonte: Elaborado pelo(a) autor(a) [[2]](#footnote-2)\*

## Desenvolvimento Incremental

O modelo incremental é uma combinação das vantagens do modelo cascata e evolucionário, tem como objetivo reduzir o retrabalho durante o desenvolvimento e permite ao cliente adiar decisões sobre os requisitos detalhados até o experimento de parte do sistema.

Os clientes fazem um esboço dos requisitos do sistema, identificam quais as funções mais importantes e quais são menos importantes e a partir disso, é definido os estágios de entrega e a cada entrega, é fornecido ao cliente, uma funcionalidade do sistema, as prioridades são definidas pelo cliente.

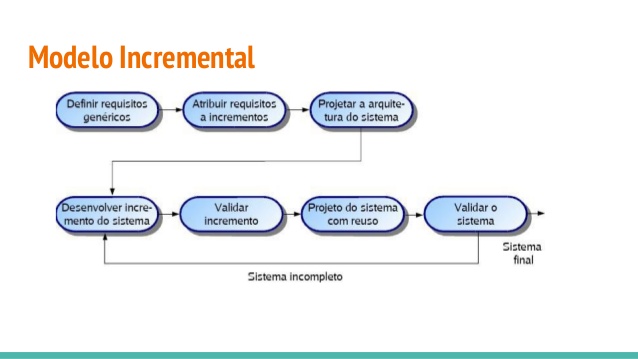
Após a identificação dos requisitos mais importantes, os mesmos são detalhados e o desenvolvimento é feito utilizando o modelo mais apropriado, durante o desenvolvimento pode ocorrer alterações nos requisitos, porém estes não são aceitos neste momento.

Considerando que neste modelo, há o incremento por partes e que a cada nova fase concluída, a mesma é entregue ao cliente para que a utilize e assim, consiga esclarecer alguns requisitos para as próximas versões e a cada estágio concluído. O incremento deve ser pequeno e com alguma funcionalidade.

Conforme afirma Summerville (2004, p. 44): “[...] não existe a necessidade de utilizar o mesmo processo para o desenvolvimento de cada incremento”. Quando os requisitos precisam ser bem definidos, pode se utilizar o modelo cascata, porém se os requisitos não estiverem bem definidos, pode se utilizar o modelo evolucionário.

Entre as diversas vantagens de usar o modelo incremental, vamos citar as principais conforme apontado por Summerville: o cliente não precisa esperar a conclusão de todo o sistema; ao fazer o uso dos primeiros incrementos, auxilia na tomada de decisões dos próximos incrementos, menor o risco de fracasso total do sistema, pois é feito ao pouco, com a participação do cliente.

**Fluxograma 03 - Desenvolvimento Incremental**



Fonte: Elaborado pelo(a) autor(a) [[3]](#footnote-3)\*

## Modelo Espiral

O modelo espiral tem como característica uma sequencia de atividades com o retorno de algumas atividades para a outra, o processo é representado como um espiral, cada loop representa uma fase (Summerville, 2004).

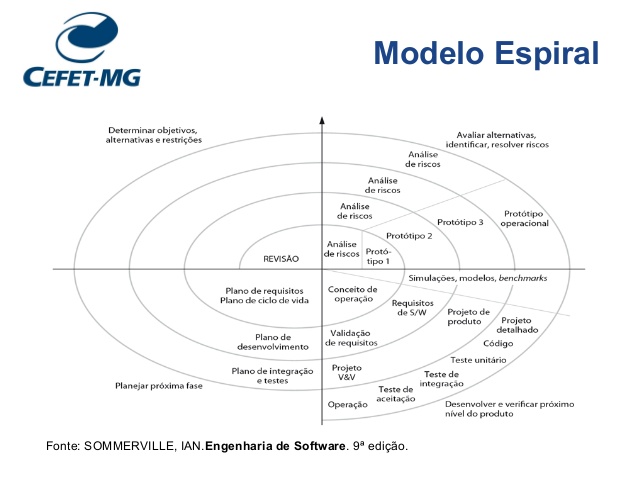
Segundo Summerville (2004, p.45) cada loop da espiral é dividido em quatro setores.

1. Definição de objetivos: é a identificação dos requisitos e das restrições e a partir disso, é desenvolvimento o plano de gerenciamento.
2. Avaliação e redução dos riscos: para cada risco identificado, e após, são tomadas providencias para a redução dos riscos.
3. Desenvolvimento e validação: após a identificação dos requisitos e avaliação dos riscos, é escolhido um modelo de desenvolvimento.
4. Planejamento: é feita uma revisão e assim é tomada a decisão sobre a continuidade do próximo loop do espiral.

O modelo espiral se destaca por ser explicita a consideração dos riscos existentes, uma vez que o risco resulta em problemas no projeto, com aumento de prazo e custos, por esse motivo, é tão importante gerenciar os riscos.

O ciclo inicia com a elaboração dos objetivos, as restrições e a identificação dos riscos e a partir disso, alguma parte do projeto é desenvolvida, seguida pelo planejamento da próxima fase, não há fases fixas, o modelo abrange também outros modelos de processos.

**Figura 01 - Modelo Espiral**



Fonte: Elaborado pelo(a) autor(a) [[4]](#footnote-4)\*

# Modelagem de casos de uso

Segundo Bezerra (2007, p. 53): “O modelo de caso de uso é uma representação das funcionalidades externamente observáveis do sistema e dos elementos externos ao sitema que interagem com ele. ”

Diagrama de Caso de Uso descreve a funcionalidade proposta para um novo sistema que será projetado, é uma excelente ferramenta para levantamento dos requisitos funcionais do sistema, visto que representa um refinamento dos requisitos funcionais.

Bezerra (2007) observa que o caso de uso direciona diversas tarefas posteriores do processo de desenvolvimento do software e desta maneira, “[...] força os desenvolvedores a moldarem o sistema de acordo com as necessidades do usuário.

## Casos de uso

Seundo Bezerra (200 p.): “[...] a especificação de uma sequência completa de interações entre um sistema e um ou mais agentes externos a esse sistema.

O caso de uso é um relato de uso de certa funcionalidade do sistem, sem revelar a estrutura e o comportamento internos desse sistema. Tem por característica a perspectiva externa, ou seja, um observador sabe quais são as funcionalidades fornecidas pelo sistema e quais são os resultados produzidos, porém não sabe como esse sistema age internamente para produzir os resultados externamente visíveis.

O caso de uso não é um passo em uma funcionalidade do sistema e sim um relato fim a fim de um dos usos do sistema por um agente externo.

Para Bezerra (2007, p. 54): “[...] a quantidade exata de casos de uso obviamente depende da complexidade do sistema em desenvolvimento: quanto mais complexo o sistema, maior a quantidade de casos de uso.

O caso de uso de um sistema é definido pela descrição narrativa das interações que ocorrem entre o elemementos externos e o sistema.

As descrições que você fornecer para casos de uso, usará vários termos relacionados ao domínio em que o sistema funciona.

Esse diagrama documenta o que o sistema faz do ponto de vista do usuário, ele descreve as principais funcionalidades do sistema e a interação dessas funcionalidades com os usuários do mesmo sistema. Com a ajuda de um diagrama de caso de uso, você pode discutir e comunicar-se:

-Os cenários em que o sistema ou o aplicativo interage com as pessoas.

- A metas que ajudam os atores a atingir

-O escopo de seu sistema.

### Cenários

Bezerra (2007) afirma que um cenário é a descrição de uma das maneiras pelas quais um caso de uso pode ser utilizado, “[...] é como a descrição de um episódio de utilização de alguma funcionalidade do sistema. “

O cenário também pode ser chamado de instância e normalmente há diversos cenários para um mesmo caso de uso.

Segundo Bezerra (2007, p. 59): “[...] denominamos realização de um caso de uso ao conjunto de diagramas e artefatos textuais que especificam como este caso de uso é executado internamente ao sistema em desenvolvimento. ”

### Atores

Ator é qualquer elemento externo ao sistema que interage com o mesmo, sendo assim, indica que os atores não fazem parte do sistema, porém faz a troca informações com o sistema.

O ator envia informações para o sistema processar, ou recebe informações processadas provenientes do sistema, representam a forma pela qual um sistema percebe o seu ambiente.

Para Bezerra (2007) um ator pode participar de muitos casos de uso e um caso de uso pode envolver a participação de vários atores, resultando na classificação dos atores em primários ou secundários.

O ator primário é aquele que inicia uma sequência de interações de um caso de uso, recebe benefício direto e as funcionalidades principais do sistema são definidas de acordo com os objetivos dos atores primários e o ator secundário supervisiona, opera, mantém ou auxilia na utilização do sistema pelos atores primários.

### Relacionamentos

Segundo Bezerra (2007, p. 61): “[...] o modelo de casos de uso possui um terceiro componente, cuja função é relacionar os atores e casos de uso. Esse componente corresponde aos relacionamentos. ”

O ator deve estar relacionado a um ou mais casos de uso do sistema e, pode haver relacionamentos entre os casos de uso ou entre os atores de um sistema.

Os relacionamentos podem ser: comunicação, inclusão, extensão e generalização.

* **Relacionamento de comunicação:** informa a que caso e uso o ator está associado., significa que esse ator troca informações com o sistema por meio através do caso de uso.
* **Relacionamento de inclusão:** é quando dois ou mais casos de uso incluem uma seqüência comum e essa seqüência pode ser descrita em outro caso de uso e vários casos de uso do sistema podem incluir o comportamento desse caso de uso comum.
* **Relacionamento de extensão:** é utilizado para modelar situações em que diferentes seqüências de interações podem ser inseridas em um mesmo caso de uso, representa um comportamento eventual, ou seja, que só ocorre sob certas condições.
* **Relacionamento de Generalização:** existe entre dois casos de uso ou entre dois atores, permite que um caso de uso herde características de um caso de uso mais genérico. O caso de uso herdeiro pode especializar o comportamento do caso de uso base.

## Diagrama de Caso de Uso

Segundo Bezerra (2007, p. 70): “[...] esse diagrama representa graficamente os atores, casos de uso e relacionamentos entre esses elementos. “ Tem como objetivo de ilustrar quais elementos externos interagem com que funcionalidades do sistema.

Inicialmente deve-se identificar todos os atores do sistema, para isto, o analista deve identificar quais as fontes de informações a serem processadas e quais são os destinos das informações geradas pelo sistema. Após a identificação dos atores, deve-se passar à identificação dos casos de uso, que podem ser primários ou secundários.

Os casos de uso primários representam os objetivos dos atores, são os processos que estão sendo automatizados pelo sistema e e os casos de uso secundários são aqueles que não traz benefício direto porém são necessários para que o sistema funcione adequadamente.

## Diagrama de Classes

Conforme Bezerra (2007, p. 112): “[...] o diagrama de classes é utilizado na construção do modelo de classes desde o nível de análise até o nível de especificação. ”

As classes são representadas por uma caixa com três comportimentos, o primeiro exibe o nome, o segundo declara os atributos e o terceiro declara as ações que o objeto realiza, sendo assm, é composta de atributos e operações.

Os atributos correspondem à descrição dos dados armazenados pelos objetos de uma classe e as operações correspondem à descrição das ações que os objetos de uma classe sabem realizar.

As classes podem se relacionar umas com as outras e esse relacionamento é representado por uma associação, que é representada no diagrama por uma linha ligando as classes às quais pertencem os objetos relacionados.

A quantidade de objetos aos quais outro objeto pode estar associado é representado pela a informação dos limites inferior e superior das associações que é chamado de multiplicidades.

**Tabela 1: Simbologia para representar multiplicidades**

|  |  |
| --- | --- |
| **Nome** | **Simbologia** |
| Apenas um | 1 |
| Zero ou Muitos | 0...\* |
| Um ou Muitos | 1...\* |
| Zero ou Um | 0...1 |
| Intervalo Específico | 1..1 |

Fonte: Bezerra, Príncipio de Análise e Projeto de Sistetmas com UML 2007.

## Diagrama de Sequencia

Segundo Bezerra (2007, p. 193): “[...] o objetivo do diagrama de seqüência é apresentar as interações entre objetos na ordem temporal em que elas acontecem.”

Na elaboração do diagrama, é possível utilizar algumas notações particulares que podem ser: linhas de vida, envio de mensagens, ocorrências de execução, criação e destruição de objetos.

* **Linha da vida:** é a notação gráfica utilizada para representar os objetos que participam da realização de de um caso de uso e após a sua identificação e esses devem ser posicionados no diagrama de seqüência correspondente;
* **Mensagens:** é uma flecha que liga uma linha de vida a outra. O objeto do qual parte a seta é aquele que está enviando a mensagem e o objeto para o qual a seta aponta é aquele que está recebendo a mensagem;
* **Ocorrências de execução:** representa o tempo em que o objeto está ativo, ou seja, o tempo em que ele realiza alguma operação;
* **Criação e destruição de objetos:** criação de um objeto é o momento em que esse objeto passa a existir no sistema, e passa a realizar suas responsabilidades e a destruição de um objeto é o momento no qual este objeto deixa de existir no sistema.

