INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SUL-RIO-GRANDENSE - CÂMPUS PASSO FUNDO
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

ROSENI MARINES GOULART

PROTÓTIPO DE UM SISTEMA WEB PARA A COMISSÃO PERMANENTE DO PESSOAL DOCENTE DO IFSUL CAMPUS PASSO FUNDO

Orientadora
Carmen Vera Scorsatto Brezolin

PASSO FUNDO
2017

ROSENI MARINES GOULART

PROTÓTIPO DE UM SISTEMA WEB PARA A COMISSÃO PERMANENTE DO PESSOAL DOCENTE DO IFSUL CAMPUS PASSO FUNDO

Monografia apresentada ao Curso de Tecnologia em Sistemas para Internet do Instituto Federal Sul-rio-grandense, Câmpus Passo Fundo, como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet.

Orientadora: Carmen Vera Scorsatto Brezolin

PASSO FUNDO

2017

ROSENI MARINES GOULART

PROTÓTIPO DE UM SISTEMA WEB PARA A COMISSÃO PERMANENTE DO PESSOAL DOCENTE DO IFSUL CAMPUS PASSO FUNDO

Trabalho de Conclusão de Curso aprovado em ____/____/____ como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet

Banca Examinadora:

_____

Carmen Vera Scorsatto Brezolin

_____

Rafael Marisco Bertei

_____

Telmo De Cesaro Junior

_____

Adilso Nunes de Souza

PASSO FUNDO

2017

DEDICATÓRIA

Ao meu pai Valter Goulart,
Pela compreensão e o estímulo em todos os momentos.
A minha mãe Inês Goulart (in memorian),
Que mesmo ausente, posso sentir sua força me iluminando.
Ao meu esposo Jhonata Ferreira,
Por me apoiar e ajudar em todo este tempo que estamos juntos.
A professora Carmen Scorsatto,
Por sempre ter acreditado em meu potencial.
Aos amigos Gabriela e Douglas Karczeski,
Por sempre me incentivarem a acreditar em meus sonhos.

EPÍGRAFE

“Desconfie do destino e acredite em você.
Gaste mais horas realizando que sonhando,
fazendo que planejando,
vivendo que esperando porque,
embora quem quase morre esteja vivo,
quem quase vive já morreu.”

Sarah Westphal

RESUMO

O corpo docente do IFSUL campus Passo Fundo a cada vinte e quatro meses, ao cumprirem uma carga horária com atividades extras, podem solicitar a Comissão Permanente do Pessoal Docente a progressão do plano de carreira, esta progressão traz benefícios aos professores, principalmente em seus salários. Percebeu-se a necessidade de que a entrega desta documentação fosse feita de forma mais dinâmica, visto que atualmente é entregue toda a documentação no findar deste tempo, ocasionando sobrecarga aos docentes. Com a implementação de um sistema web, os docentes poderão antecipar o envio dessa documentação conforme as atividades forem sendo realizadas, melhorando então a organização desta documentação e diminuindo a sobrecarga dos docentes envolvidos no processo. A proposta para este trabalho é desenvolver um protótipo para fazer o gerenciamento dos documentos necessários para comprovar as atividades dos docentes.

Palavras-chave: PHP. Sistema Web.

ABSTRACT

The faculty of the IFSUL campus Passo Fundo every twenty-four months, while fulfilling a workload with extra activities may ask the Standing Committee of Teaching Staff to progress the career plan, this progression brings benefits to teachers, especially in their salaries. The need for the delivery of this documentation was made more dynamic, since all the documentation is now delivered at the end of this time, causing an overload to the teachers. With the implementation of a web system, teachers can anticipate sending this documentation as the activities are being carried out, improving the organization of this documentation and reducing the overload of the teachers involved in the process. The proposal for this work is to develop a prototype to manage the documents necessary to prove the activities of the teachers.

Keywords: PHP. Web system.

LISTA DE FIGURAS

Figura 1 - HTML 18
Figura 2 - Servidor web 19
Figura 3 - Apache web Server 20
Figura 4 - Ranking dos SGBD de acordo com popularidade 21
Figura 5 - Ranking Linguagens de Programação IEEE 2016 25
Figura 6 - PHP 26
Figura 7 - Definir cor a trecho em HTML 27
Figura 8 - CSS 28
Figura 9- Diagrama de Casos de Uso 37
Figura 10 - Modelo Entidade Relacionamento 39
Figura 11- Conexão com Banco de Dados 43
Figura 12 - Modelo Parte 1 44
Figura 13 - Modelo Parte 2 45
Figura 14 - Modelo Parte 3 46
Figura 15 - Controle 47
Figura 16 - Visão Parte 1 48
Figura 17 - Visão Parte 2 48
Figura 18 - Visão Parte 3 49
Figura 19 - Visão em Tela 50
Figura 20 - Tela Avaliador 51
Figura 21 - Tela Professor 52
Figura 22 - Base de Dados PHP 53
Figura 23 - login.php 53
Figura 24 - login.html 1 54
Figura 25 - login.html 2 55
Figura 26 - Cookie Completo 55
Figura 27 - Cookie Limitado 56
Figura 28 - Organização de arquivos 56
Figura 29 - Tele inicial com login 57
Figura 30 – logout.php 57
Figura 31 - Upload de Arquivo 59
Figura 32 - Pasta Upload 59