## Diagrama de Atividade

Segundo Bezerra (2007, p. 307): “um diagrama de atividade é um tipo especial de diagrama de estados, em que são representados os estados de uma atividade, em vez dos estados de um objeto. “

Os diagramas de atividade são orientados a fluxos de controle, representam ações paralelas juntamente com a sua sincronização, os elementos de um diagrama de atividade podem ser divididos em dois grupos: fluxos de controle seqüenciais e os fluxos de controle paralelos.

### Fluxos de controle sequenciais

Estado ação, Estado atividade, Estados inicial e final, e condição de guarda, Transição de término e Pontos de ramificação e de união. O diagrama de atividade deve ter um estado inicial; e pode ter também vários estados finais e guardas associados a transições. Essa transição significa o término de um passo e o consequente início do outro e quando o fluxo de controle chega a um ponto de ramificação, uma e somente uma das condições de guarda deve ser verdadeira.

### Fluxos de controle paralelos

Para Bezerra (2007) significa que pode haver dois ou mais fluxos de controle sendo executados simultaneamente em um diagrama de atividades.

Para sincronizar dois ou mais fluxos paralelos, as barras de sincronização são utilizadas: barra de bifurcação (fork) e barra de junção (join).

* **Barra de bifurcação:** recebe uma transição de entrada e cria dois ou mais fluxos de controle paralelos e a partir desse momento, cada um dos fluxos criados é executado independentemente e em paralelo com os demais.
* **Barra de junção**: recebe duas ou mais transições de entrada e une os fluxos de controle em um único fluxo, tem o objetivo de sincronizar fluxos paralelos.

# TOPOLOGIA DE REDES

As redes de computadores possibilitam que as pessoas consigam trabalhar em equipes, compartilhando informações, dados, melhorando o desempenho da realização de suas tarefas, e estão presentes no dia-a-dia de todos nós. São estruturas sofisticadas e complexas, que mantém os dados e as informações ao alcance de seus usuários.

A topologia de redes que descreve como elas está interligada, tanto do ponto de vista físico, como o lógico. A topologia física representa como as redes estão conectadas (layout físico) e o meio de conexão dos dispositivos de redes (nós ou nodos). Já a topologia lógica refere-se à forma com que os nós se comunicam através dos meios de transmissão. (INFOESCOLA, 2017).

Em redes de comunicação, um nodo ou nó é um ponto de conexão, seja um ponto de redistribuição ou um terminal de comunicação. A definição de um nó depende da rede e da camada de protocolo referida. Um nó de rede física é um dispositivo eletrônico [ativo](https://pt.wikipedia.org/w/index.php?title=Passividade_(engenharia)&action=edit&redlink=1) que está ligado a uma rede, e é capaz de enviar, receber ou transmitir informações através de um canal de comunicação. Um ponto de distribuição [passivo](https://pt.wikipedia.org/w/index.php?title=Passividade_(engenharia)&action=edit&redlink=1), como um [quadro de distribuição](https://pt.wikipedia.org/wiki/Quadro_de_distribui%C3%A7%C3%A3o) ou [*patch panel*](https://pt.wikipedia.org/w/index.php?title=Patch_panel&action=edit&redlink=1), não é, portanto, um nó. (FERNANDEZ, 2015).

## Tipos de conexão

### Ponto a Ponto

FOROUZAN, B (2008) destaca que uma conexão ponto a ponto gera um *link* dedicado entre dois dispositivos e toda a capacidade do *link* é reservada para a transmissão entre os dois dispositivos. A maioria das conexões ponto a ponto usa um pedaço real de fio ou de cabo para conectar as duas extremidades, um exemplo de conexão ponto a ponto é quando mudamos um canal de televisão por controle remoto infravermelho.

### Multiponto

Uma conexão multiponto (multidrop) é uma conexão na qual mais de dois dispositivos compartilham um único *link*. Em um ambiente multiponto, a capacidade do canal é compartilhada, seja de forma espacial, ou seja, de forma temporal. (BEHROUZ A. FOROUZAN, 2008, p.8).

## Tipos de Topologia

### Topologia de Barramento

Esta é a topologia física utilizada pelas redes Ethernet, que utilizam cabos coaxiais. Neste tipo de rede um PC é ligado ao outro, usando vários segmentos de cabos e conectores, onde uma ponta é ligada na placa de rede e as outras duas são ligadas às estações vizinhas. Nas duas extremidades da rede temos terminadores, que absorvem os sinais, evitando que eles retornem na forma de interferência.

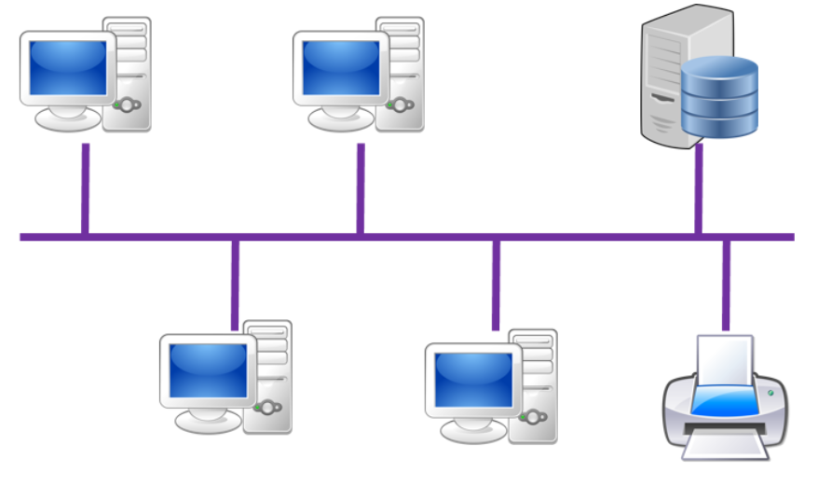
Vantagens:

* Uso de cabo é econômico;
* Barata e fácil de trabalhar e instalar;
* Simples e relativamente confiável;
* Fácil expansão.

Desvantagens:

* Rede pode ficar extremamente lenta em situações de tráfego pesado;
* Problemas são difíceis de isolar;
* Falha no cabo paralisa a rede inteira.

**Figura 02 - Topologia de Barramento**



Fonte: Bóson Treinamentos em Tecnologia [[5]](#footnote-5)\*

### Topologia Estrela

Como o nome indica esta topologia tem a forma de uma estrela, ela consiste em vários cabos que unem cada dispositivo a um ponto central. As redes Ethernet a 10 Mbps  são baseadas numa estrutura em estrela, e onde cada dispositivo da rede está ligado a um HUB  por cabos “RJ45”. Os dispositivos não são ligados diretamente entre si, ela não permite tráfego direto entre os dispositivos, a central trabalha como se fosse uma central telefônica: se um dispositivo quiser enviar dados para outro dispositivo, ele deve enviar os dados ao controlador central que, então, os retransmite ao outro dispositivo conectado.

Vantagens:

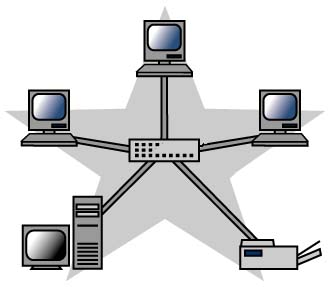
* A codificação e adição de novos computadores são simples;
* Gerenciamento centralizado;
* Falha de um *link* não afeta o restante da rede;
* Simplicidade no protocolo de comunicações;

Resume-se a selecionar qual o nó periférico que em cada momento está ligado ao nó central.

Desvantagens:

* Uma falha no dispositivo central paralisa a rede inteira;
* Maior comprimento de cabo para efetuar ligações à distância máximas sem amplificação é de apenas 100 m.

**Figura 03 - Topologia Estrela**



Fonte: Bóson Treinamentos em Tecnologia [[6]](#footnote-6)\*

### Topologia Anel

Na topologia em anel os pacotes circulam por todos os dispositivos da rede, tendo cada um o seu endereço. O fluxo de informação é unidirecional, existindo um dispositivo “HUB” que intercepta e gera o fluxo de dados que entra e sai do anel, de dispositivo para dispositivo, até atingir seu destino.

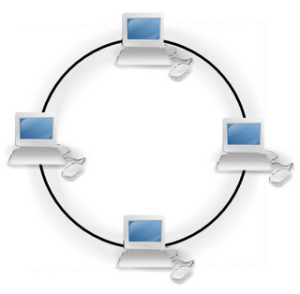
Vantagens:

* Todos os computadores acessam a rede igualmente;
* Desempenho não é impactado com o aumento de usuários;
* Fácil de ser instalado e reconfigurado.

Desvantagens:

* Falha de um computador pode afetar o restante da rede;
* Problemas são difíceis de isolar;
* Ampliação da rede, inclusão de novas estações ou servidores implica na paralisação da rede.

**Figura 04 - Topologia Anel**

[](https://jbgsm.files.wordpress.com/2010/01/anel.png)

Fonte: Bóson Treinamentos em Tecnologia [[7]](#footnote-7)\*

### Topologia Hibrida ou Mista

Este tipo de topologia é uma topologia cujo qual combina duas ou mais outras topologias existentes. Construindo uma rede em cima dessa topologia podendo se beneficiar das vantagens de cada topologia. Por exemplo, podemos ter uma topologia principal de anel, e cada ramificação conectando várias estações em uma topologia de barramento.

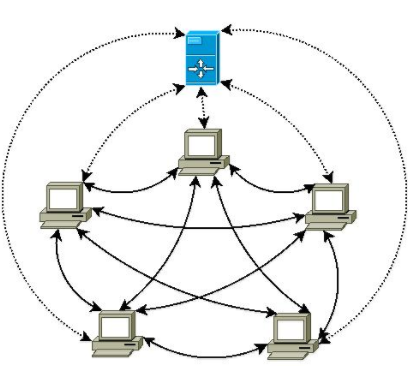
Vantagens:

* Tem a possibilidade de ligar varias redes;
* Tem como característica as ligações ponto a ponto e multiponto;
* Tem melhor tolerância a falhas.

Desvantagens:

* A ligação não é uniforme;
* Custo de cabos caros;
* Instalação e configuração difícil.

**Figura 05 - Topologia Hibrida ou Mista**



Fonte: Bóson Treinamentos em Tecnologia [[8]](#footnote-8)\*

### Topologia em Malha

Nessa topologia, cada dispositivo possui um *link* ponto a ponto dedicado com cada um dos demais dispositivos. O termo *dedicado* significa que o *link* transporta tráfego apenas entre os dois dispositivos que ele conecta. Uma topologia em malha oferece várias vantagens em relação as demais topologias de rede, pois, o uso de links dedicados garante que cada conexão seja capaz de transportar seu próprio volume de dados, eliminando, portanto, os problemas de tráfego que possam ocorrer quando os links tiverem de ser compartilhados por vários dispositivos.

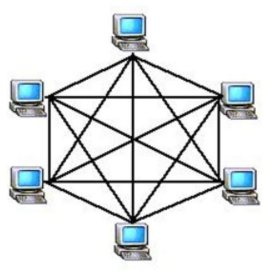
Vantagens:

* Maior redundância e confiabilidade;
* Facilidade de diagnóstico;
* Não há necessidade de decisões de por onde enviar a mensagem (roteamento).

Desvantagens:

* Instalação com custo muito alto;
* Grande quantidade de ligações;
* Dependendo de números de computadores conectados diminui a velocidade de tráfego.

**Figura 06 - Topologia em Malha**



Fonte: Bóson Treinamentos em Tecnologia [[9]](#footnote-9)\*

### Topologia Ponto-a-Multiponto

Esta topologia lembra, fisicamente, uma topologia estrela, porém sem que necessariamente todos os nós da rede possam se comunicar completamente uns com os outros.

Um exemplo típico dessa topologia são as redes 802.11 (sem fio), na qual podemos ter diversos pontos de rede se conectando a um Access Point (AP / Ponto de Acesso), o qual gerencia a conectividade entre os pontos de rede e pode impedir que as estações e comuniquem entre si, somente trocando dados com o próprio AP.

**Figura 07 - Topologia Ponto-a-Multiponto**



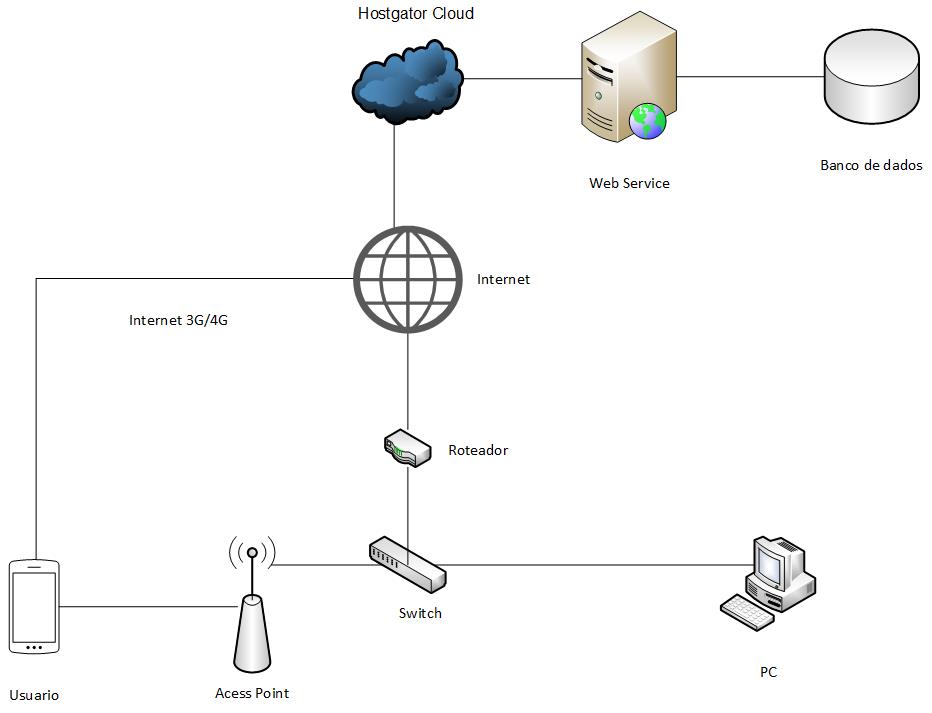
Fonte: Bóson Treinamentos em Tecnologia [[10]](#footnote-10)\*

### Topologia do PIM

Esta topologia lembra, fisicamente, uma topologia estrela, porém sem que necessariamente todos os nós da rede possam se comunicar completamente uns com os outros.

Um exemplo típico dessa topologia são as redes 802.11 (sem fio), na qual podemos ter diversos pontos de rede se conectando a um Access Point (AP / Ponto de Acesso), o qual gerencia a conectividade entre os pontos de rede e pode impedir que as estações e comuniquem entre si, somente trocando dados com o próprio AP.

**Figura 8 - Topologia PIM**



Fonte: Bóson Treinamentos em Tecnologia [[11]](#footnote-11)\*

# Ferramentas

## HTML5

O HTML está presente a mais de 10 anos, e tem uma grande aceitação dos usuários por todo o mundo. O HTML5 é uma linguagem de marcação de hipertexto para apresentar e estruturar o conteúdo na Web. Em português HTML significa Linguagem de marcação de Hipertexto (Um texto que apresenta diferentes caminhos de leitura).

É a quinta versão da linguagem HTML, esta nova versão traz novas funcionalidades e passou a ter mais semântica e acessibilidade. Sua principal característica é para melhorar a linguagem como suporte para as mais recentes multimídias e manter legível por seres humanos e compreendida por computadores e outros dispositivos. E também para permitir aos programadores gerar códigos mais organizados.

O HTML5 adiciona várias funções sintáticas, tais funções são projetadas para tornar mais fácil a inclusão e manipulação do conteúdo gráfico e multimídia na web. Utilizamos elementos de HTML para introduzir imagens parágrafos, títulos, subtítulos, vídeos, tabelas, listas, etc. Os elementos do HTML são chamados de Tags. Todos elementos de HTML podem ser identificados, porque sempre começam e terminam com os delimitadores “<” e “>”.

De acordo com Silvio Ferreira (Ferreira, 2013, pag 56): “Portanto programar com HTML5 exige uma nova forma de pensar. Ao criar uma página para Web, tudo o que for feito e construído nela passa a ter significado. A página irá possuir cabeçalhos, menus, artigos, conteúdo relacionado e rodapé. Em cada parte dela possui novas tags que as identificam categoricamente no código”.

## Bootstrap

Bootstrap é um framework front-end que está disponível desde 2011, foi desenvolvido por Jacob Thorton e Mark Oto, engenheiros do Twitter, para ajudar e facilitar o trabalho dos desenvolvedores web a criar sites com tecnologia mobile, tornando o desenvolvimento mais rápido e fácil, sem precisar inicialmente digitar uma linha se quer, de CSS e com inúmeras bibliotecas prontas disponíveis. Oferece padrões para HTML, JavaScript e CSS.

Possui uma interface bastante amigável e moderna, oferece uma enorme quantidade de plug-ins, além de ter o código aberto, ou seja, é totalmente free. Sua principal aplicação seria na criação de sites responsivos (Mobile) e focado para dispositivos móveis na web.

No Bootstrap é preciso apenas inicializar o script e adicionar algumas configurações no código, como no exemplo a seguir:

1 <a href="#" title="Título que aparece como tooltip" data-toggle="tooltip">Link com Tooltip</a>

2 <script>

3 $(function(){

4 $('[data-toggle="tooltip"]').tooltip();

5 })

6 < /script>

O Bootstrap traz uma série de recursos:

-Reset CSS

-Estilo Visual base para maioria das tags

-Ícones

-Grids pronto para uso

-Componentes CSS

-Plugins JavaScript

-Tudo responsivo e mobile-first

## CSS3

Com o aparecimento de sites mais complexos, surgiram as CSS como uma solução as deficiências e limitações da linguagem HTML. A formação de arquivos HTML compostos de informações e formatações misturadas, levando sempre em consideração o aspecto estético das páginas, resultaram em uma consequência negativa.

A CSS tem como função extrair a formatação de uma página do código HTML, e separa-lo do conteúdo propriamente dito, aumentar a organização e definir a formatação de uma ou várias páginas, tornando mais fácil manter um padrão de cores, fontes e estilos. Ao utilizar um arquivo CSS, é possível habilitar vários arquivos CSS diferentes, e quando desejar mudar o layout de alguma delas, basta alterar a chamada ao arquivo CSS.

Segundo Renata Hiromi Minami Miyaguku (Miyaguku, 2007, pag.95) “As CSS podem ser definidas de três formas: diretamente no código HTML, em uma seção própria para definições das formatações no hearder do documento, ou em um arquivo CSS externo. Se houver uma formatação de um determinado elemento dentro do documento HTML, e este também fizer referência a um arquivo CSS externo, por exemplo, será respeitada a formatação definida no documento HTML”.

AS CSS também são muito úteis e funcionais na formatação das cores, tanto para definição quanto para alteração. As cores dos elementos de uma página podem ser definidas de quatro formas:

Nome da cor, em que existem aproximadamente 150 nomes de cores válidos que podem ser utilizados.

Valores RGB, em que RGB (Red, Green e Blue), as cores são obtidas a partir da mistura dessas três cores.

Percentual RGB, em que é uma variação do sistema RGB que utiliza valores percentuais para balancear o nível de cada uma das três cores.

Valor Hexadecimal, cada cor tem sua representação em escala hexadecimal, composto pelo caractere “#”, seguido por uma sequência de seis caracteres alfanuméricos (0-9 e A-F).

Com a atualização da CSS3 se obteve mais controle sobre a forma. Assim a novidade mais importante para os desenvolvedores de web, está na incorporação de novos mecanismos para obter maior controle sobre o estilo com o qual se mostram os elementos das páginas.

Antigamente, isso era feito no próprio HTML. Se quisesse um título em vermelho, era só fazer:

<h1><font color="red">Mirror Fashion anos 90</font></h1>

Além da tag font, várias outras tags de estilo existiam. Mas isso é passado. Tags HTML para estilo são má prática hoje em dia e jamais devem ser usadas. Em seu lugar, surgiu o CSS, que é uma outra linguagem, separada do HTML, com objetivo único de cuidar da estilização da página.

## JavaScript

Visando a necessidade de uma maior interação do usuário com as páginas da web, foi criado o LiveScript, uma linguagem simples, que permitia a execução de scripts. Posteriormente foi adicionado ao Internet Explorer o suporte a scripts, escrito em VBScripts e foi criado sua própria versão de JavaScript.

JavaScript é uma linguagem de script do lado do cliente mais amplamente utilizada, que pode ser executada em várias plataformas. Sua combinação vem das linguagens Java e C, foi criada para que scripts pudessem ser executados do lado do cliente e interagir com o usuário, sem a necessidade deste script passar pelo servidor, e também para trabalhar com aplicações interativas nas páginas HTML, mas não é uma boa ideia utiliza-la como linguagem independente. Diferente da maioria das linguagens de programação o JavaScript não possui o conceito de entrada e saída.

Nunca devemos confundir Java com JavaScript. JavaScript é uma linguagem de script, que não tem classes, ao invés disso, a funcionalidade de classe é realizada por protótipos de objetos, enquanto que java é uma complexa linguagem de programação. Cada linguagem tem um conceito e um design diferente.

Hoje em dia, a JavaScript é a linguagem de script mais conhecida. Pode ser utilizada nos navegadores mais populares como Chrome, Fiferefox, Internet Explorer, Opera e Safari.

### Camadas de desenvolvimento

Geralmente as aplicações da Web são construídas respeitando o conceito de desenvolvimento em camadas. As três camadas do desenvolvimento são:

-Camada de estruturação dos conteúdos, que consiste na marcação HTML.

-Camada de apresentação, representada pelas folhas de estilo.

-Camada de comportamento, onde entram os scripts que definem o comportamento das aplicações.

Os códigos JavaScript funcionam sob a forma de funções e métodos, que podem estar presentes em qualquer trecho do código. Essas funções são declaradas com <script> e finalizadas com </script>.

“Design não é só como uma coisa aparenta, é como ela funciona. ” – SteveJobs.

## Microsoft Visio

O Microsoft Visio é um programa da Microsoft que permite compor diversos tipos de gráficos de organização diferentes, como fluxogramas, organogramas e diagramas em geral.

### Interface e funções do programa

O Microsoft Visio disponibiliza diversas ferramentas para auxiliar no gerenciamento de projetos e milhares de figuras, formas e desenhos diferentes, permitindo que você possa desenhar desde o planejamento de uma estrutura de rede para aplicação em uma empresa até um organograma empresarial, além de WBS, fluxograma, diagramas de modelagem para desenvolvimento, entre outros.

A utilização do programa é simples, pois sua interface é organizada da mesma forma que os produtos do Microsoft Office, o que traz certa familiaridade para encontrar funções.

### Vantagens do Visio para sua organização

O Microsoft Visio é uma excelente ferramenta para desenvolver projetos e, como a maioria dos programas da Microsoft, conta com uma ampla gama de opções. A montagem dos diagramas é inteiramente feita por “clicar e arrastar” imagens e mesmo inserções de textos ou ligações entre figuras são extremamente simples por contarem com botões para a função.

A interface do programa é muito “visual”, o que permite que o uso de suas ferramentas se torne intuitivo, mesmo para usuários iniciantes. Não traz maiores dificuldades em seu uso, pois é uma ferramenta cuja organização de menus e conteúdos traz familiaridade, especialmente para usuários do pacote Office da Microsoft.

Da mesma forma, as funcionalidades são intuitivas e mesmo que em um primeiro momento você não entenda perfeitamente como aplicá-las, a utilização de certa forma demonstra o que é preciso para completar o uso da função. (TECHSOUPBRASIL, 2017).

**Figura 09 - Logo Microsoft Visio**



Fonte: Microsoft Visio [[12]](#footnote-12)\*

## PhoneGAP

O PhoneGap é um software framework de código aberto, totalmente gratuito, que permite a criação de aplicações móveis utilizando APIs (*Application Programming Interface*ou Interface de Programação de Aplicativos) padronizadas da web. É indicado para desenvolvimento de aplicativos de pequeno e médio porte.

O software foi desenvolvido pela Nitobi em 2008, passando a dar suporte primeiramente a Iphone, Android e Blackberry 4, posteriormente a Symbian e WebOS e finalmente a Windows Phone 7. Em 2011, a Adobe adquiriu a Nitobi software. Em outubro de 2011, o PhoneGap foi doado para a Apache Software Foundation (ASF), sob o nome Apache Cordova. Através da ASF, o desenvolvimento do PhoneGap passa a garantir administração aberta do projeto. Permanecendo livre e de código aberto sob a licença Apache, Versao 2.0.

Para entender melhor, portanto, o PhoneGap é um conjunto de APIs que permite que o desenvolvedor acesse as funções nativas dos dispositivos, como câmera, agenda, etc; através de JavaScript, HTML5 e CSS3, em vez de linguagens específicas de dispositivo, como Objective-C. O desenvolvimento é como o de qualquer site e por isso oferece uma maior facilidade de aprendizado. Dispensando, por exemplo, longas horas de dedicação a sistemas mais complicados, como Java, além de reduzir custos do projeto. (GETNINJAS, 2017).