Figura 33 - Tela de Avaliação 60
Figura 34 - Link para mostrar arquivo 60

LISTA DE TABELA

Tabela 1– Classificação e finalidade dos elementos de interface 30

LISTA DE ABREVIATURAS E SIGLAS

IFSUL – Instituto Federal Sul-rio-grandense

TCC – Trabalho de Conclusão de Curso

SQL - Structured Query Language, ou Linguagem de Consulta Estruturada

IDEs - *Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado*

CPPD - Comissão Permanente do Pessoal Docente

CSS - *Cascading Style Sheets, ou estilo em cascata*

PHP - Hypertext Preprocessor Hypertext Transfer Protocol (HTTP)

SGBD - Sistema de Gestão de Banco de Dados

UML - *Unified Modeling Languagem, ou Linguagem de Modelagem Unificada*

MVC - Modelo-visão-controle

W3C – World Wide Web Consortium

SUMÁRIO

1. INTRODUÇÃO 12

2. OBJETIVOS 14
2.1 OBJETIVOS ESPECÍFICOS 14

3. DESENVOLVIMENTO 15
3.1 TECNOLOGIAS USADAS 15
3.1.1 Protocolo HTTP 15
3.1.2 HTML 4 17
3.1.3 Servidor WEB 18
3.1.4 Banco De Dados 20
3.1.5 PHP 5 22
3.1.6 Estilos, Funções e Visual 26
3.1.7 XAMPP 33
3.2 RECURSOS PARA ESPECIFICAÇÃO 33
3.2.1 Modelagem Relacional 33
3.3 UML 35
3.3.1 Diagrama de Casos de Uso 35
3.4 ESTUDO DE CASO 36
3.4.1 Diagrama de Casos de Uso 36
3.4.2 Modelo Entidade Relacionamento 38
3.4.3 Modelagem Do Bando De Dados 39
3.4.4 Conexão Com Banco De Dados 42
3.4.5 MVC 43
3.4.1 Controle 46
3.4.2 Visão 47
3.5 TIPOS DE ACESSO 51
3.5.1 Usuário Avaliador 51
3.5.2 Usuário Professor 52
3.5.3 Login 52
3.5.4 Controle De Acesso 55
3.5.5 Logout 57
3.6 UPLOAD DO ARQUIVO 58

3.7 DOWNLOAD DO ARQUIVO 59

4. CONSIDERAÇÕES FINAIS 61

5. TRABALHOS FUTUROS 62

REFERÊNCIAS 63

ANEXOS 64

1. INTRODUÇÃO

Um sistema que esteja disponível em qualquer lugar pode trazer diversas facilidades aos seus usuários. Uma aplicação Web permite que seu acesso seja feito de qualquer lugar, desde que o usuário tenha acesso a Internet e um navegador, sem a necessidade de instalar o sistema no dispositivo e executam nas mais diversas plataformas.

Através de uma aplicação que está sempre disponível, onde quer que o usuário esteja, surge uma variada gama de possibilidades de aplicações para o uso pessoal, ou até mesmo de uma organização, que abrangem de entretenimento até uma aplicação mais complexa que envolva uma sequência de processos a fim de resolver um problema.

Outra vantagem é que aplicações Web quando passam por atualizações, estas são feitas diretas em seus servidores, e o usuário final não precisa ficar se preocupando com versões, pois o sistema será sempre o atual.

Outro conceito importante é o de mobilidade que é o de o sistema funcionar nas mais diversas plataformas, claro que estas aplicações devem possuir funções capazes de adaptar o seu uso nos mais diversos dispositivos que possam vir a serem utilizados, e o de disponibilidade, permitir acesso de qualquer lugar, regendo os princípios da Internet.

A partir da análise sobre a implementação de um sistema de controle de documentos que são enviados por diversos usuários, a plataforma que melhor se encaixou para suprir as necessidades, é a de um sistema Web. O protótipo para o sistema proposto trata de fazer o gerenciamento dos documentos necessários para comprovar as atividades que os docentes do Instituto Federal Sul-rio-grandense (IFSUL), campus Passo Fundo, realizam dentro de um determinado tempo para que seja possível a progressão funcional. Sendo possível que cada docente tenha seu acesso único, e que possa controlar e enviar suas atividades para avaliação durante o decorrer do tempo determinado, e não só quando estiver com todas as tarefas completas ao findar desse tempo. O avaliador de tarefas tem acesso a estas informações fornecidas pelos docentes e avalia com antecedência as atividades enviadas, possibilitando maior organização e permitindo melhor aproveitamento de

tempo das partes envolvidas, pois cada usuário, tanto docente como avaliador, tem acesso ao sistema em qualquer lugar que estiver.