**Figura 10 - PhoneGap**



Fonte: Alpha Software [[13]](#footnote-13)\*

## Sublime Text

Um editor de texto é uma ferramenta essencial em qualquer sistema operacional, este tipo de ferramenta integra muitas funções direcionadas á programação.

O Sublime é um software multiplataforma de edição de texto que foi desenvolvido em Pyton e feito para ser simples. É um editor de texto bastante fácil de usar, porém com muitos recursos e funcionalidades. Mesmo com tantos recursos é uma ferramenta muito rápida.

## Xampp

É um servidor independente de plataforma, sua base principal é o MySQL, o qual foi substituído pelo MariaDB, também é considerado o pacote com os principais servidores de código aberto do mercado. Com ele é possível rodar sistemas como WordPress e Drupal, localmente, o que facilita e agiliza o desenvolvimento.

O Xampp está disponível para quatro sistemas operacionais:

-Windows

-Linux

-Mac OS X

-Solaris

Traz ainda em sua index (http: //127.0.0.1) um menu com a relação de seus utilitários e alguns exemplos em php e perl. O principal objetivo do Xampp é construir uma distribuição fácil de instalar para desenvolvedores entrarem no mundo do Apache.

## Pencil Wireframe

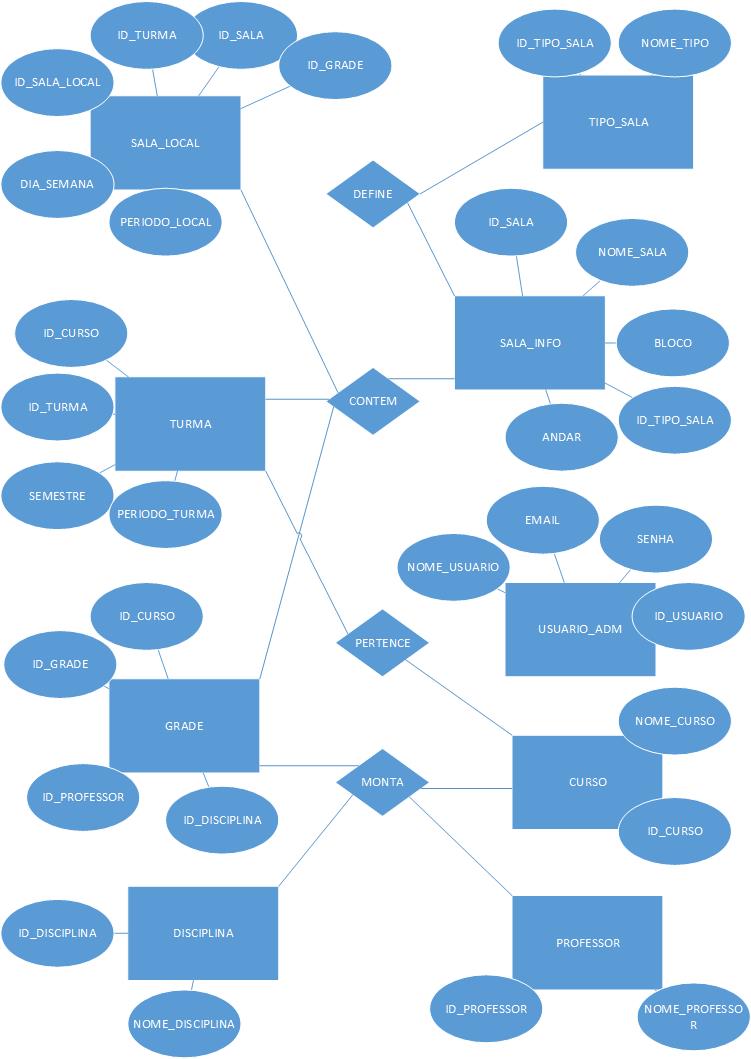
Wireframe também conhecido como uma página de modelo esquemático, é um guia visual que representa a estrutura de website.

Pencil Wireframe é uma ferramenta de código aberto e que tem o objetivo de facilitar o protótipo de interfaces de sistemas e páginas web. É usado para fazer o layout do software.

## MER

O processo começa com três tabelas, sendo elas: Curso, Disciplina e Professor, que armazenam os seguintes atributos: Nome e ID, que irá variar de acordo com as suas tabelas. Essas entidades, irão fornecer suas informações para a entidade grade, que enviará todas as informações para sala local, junto com outras duas entidades, sala info e turma, que contém os seguintes atributos: nome, bloco, andar, tipo da sala, semestre e período da turma. A entidade tipo sala fica responsável por definir o tipo da sala, se ela vai ser um laboratório, sala e etc... O usuário adm contém os atributos: nome usuário, email e senha, o qual servirá para consulta quando o login for realizado no sistema.

**Fluxograma 04 - MER**

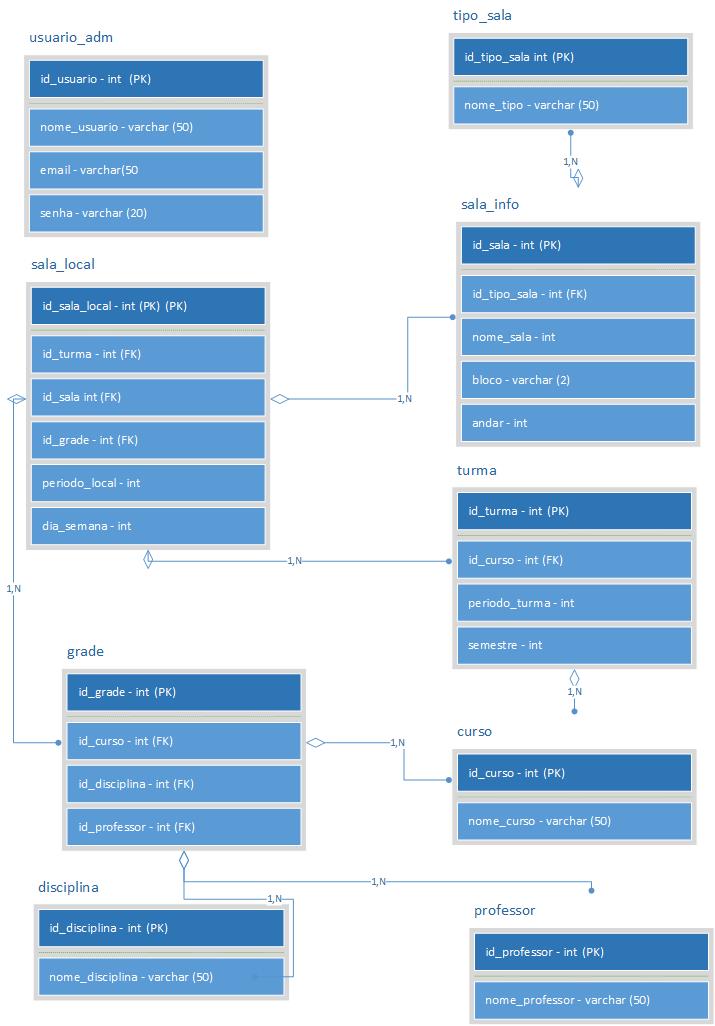


Fonte: Elaborado pelo(a) autor(a) [[14]](#footnote-14)\*

## DER

O banco de dados contém nove tabelas, sendo elas: usuario\_adm, tipo\_sala, sala\_local, sala\_info, turma, grade, curso, disciplina, professor. A tabela sala\_local, fica responsável por armazenar todas as informações sobre qual sala será ocupada, qual turma ira ocupá-la e qual disciplina essa turma terá. A tabela grade, fica responsável por armazenar as informações que contem na tabela curso, professor e disciplina, formando assim uma grade que mostra as matérias e cursos que esses professores podem dar aula. Sala\_info armazena tudo que é referente as informações da sala, qual bloco ela pertence, nome, andar e o tipo, e faz uso das informações constantes na tabela tipo\_sala. Em turma, as informações semestre, período e o curso (que puxa as informações da tabela curso), ficam armazenada para conter as informações necessárias para identificar uma turma. Usuario\_adm contem o nome, email e senha do responsável pelos cadastros.

**Fluxograma 05 - DER**



Fonte: Elaborado pelo(a) autor(a) [[15]](#footnote-15)\*

# PSEUDOCODIGO

ALGORITIMO CURSO\_FORM

Variáveis:

nome\_curso: cadeia

id: inteiro

INICIO

SE id<>null ENTAO

curso = select curso where id\_curso = id

nome\_curso <- curso.nome\_curso

senao

nome\_curso <- ‘ ’

id <- 0

FIMSE

SE id\_usuario<>null ENTAO

Escreval(“cadastrar curso”)

Escreval(“nome do curso”)

Leia(nome\_curso)

curso\_op(id, nome\_curso)

Escreval(“nome do curso)

Escreval(“editar”)

Escreval(“excluir”)

CURSO = SELECT CURSO

cont = 0

ENQUANTO cont<curso.lenght faca

Escreval(“nome do curso”, curso.nome\_curso)

Escreval(“EDITAR”)

Escreval(“EXCLUIR”)

SENAO

Login\_fomr()

FIMSE

FUNCAO curso\_op(id inteiro, nome\_curso cadeia)

Variável:

id: inteiro

nome\_curso: cadeia

INICIO

Id<-id

nome\_curso <-nome\_curso

SE id>0 ENTAO

UPDATE CURSO SET nome\_cuso WHERE id\_curso = id

SENAO

INSERT INTO CURSO NOME\_CURSO

ALGORITMO CURSO\_FORM;

FIMSE

FIMFUNCAO

FUNCAO EXCLUIR()

Variavel:

id: inteiro

tabela, campo, pagina: cadeia

INICIO

id <- id

tabela <-tabela

campo <- campo

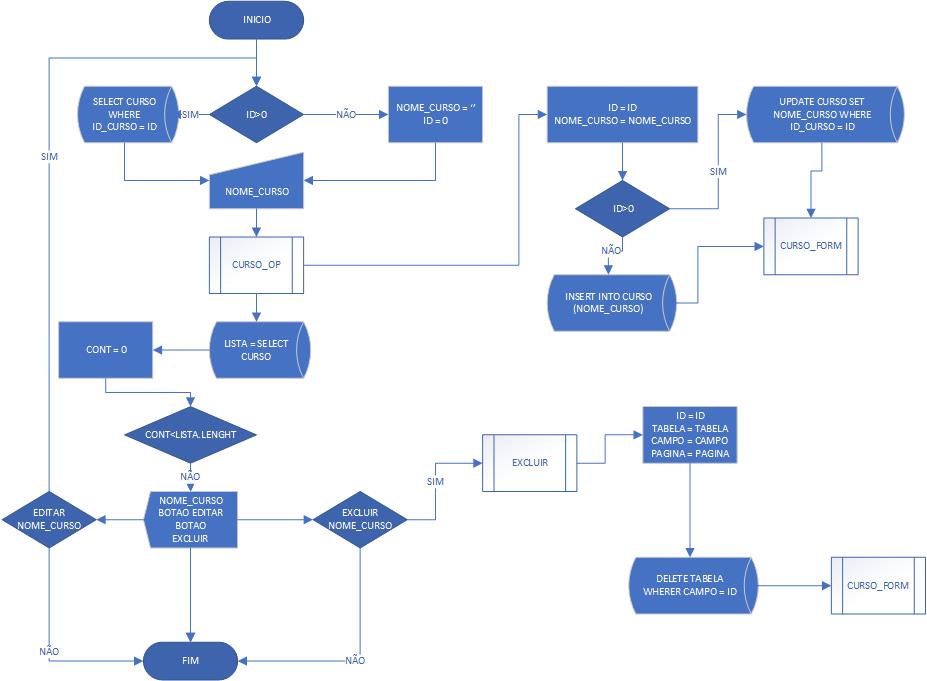
pagina <- pagina

DELETE TABELA WHERER CAMPO = ID

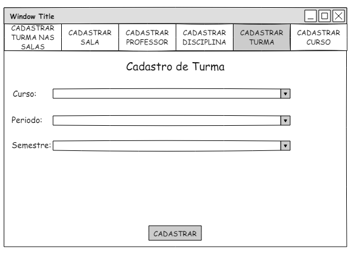
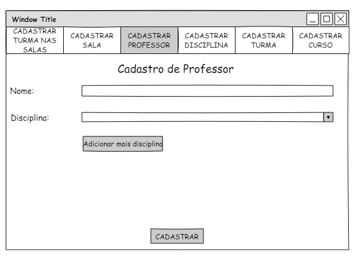
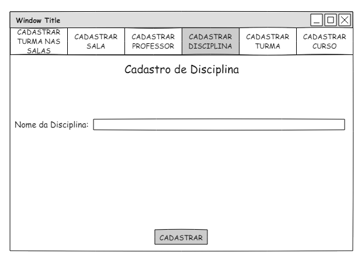
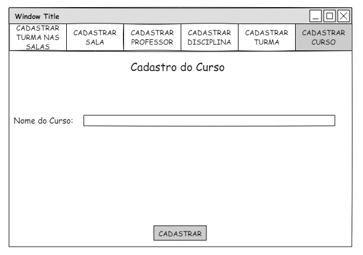
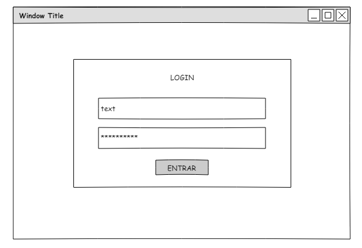
ALGORITMO CURSO\_FORM;

FIMFUNCAO

# FLUXOGRAMA LOCALIZA-SE

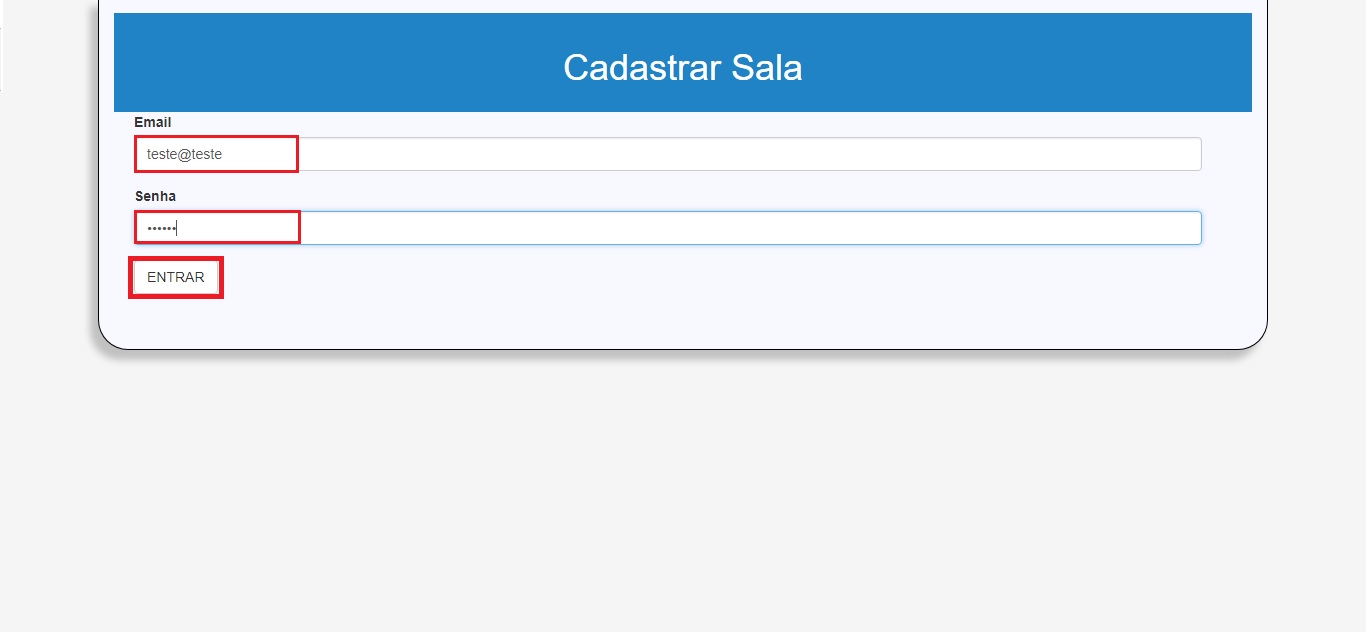


# WIREFRAME



# MAPA DO SITE

**Figura 11 – Pagina Inicial**



Fonte: Elaborado pelo(a) autor(a) [[16]](#footnote-16)\*

O usuário que estiver cadastrado, colocará nos campos “E-MAIL” e “SENHA” as informações válidas e depois clicar em “ENTRAR” para acessar o painel de cadastro.

**Figura 12 - Cadastrar Sala**



Fonte: Elaborado pelo(a) autor(a) [[17]](#footnote-17)\*

Logo após realizar o login, você será encaminhado para tela principal, onde observa-se o menu, que contém todo o conteúdo do software.

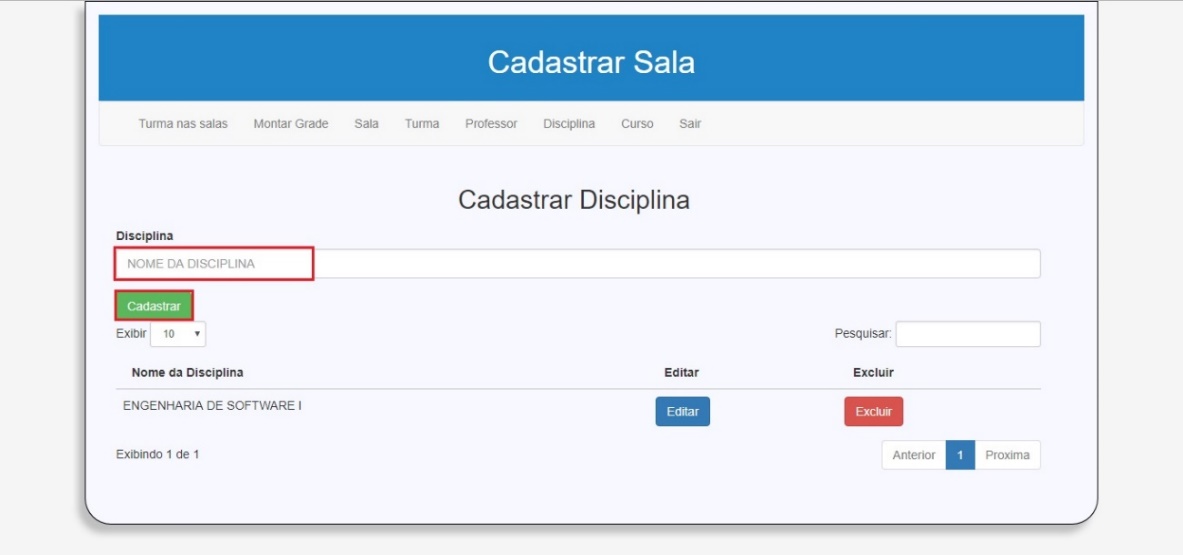
**Figura 13 - Cadastrar Curso**



Fonte: Elaborado pelo(a) autor(a) [[18]](#footnote-18)\*

Ao clicar em “CURSO”, no menu principal, será encaminhado para a parte de cadastramento de cursos, onde deverá inserir o nome do curso no campo “NOME DO CURSO” e depois clicar em “CADASTRAR”, para efetuar o cadastro.

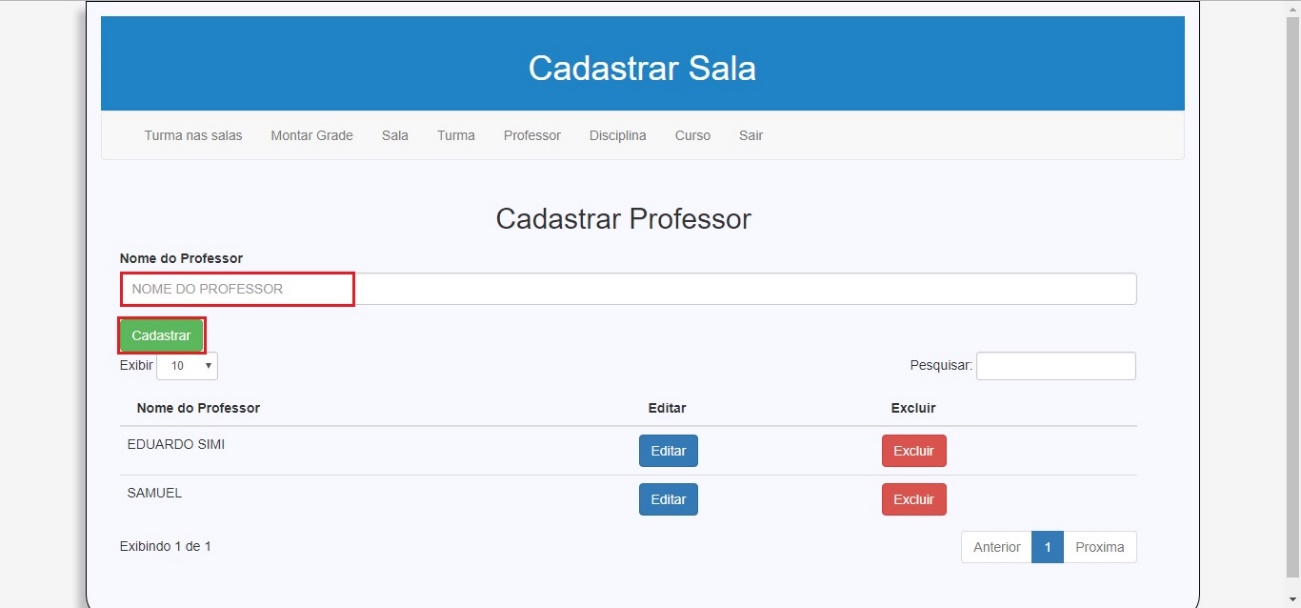
**Figura 14 - Cadastrar Disciplina**



Fonte: Elaborado pelo(a) autor(a) [[19]](#footnote-19)\*

Ao clicar em “DISCIPLINA”, no menu principal, será encaminhado para a parte de cadastramento de disciplinas, onde deverá inserir o nome da disciplina no campo “NOME DA DISCIPLINA” e depois clicar em “CADASTRAR”, para efetuar o cadastro.

**Figura 15 - Cadastrar Professor**



Fonte: Elaborado pelo(a) autor(a) [[20]](#footnote-20)\*

Ao clicar em “PROFESSOR”, no menu principal, será encaminhado para a parte de cadastramento de professores, onde deverá inserir o nome do professor no campo “NOME DA PROFESSOR” e depois clicar em “CADASTRAR”, para efetuar o cadastro.

**Figura 16 - Cadastrar Turma**



Fonte: Elaborado pelo(a) autor(a) [[21]](#footnote-21)\*

Ao clicar em “TURMA”, no menu, será encaminha para a parte de cadastramento de turma, onde terá 3 campos para ser selecionado:

1º campo com o nome “CURSO”: será listado todos os cursos da universidade;

2º campo com o nome “PERIODO”: será listado os três períodos: manhã, tarde e noite, o mesmo deve ser preenchido de acordo com o curso.

3º campo com o nome “SEMESTRE”: será listado 10 semestres, no qual será escolhido o semestre que pertence aquela turma.

**Figura 17 - Cadastrar Sala**



Fonte: Elaborado pelo(a) autor(a) [[22]](#footnote-22)\*

Ao clicar em “SALA”, no menu, será encaminha para a parte de cadastramento de sala, que contém 4 campos, 2 campos serão preenchidos e 2 campos serão selecionados.

No 1º campo com o nome “BLOCO DA SALA”, insira a letra que pertence aquela sala, no 2º campo com o nome “SALA”, escreva o número da sala, no 3º campo com o nome “ANDAR”, selecione o andar indo do 2º ANDAR até o 3º SUBSOLO, no 4º campo com o nome “TIPO DA SALA”, escolha qual tipo aquela sala pertence, por exemplo se ela for Laboratório, Sala ou etc...

**Figura 18 - Montar a Grade**

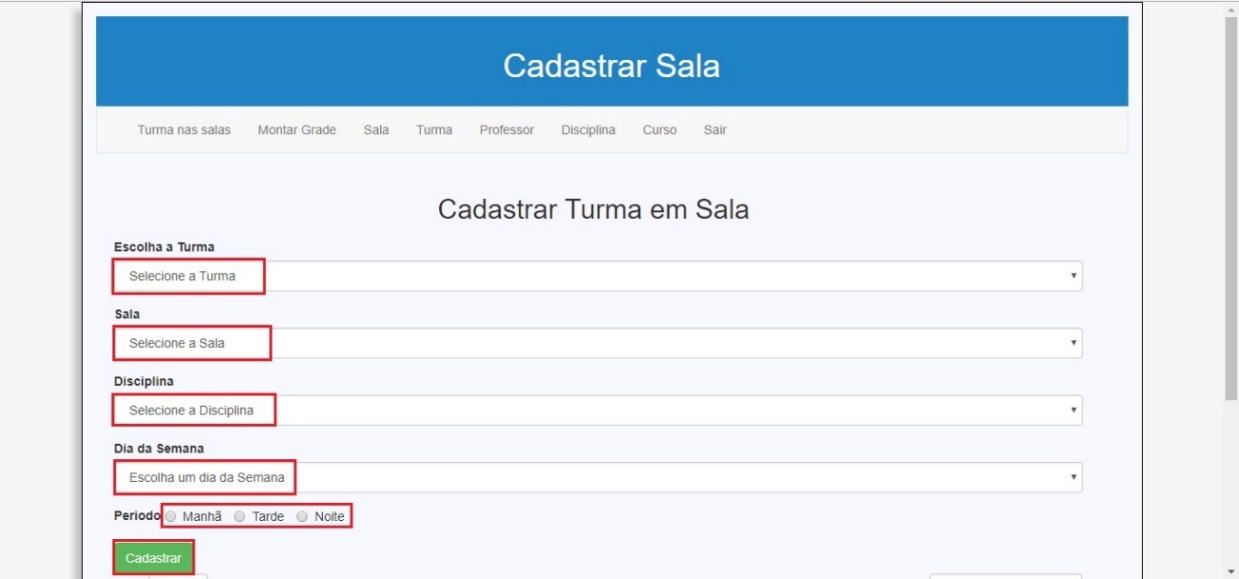


Fonte: Elaborado pelo(a) autor(a) [[23]](#footnote-23)\*

Clicando em “MONTAR GRADE”, no menu, você será encaminha para a parte de cadastramento de grade, onde irá conter 3 campos para ser selecionados.

No 1º campo com o nome “CURSO, selecione um dos cursos cadastrados, no 2º campo com o nome “DISCIPLINA”, selecione uma das disciplinas cadastradas, no 3º campo com o nome “PROFESSOR”, selecione o professor que leciona a matéria selecionada acima para o curso também selecionado.

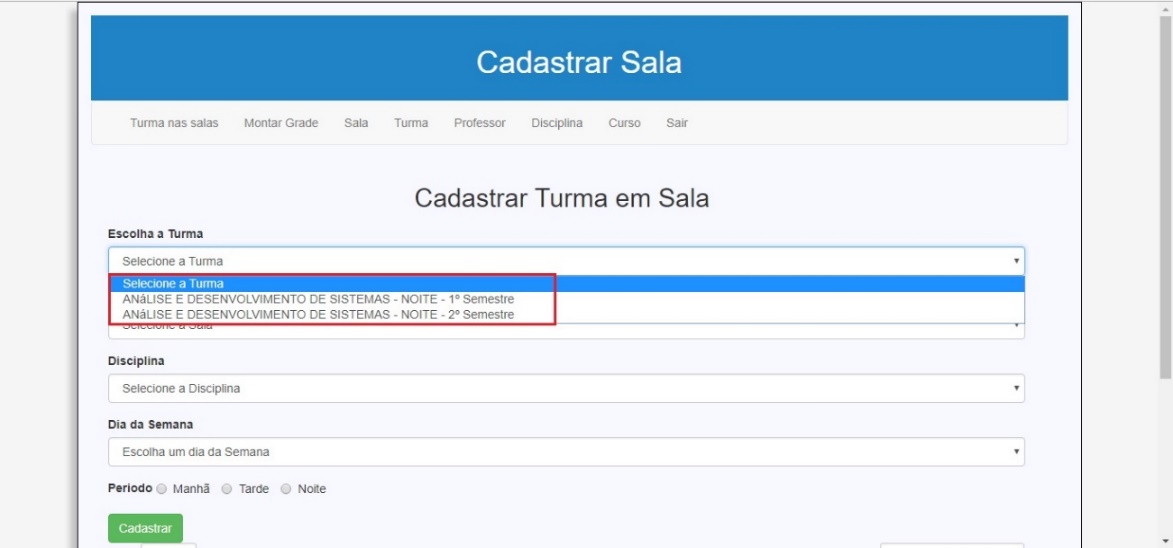
**Figura 19 - Cadastrar Turma em Sala**



Fonte: Elaborado pelo(a) autor(a) [[24]](#footnote-24)\*

Clicando em “TURMA NAS SALA”, no menu, você será encaminha para a parte de cadastramento de turma em cada sala, onde irá conter 5 campos para ser selecionados.

**Figura 20 - Cadastrar Turma em Sala**



Fonte: Elaborado pelo(a) autor(a) [[25]](#footnote-25)\*

Ao selecionar o 1º campo com o nome “ESCOLHA A TURMA”, aparecera o nome do curso, com o período e o semestre, que foi cadastrado na aba da turma.

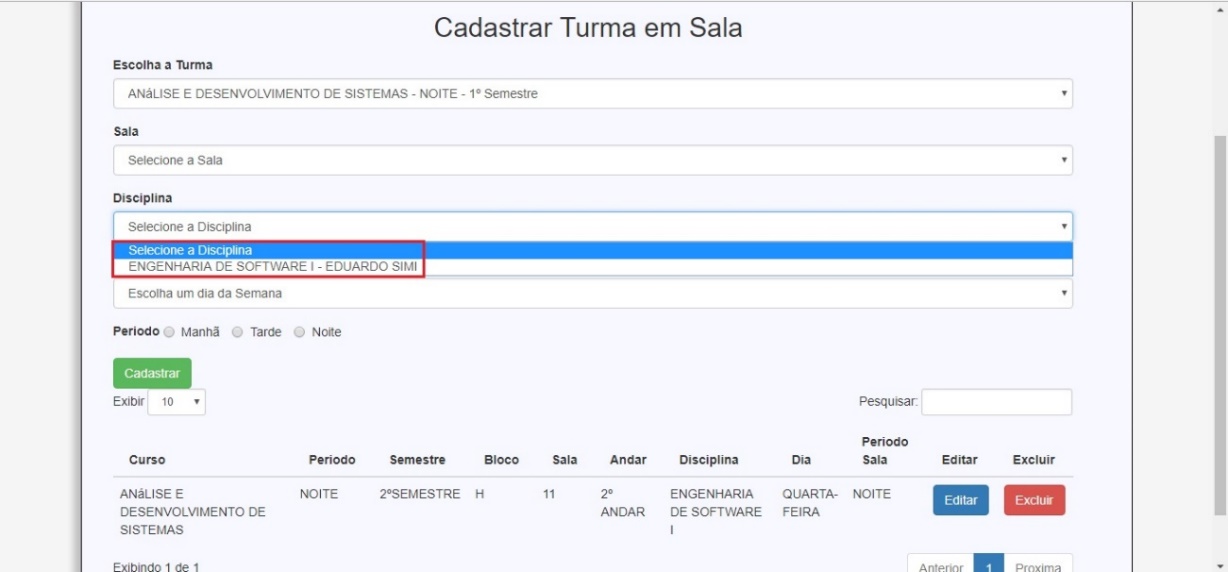
**Figura 21 - Cadastrar Turma em Sala**



Fonte: Elaborado pelo(a) autor(a) [[26]](#footnote-26)\*

Ao selecionar o 2º campo com o nome “SALA”, aparecera primeiro o tipo da sala(laboratório, sala e etc...), o bloco e o nome da sala(número) e logo após o andar que essa sala pertence.

**Figura 22 - Cadastrar Turma em Sala**



Fonte: Elaborado pelo(a) autor(a) [[27]](#footnote-27)\*

No 3º campo com o nome de “DISCIPLINA”, ira aparecer duas informações, 1º o nome da disciplina e logo depois o professor que leciona tal disciplina. Nesse campo só ira aparecer as matérias que pertence ao curso selecionado acima.

No 4º campo com o nome de “DIA DA SEMANA”, você ira escolher o dia que aquela sala será ocupada pela aquela turma, contendo os dias de Segunda á Sábado.

E no 5º campo com o nome de “PERÍODO”, selecione entre as 3 opções (manhã, tarde e noite) qual horário aquela sala será ocupada.

Em todas as páginas contém as informações listadas abaixo.

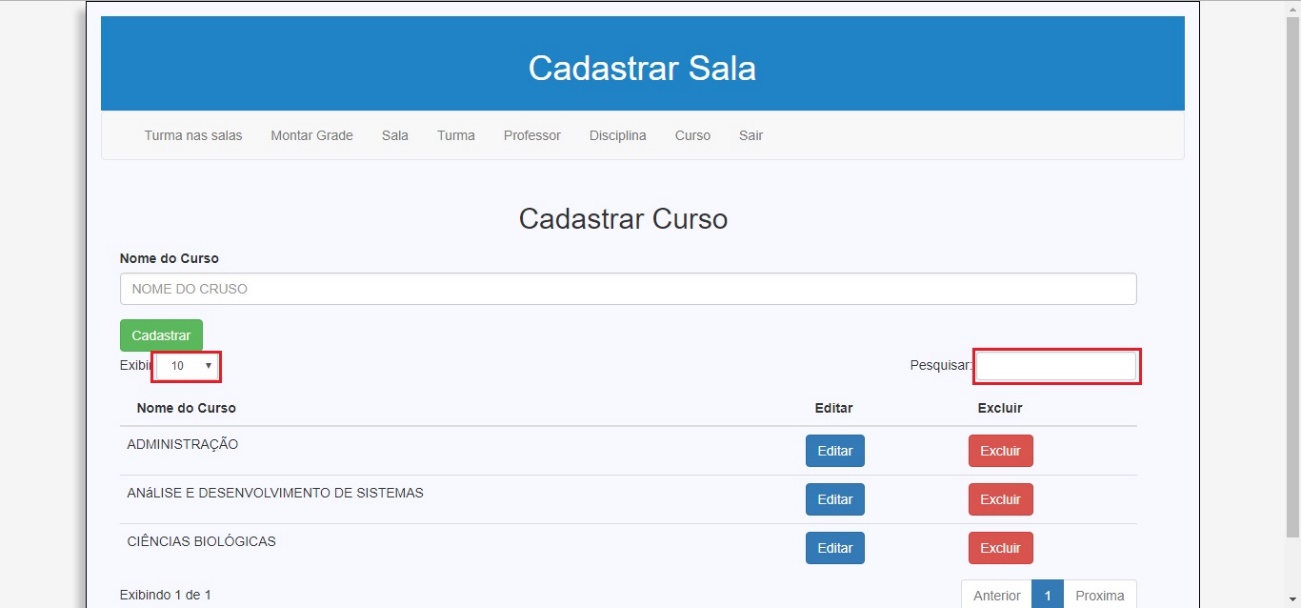
**Figura 23 - Cadastrar Curso**



Fonte: Elaborado pelo(a) autor(a) [[28]](#footnote-28)\*

Todas as informações cadastradas são exibidas logo abaixo dos campos, assim que cadastrado corretamente, o nome já ira aparecer na listagem.

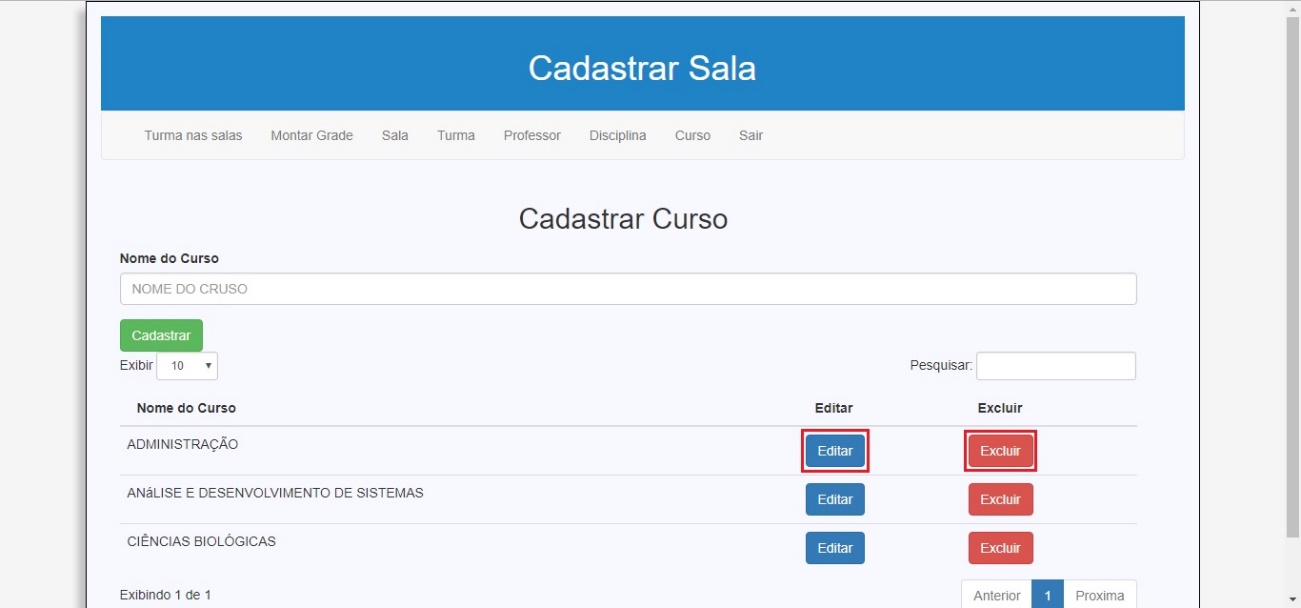
**Figura 24 - Cadastrar Curso**



Fonte: Elaborado pelo(a) autor(a) [[29]](#footnote-29)\*

Na seleção “EXIBIR” você escolhe a quantidade de informações a ser exibidas abaixo, ao lado temos o campo “PESQUISAR”, onde será feita uma filtragem de acordo com o “nome” colocado no campo.

**Figura 25 - Cadastrar Curso**



Fonte: Elaborado pelo(a) autor(a) [[30]](#footnote-30)\*

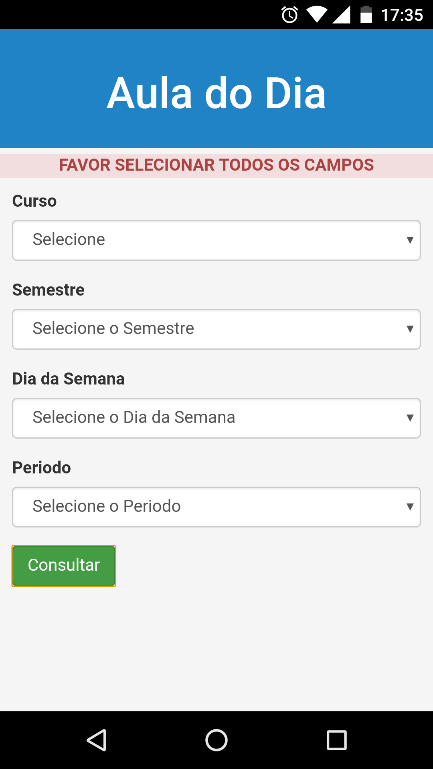
Botão editar – Ao clicar no botão “EDITAR” o(s) campo(s) acima irão ser preenchidos de acordo com as opções que foram exibidas, após isso, é só fazer as alterações e clicar em “CADASTRAR” novamente.

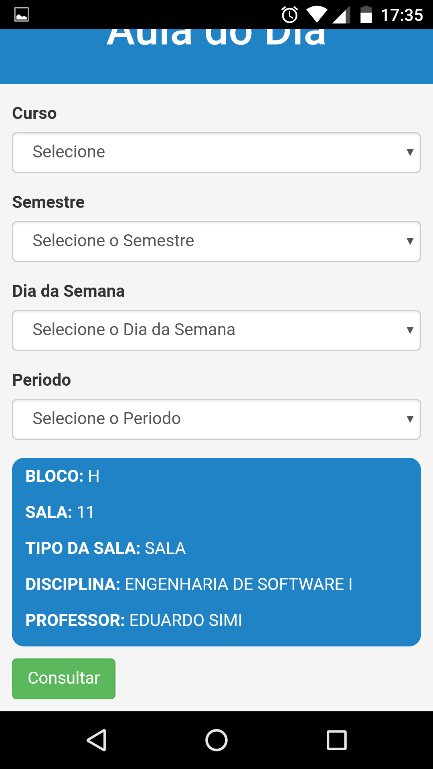
Botão excluir – Ao clicar no botão “EXCLUIR”, você terá que ter certeza do que está fazendo, pois essa opção ira apagar todas as informações seguintes, por exemplo, se a opção curso (qualquer curso) for apagada, todas as informações que forem atreladas a ela (turma, grade e turma nas salas) serão apagadas.

# MANUAL DO APLICATIVO



Após baixar o app, você irá observar 4 campos para ser selecionados, o 1º campo com o nome de “CURSO” é para selecionar o curso desejado, no 2º campo com o nome de “SEMESTRE” ira aparecer 10 semestres para ser selecionados, no 3º campo com o nome de “DIA DA SEMANA” escolha o dia da semana na qual quer saber a aula que terá e a sala que tem que ir, no 4º campo com o nome de “PERIODO” escolha o período do curso, sé é manhã tarde ou noite.

Caso o botão “CONSULTAR” seja pressionado e nenhum campo acima seja selecionado, ira aparecer uma mensagem de erro na qual ficara amostra por apenas 5 segundos.

Após selecionar todos os campos corretamente e realizar a consultar, aparecera um campo com as informações necessárias para aquele dia, as informações são as seguintes: Bloco, sala, tipo sala, disciplina e Professor. Essas informações desaparecem depois de 10 segundos.

# CONSIDERAÇÕES FINAIS

Através deste projeto é possível notar que o trabalho realizado permitiu que os integrantes do grupo desenvolvessem um trabalho em uma área em que muitos ainda não possuem experiência, nos apresentou novas metodologias de aprendizado, dando uma noção de como é o desenvolvimento de um aplicativo, auxiliou também no crescimento pessoal de todo o grupo, com a divisão das tarefas e o trabalho em equipe.

A intenção do trabalho é a de comprovar a necessidade do desenvolvimento do aplicativo capaz de suprir as necessidades dos alunos de se localizarem dentro da Universidade, principalmente no início do semestre.

Portanto existe a possibilidade de desenvolvimento e implantação do aplicativos, considerando que terá um nível de aceitação promissor e posteriormente devesse acrescentar novas funções, uma vez que passado o início do ano letivo, o aplicativo se tornará desnecessário, sendo assim, é importante frisar a necessidade de feedback com os usuários, afim de implementar novas funcionalidades de acordo com as opiniões dos mesmos.

# **REFERÊNCIAS BIBLIOGRÁFICAS**

“ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Informação e documentação: Trabalhos acadêmicos: Apresentação: NBR 14724**. Rio de Janeiro: ABNT, 2005.”

“BÓSON TREINAMENTO. **Tipos de topologia.** Disponível em:< http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/>. Acesso em: 23 Nov. 2017. ”

“DESIGN E TECNOLOGIA. **PhoneGap.** Disponível em: < https://www.getninjas.com.br/guia/design-e-tecnologia/aplicativos-para-celular-e-redes-sociais/o-que-e-phonegap>. Acesso em: 24 Nov. 2017. ”

“FOROUZAN, B. A., **Comunicação de Dados e Redes de Computadores**. Editora, São Paulo, McGraw-Hill, 2008.”

“MACEDO, D. **Topologia de rede de computadores.** Disponível em: <<http://www.diegomacedo.com.br/topologias-de-rede-de-computadores/>>. Acesso em: 21 Nov. 2017. ”

“REDES DE COMUNICAÇÃO**. Nós de redes de comunicação.** Disponível em: < https://pt.wikipedia.org/wiki/Nó\_(redes\_de\_comunicação)>. Acesso em: 22 Nov. 2017. ”

“SIGMA PLUS. **Bootstrap.** Disponível em: < https://www.sigmaplus.net.br/conhecendo-o-bootstrap>. Acesso em: 24 Nov. 2017. ”

“SUMMERVILLE, I**. Engenharia de Software.** 6. ed. São Paulo, Editra Texto, 2003.”

“TECH SOU BRASIL. **Microsoft Visio.** Disponível em: < https://www.techsoupbrasil.org.br/node/11863>. Acesso em: 23 Nov. 2017. ”

# APÊNDICE A – DOCUMENTO DE REQUISITO DO SISTEMA

IDENTIFICAÇÃO DO PROJETO

|  |  |
| --- | --- |
| **Data** | 01/11/2017 |
| **Responsável** | Bruna Pessini Biudes Silveira Molina, Carlos Augusto Specchi, Matheus Cruz Pereira, Rosana Santos De Abreu |
| **Autor (es)** | Bruna Pessini Biudes Silveira Molina, Carlos Augusto Specchi, Matheus Cruz Pereira, Rosana Santos De Abreu |
| **Identificação** |  |
| **Localização** | Sorocaba |
| **Versão** | 1.0 |

REVISÕES

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Autor** | **Descrição** |
| 11/10/2017 | 1.0 | Bruna P. B.S. Molina | Alteração dos requisitos |
| 16/10/2017 | 1.2 | Rosana S. Abreu | Modificação das ferramentas a serem utilizadas |
| 18/10/2017 | 1.3 | Carlos A. Specchi | Modificação na topologia |
| 23/10/2017 | 1.4 | Matheus C. Pereira | Alteração no site do administrador |
| 20/10/2017 | 1.5 | Matheus C. Pereira | Alteração da tela de cadastro do curso |
| 29/11/2017 | 1.6 | Bruna P. B. S. Molina | Modificação geral no documentos |

CARACTERIZAÇÃO DO PROBLEMA

|  |  |
| --- | --- |
| **Problema** | A localização dos alunos e visitantes dentro do Campus da Universidade Paulista em Sorocaba. |
| **Quem afeta** | Novos alunos e visitantes |
| **Proposta de Solução** | Desenvolver um aplicativo, que os alunos terão acesso ao período, curso, semestre, bloco e sala que os mesmos deverão se dirigir e ao acessarem os auditórios, terão acesso a agenda para verificação das palestras e eventos programados para acontecer. |

PESSOAL ENVOLVIDO NO PROCESSO DE NEGÓCIO (STAKEHOLDERS)

|  |  |  |
| --- | --- | --- |
| **Nome/Função** | **Descrição da Função** | **Responsabilidade** |
| Bruna P. B.S. Molina | Efetuar o levantamento dos requisitos | Analisar as necessidades e definir as prioridades. |
| Carlos A. Specchi | Definir a topologia. | Analisar os equipamentos necessários para o bom desempenho do aplicativo. |
| Rosana S. Abreu | Definir as ferramentas a serem utilizadas . | Fazer o levantamento das melhores ferramentas para auxiliar no desenvolvimento do banco de dados, do site e do aplicativo. |
| Matheus C. pereira | Desenvolver o banco de dados e o aplicativo. | Criar um banco de dados utilizando os requisitos, desenvolver um site para o administrador atualizar as informações e criar o aplicativo. |

REQUISITOS FUNCIONAIS

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Descrição** |
| 1 |  | Diminuir o tempo perdido localizando as salas. |
| 2 |  | Identificação das salas de acordo com os cursos. |
| 3 |  | Agilidade no encontro das salas e laboratórios. |
| 4 |  | Controle dos usuários que terão acesso ao banco de dados. |

REQUISITOS NÃO FUNCIONAIS

|  |  |
| --- | --- |
| **ID** | **Descrição** |
| 1.1 | A base de dados deve ser protegida para acesso apenas de usuários autorizados. |
| 1.2 | O tempo de resposta do sistema não deve ultrapassar 30 segundos. |

PRECEDÊNCIA E PRIORIDADE

|  |  |  |
| --- | --- | --- |
| **ID do Requisito** | **Precedência** | **Prioridade** |
| 4 | 1 | Alta |
| 2 | 2 | Alta |
| 1 | 3 | Média |
| 3 | 4 | Baixa |

# APÊNDICE B – CODIGO FONTE

<?php require\_once("header.php");

require\_once("conexao.php");

if (isset($\_GET['id'])) {

$id = $\_GET['id'];

$query = "SELECT sala\_local.\*, turma.\*, curso.\*, sala\_info.\*, tipo\_sala.\*, grade.\*, disciplina.\*, professor.\*

FROM sala\_local

INNER JOIN turma ON sala\_local.id\_turma = turma.id\_turma

INNER JOIN curso ON turma.id\_curso = curso.id\_curso

INNER JOIN sala\_info ON sala\_local.id\_sala = sala\_info.id\_sala

INNER JOIN tipo\_sala ON sala\_info.id\_tipo\_sala = tipo\_sala.id\_tipo\_sala

INNER JOIN grade ON sala\_local.id\_grade = grade.id\_grade

INNER JOIN disciplina ON grade.id\_disciplina = disciplina.id\_disciplina

INNER JOIN professor ON grade.id\_professor = professor.id\_professor

WHERE id\_sala\_local = $id";

$infoSalaLocal = mysqli\_query($conexao, $query);

$editarSalaLocal = mysqli\_fetch\_assoc($infoSalaLocal);

$id\_turma = $editarSalaLocal['id\_turma'];

$id\_sala = $editarSalaLocal['id\_sala'];

$id\_grade = $editarSalaLocal['id\_grade'];

$nome\_curso = $editarSalaLocal['nome\_curso'];

$nome\_tipo = $editarSalaLocal['nome\_tipo'];

$nome\_sala = $editarSalaLocal['nome\_sala'];

$nome\_disciplina = $editarSalaLocal['nome\_disciplina'];

$bloco = $editarSalaLocal['bloco'];

$andar = $editarSalaLocal['andar'];

$dia\_semana = $editarSalaLocal['dia\_semana'];

$periodo\_local = $editarSalaLocal['periodo\_local'];

$semestre = $editarSalaLocal['semestre'];

$periodo\_turma = $editarSalaLocal['periodo\_turma'];

} else {

$id = 0;

$id\_turma = '';

$id\_sala = '';

$id\_grade = '';

$nome\_curso = '';

$nome\_tipo = '';

$nome\_sala = '';

$nome\_disciplina = '';

$bloco = '';

$andar = '';

$dia\_semana = '';

$periodo\_local = '';

$semestre = '';

$periodo\_turma = '';

}

if (isset($\_SESSION['id\_usuario'])) { ?>

<div class="titulo">

<h2>Cadastrar Turma em Sala</h2>

</div>

<form action="turmaSala\_op.php" method="post">

<input type="hidden" name="id" value="<?=$id?>">

<div class="form-group">

<label for="exampleInputEmail1">Escolha a Turma</label>

<select name="id\_turma" id="id\_turma" class="form-control" required>

<option value="">Selecione a Turma</option>

<?php

$query = "SELECT turma.\*, curso.\* FROM turma LEFT JOIN curso ON turma.id\_curso = curso.id\_curso ORDER BY nome\_curso, semestre ASC";

$turma = mysqli\_query($conexao, $query);

while($listaTurma = mysqli\_fetch\_assoc($turma)){

if($listaTurma['periodo\_turma'] == 1){ ?>

<option value="<?=$listaTurma['id\_turma']?>" <?php if ($id\_turma == $listaTurma['id\_turma']) echo "selected";?>><?=$listaTurma['nome\_curso']?> - MANHÃ - <?=$listaTurma['semestre']; ?>º Semestre</option>

<?php } elseif ($listaTurma['periodo\_turma'] == 2) { ?>

<option value="<?=$listaTurma['id\_turma']?>" <?php if ($id\_turma == $listaTurma['id\_turma']) echo "selected";?>><?=$listaTurma['nome\_curso']?> - TARDE - <?=$listaTurma['semestre']; ?>º Semestre</option>

<?php } else { ?>

<option value="<?=$listaTurma['id\_turma']?>" <?php if ($id\_turma == $listaTurma['id\_turma']) echo "selected";?>><?=$listaTurma['nome\_curso']?> - NOITE - <?=$listaTurma['semestre']; ?>º Semestre</option>

<?php }

} ?>

</select>

</div>

<div class="form-group">

<label for="exampleInputEmail1">Sala</label>

<select name="id\_sala" id="" class="form-control">

<option value="">Selecione a Sala</option>

<?php

$query = "SELECT sala\_info.\*, tipo\_sala.\* FROM sala\_info INNER JOIN tipo\_sala ON sala\_info.id\_tipo\_sala = tipo\_sala.id\_tipo\_sala ORDER BY nome\_sala ASC";

$sala = mysqli\_query($conexao, $query);

while ($listaSala = mysqli\_fetch\_assoc($sala)) {

switch ($listaSala['andar']) {

case 1: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - 2º ANDAR</option>

<?php break;

case 2: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - 1º ANDAR</option>

<?php break;

case 3: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - TERREO</option>

<?php break;

case 4: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - 1º SUBSOLO</option>

<?php break;

case 5: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - 2º SUBSOLO</option>

<?php break;

default: ?>

<option value="<?=$listaSala['id\_sala'];?>" <?php if ($id\_sala == $listaSala['id\_sala']) echo "selected";?>><?=$listaSala['nome\_tipo'] ?> - BLOCO <?=$listaSala['bloco']; ?> - <?=$listaSala['nome\_sala']?> - 3º SUBSOLO</option>

<?php break;

}

} ?>

</select>

</div>

<div class="form-group">

<label for="exampleInputEmail1">Disciplina</label>

<select name="id\_grade" id="option" class="form-control">

<option value="">Selecione a Disciplina</option>

<?php

$query = "SELECT grade.\*, disciplina.\*, professor.\* FROM grade

INNER JOIN disciplina ON grade.id\_disciplina = disciplina.id\_disciplina

INNER JOIN professor ON grade.id\_professor = professor.id\_professor

ORDER BY nome\_disciplina ASC";

$disciplina = mysqli\_query($conexao, $query);

while ($listaDisciplina = mysqli\_fetch\_assoc($disciplina)) { ?>

<option value="<?=$listaDisciplina['id\_grade']; ?>" <?php if ($id\_grade == $listaDisciplina['id\_grade']) echo "selected"; ?>><?=$listaDisciplina['nome\_disciplina']; ?> - <?=$listaDisciplina['nome\_professor']; ?></option>

<?php } ?>

<script>

$("#id\_turma").change(function() {

$.ajax({url: "ajax.php?id="+$('#id\_turma :selected').val(),

success: function(result){

$('#option').html(result);

}});

});

</script>

</select>

</div>

<div class="form-group">

<label for="exampleInputEmail1">Dia da Semana</label>

<select name="dia\_semana" id="" class="form-control" required>

<option value="">Escolha um dia da Semana</option>

<option name="dia\_semana" value="1" <?php if ($dia\_semana == 1) echo "selected";?>>SEGUNDA-FEIRA</option>

<option name="dia\_semana" value="2" <?php if ($dia\_semana == 2) echo "selected";?>>TERÇA-FEIRA</option>

<option name="dia\_semana" value="3" <?php if ($dia\_semana == 3) echo "selected";?>>QUARTA-FEIRA</option>

<option name="dia\_semana" value="4" <?php if ($dia\_semana == 4) echo "selected";?>>QUINTA-FEIRA</option>

<option name="dia\_semana" value="5" <?php if ($dia\_semana == 5) echo "selected";?>>SEXTA-FEIRA</option>

<option name="dia\_semana" value="6" <?php if ($dia\_semana == 6) echo "selected";?>>SABADO</option>

</select>

</div>

<div class="form-group">

<label class="radio\_inline">Periodo</label>

<label class="radio-inline">

<input type="radio" name="periodo\_local" id="inlineRadio1" value="1" <?php if ($periodo\_local == 1) echo "checked";?>> Manhã

</label>

<label class="radio-inline">

<input type="radio" name="periodo\_local" id="inlineRadio2" value="2" <?php if ($periodo\_local == 2) echo "checked";?>> Tarde

</label>

<label class="radio-inline">

<input type="radio" name="periodo\_local" id="inlineRadio3" value="3" <?php if ($periodo\_local == 3) echo "checked";?>> Noite

</label>

</div>

<button type="submit" class="btn btn-success">Cadastrar</button>

</form>

<table class="table">

<thead>

<th>Curso</th>

<th>Periodo</th>

<th>Semestre</th>

<th>Bloco</th>

<th>Sala</th>

<th>Andar</th>

<th>Disciplina</th>

<th>Dia</th>

<th>Periodo Sala</th>

<th>Editar</th>

<th>Excluir</th>

</thead>

<tbody>

<?php //Fazer listagem de cursos

$query = "SELECT sala\_local.\*, turma.\*, curso.\*, sala\_info.\*, tipo\_sala.\*, grade.\*, disciplina.\*, professor.\*

FROM sala\_local

INNER JOIN turma ON sala\_local.id\_turma = turma.id\_turma

INNER JOIN curso ON turma.id\_curso = curso.id\_curso

INNER JOIN sala\_info ON sala\_local.id\_sala = sala\_info.id\_sala

INNER JOIN tipo\_sala ON sala\_info.id\_tipo\_sala = tipo\_sala.id\_tipo\_sala

INNER JOIN grade ON sala\_local.id\_grade = grade.id\_grade

INNER JOIN disciplina ON grade.id\_disciplina = disciplina.id\_disciplina

INNER JOIN professor ON grade.id\_professor = professor.id\_professor

ORDER BY nome\_curso, semestre ASC";

$resulSalaLocal = mysqli\_query($conexao, $query);

while($lista = mysqli\_fetch\_assoc($resulSalaLocal)){ ?>

<tr>

<td><?=$lista['nome\_curso']?></td>

<?php if ($lista['periodo\_turma'] == 1) { ?>

<td>MANHÃ</td>

<?php } elseif ($lista['periodo\_turma'] == 2) { ?>

<td>TARDE</td>

<?php } else { ?>

<td>NOITE</td>

<?php } ?>

<td><?=$lista['semestre']?>ºSEMESTRE</td>

<td><?=$lista['bloco']?></td>

<td><?=$lista['nome\_sala']?></td>

<?php if ($lista['andar'] == 1) { ?>

<td>2º ANDAR</td>

<?php } elseif ($lista['andar'] == 2) { ?>

<td>1º ANDAR</td>

<?php } elseif ($lista['andar'] == 3) { ?>

<td>TERREO</td>

<?php } elseif ($lista['andar'] == 4) { ?>

<td>1º SUBSOLO</td>

<?php } elseif ($lista['andar'] == 5) { ?>

<td>2º SUBSOLO</td>

<?php } else { ?>

<td>3º SUBSOLO</td>

<?php } ?>

<td><?=$lista['nome\_disciplina']?></td>

<?php switch ($lista['dia\_semana']) {

case 1: ?>

<td>SEGUNDA-FEIRA</td>

<?php break;

case 2: ?>

<td>TERÇA-FEIRA</td>

<?php break;

case 3: ?>

<td>QUARTA-FEIRA</td>

<?php break;

case 4: ?>

<td>QUINTA-FEIRA</td>

<?php break;

case 5: ?>

<td>SEXTA-FEIRA</td>

<?php break;

case 6: ?>

<td>SABADO</td>

<?php break;

default: ?>

<td>Error</td>

<?php break;

}

switch ($lista['periodo\_local']) {

case 1: ?>

<td>MANHÃ</td>

<?php break;

case 2: ?>

<td>TARDE</td>

<?php break;

default: ?>

<td>NOITE</td>

<?php break;

} ?>

<td><a href="turmaSala\_form.php?id=<?=$lista['id\_sala\_local']?>"><button type="button" he class="btn btn-primary">Editar</button></a></td>

<td><a href="excluir.php?tabela=sala\_local&campo=id\_sala\_local&pagina=turmaSala\_form&id=<?=$lista['id\_sala\_local']?>"><button type="button" class="btn btn-danger">Excluir</button></a></td>

</tr>

<?php } ?>

</tbody>

</table>

<?php } else {

header("Location: login\_form.php");

} ?>

<?php require\_once("footer.php"); ?>

1. \* Fluxograma 01 – Modelo cascata. Elaborado pelo(a) autor(a), 2017. [↑](#footnote-ref-1)
2. \* Fluxograma 02 – Desenvolvimento revolucionario. Elaborado pelo(a) autor(a), 2017. [↑](#footnote-ref-2)
3. \* Fluxograma 03 – Desenvolvimento incremental. Elaborado pelo(a) autor(a), 2017. [↑](#footnote-ref-3)
4. \* Figura 01 – Exemplo de um modelo espiral. Elaborado pelo(a) autor(a), 2017. [↑](#footnote-ref-4)
5. \* Figura 02 – Topologia de Barramento. Disponivel em < http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017.

   . [↑](#footnote-ref-5)
6. \* Figura 03 – Topologia Estrela. Disponivel em < http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017. [↑](#footnote-ref-6)
7. \* Figura 04 – Topologia Anel. Disponivel em < http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017. [↑](#footnote-ref-7)
8. \* Figura 05 – Topologia Hibrida ou Mista.Disponivel em <http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017.

   . [↑](#footnote-ref-8)
9. \* Figura 06 – Topologia em Malha. Disponivel em < http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017. [↑](#footnote-ref-9)
10. \* Figura 07 – Topologia Ponto-a-Multiponto. Disponivel em: <http://www.bosontreinamentos.com.br/ redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017. [↑](#footnote-ref-10)
11. \* Figura 08 – Topologia PIM. Disponivel em < http://www.bosontreinamentos.com.br/redes-computadores/topologias-de-redes-curso-de-redes-de-computadores/ >. Acesso em: 20/11/2017.

    . [↑](#footnote-ref-11)
12. \* Figura 09 – Logo Microsoft Visio.Disponivel em < https://blogs.office.com/en-us/visio/?eu=true >. Acesso em: 17/11/2017.

    . [↑](#footnote-ref-12)
13. \* Figura 10 – PhoneGap. Disponivel em < https://www.alphasoftware.com/v4/phonegap-build/ >. Acesso em: 13/11/2017. [↑](#footnote-ref-13)
14. \* Fluxograma 04 – MER. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-14)
15. \* Fluxograma 05 – DER. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-15)
16. \* Figura 11 – Pagina Inicial. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-16)
17. \* Figura 12 – Cadastrar Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-17)
18. \* Figura 13 – Cadastrar Curso. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-18)
19. \* Figura 14 – Cadastrar Disciplina. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-19)
20. \* Figura 15 – Cadastrar Professor. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-20)
21. \* Figura 16 – Cadastrar Turma. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-21)
22. \* Figura 17 – Cadastrar Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-22)
23. \* Figura 18 – Montar Grade. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-23)
24. \* Figura 19 – Cadastrar Turma em Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-24)
25. \* Figura 20 – Cadastrar Turma em Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-25)
26. \* Figura 21 – Cadastrar Turma em Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-26)
27. \* Figura 22 – Cadastrar Sala. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-27)
28. \* Figura 23 – Cadastrar Curso. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-28)
29. \* Figura 24 – Cadastrar Curso. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-29)
30. \* Figura 25 – Cadastrar Curso. Elaborado pelo(a) autor(a), 2017.

    . [↑](#footnote-ref-30)