Com a implementação desta aplicação, o gerenciamento desses documentos se darão de forma dinâmica, mais organizada, entre outras vantagens, tornando o processo mais eficaz.

1.1. MOTIVAÇÃO

Os docentes do IFSUL campus Passo Fundo, para fins de progressão funcional, podem participar de diversas atividades especificas, e a cada vinte e quatro meses, devem entregar um relatório com todas estas atividades descritas e com seus comprovantes à CPPD, Comissão Permanente do Pessoal Docente, que possuem dentre diversas funções o controle progressão funcional. Atualmente esses relatórios e documentos são entregues pelos docentes de forma manual, ao encerrar o prazo juntamente com uma planilha do Excel, com as atividades que foram exercidas no decorres desse tempo entre os vinte e quatro meses. Por mais que o professor tente manter os seus documentos e planilhas organizadas, ao findar o tempo, ele tem o trabalho de ter que juntar todos estes arquivos e documentos e entregar ao avaliador da CPPD. Isso também acarreta na sobrecarga de trabalho do avaliador, visto que este terá diversos documentos e planilha para conferir e avaliar em um determinado tempo.

Percebeu-se que se faz necessário alguma ferramenta que facilite os envios dos dados e documentos de forma mais organizada.

A construção de um sistema web que faça este controle tornará mais organizado e dinâmico o gerenciamento das informações dos docentes.

2. OBJETIVOS

Fazer o estudo de diferentes tecnologias para o desenvolvimento de um protótipo de um sistema Web em que torne possível o envio, armazenamento e permitir que o avaliador possa fazer a conferência dos documentos enviados para a Comissão Permanente do Pessoal Docente (CPPD) do IFSUL campus Passo Fundo.

2.1 OBJETIVOS ESPECÍFICOS

Para que seja possível a realização do trabalho proposto será necessário estudar a tecnologia PHP, com o uso de um framework chamado jQuery UI que será o responsável pela parte visual do sistema, para que torne possível a implementação de um sistema em que o usuário possa fazer o cadastro de informações e o seu envio.

Pesquisar sobre como criar uma aplicação web que permita que os dados e arquivos sejam enviados eletronicamente e armazenados em banco de dados através do sistema de gerenciamento de banco de dados postgreSQL.

Projetar o protótipo do sistema web de forma em que seja possível efetivar buscas e listagens das informações armazenadas.

3. DESENVOLVIMENTO

Após a aplicação ser analisada, pode ser então definido a forma para a criação do sistema. Designando-se então as linguagens e recursos necessários para o seu desenvolvimento.

3.1 TECNOLOGIAS USADAS

Para o desenvolvimento da aplicação proposta, foram estudadas algumas linguagens. Linguagens essas bases para o entendimento da biblioteca JavaScript jQuery UI, como HTML, CSS, JavaScript e jQuery, assim como a própria biblioteca jQuery UI e a linguagem de programação PHP. Também sendo explanados sobre servidores web e banco de dados PostgreSQL.

3.1.1 Protocolo HTTP

Gouley e Totty (2002) definem o *Hypertext Transfer Protocol* (HTTP) como um processo de comunicação utilizada para troca de dados entre o navegador e o servidor web. Este protocolo é utilizado quando algo é digitado na barra de endereços do navegador, seja ela uma aplicação ou um site, a transferência da página até a plataforma do cliente e suas respostas ao servidor web são realizadas com este protocolo. A comunicação HTTP entre o cliente e servidor é feita através de mensagens. O cliente envia mensagens de requisição e o servidor envia uma mensagem de resposta para o cliente.

O protocolo HTTP é baseado em requisições e respostas entre clientes e servidores. O cliente, navegador ou dispositivo que fará a requisição, solicita um determinado recurso, enviando um pacote de informações contendo alguns cabeçalhos um URI ou, mais especificamente, URL(Localizador de Recursos Universal e como o próprio nome diz se refere ao local, o Host que você quer acessar determinado recurso, a URL tem por objetivo associar um endereço remoto com um nome de recurso na Internet.). O servidor recebe estas informações e envia uma resposta, que pode ser um recurso ou um simplesmente outro cabeçalho.

3.1.1.1 Métodos HTTP

Quando serão requisitados, é preciso que seja especificado qual o método será utilizado. Os métodos HTTP, também conhecidos como verbos, identificam qual a ação que deve ser executada em um determinado recurso. Existem 5 métodos mais utilizados.

- GET

Solicita a representação de um determinado recurso. É definido como um método seguro e não deve ser usado para disparar uma ação (remover um usuário, por exemplo).

- POST

As informações enviadas no corpo (body) da requisição são utilizadas para criar um novo recurso. Também é responsável por fazer processamentos que não são diretamente relacionados a um recurso.

- DELETE

Remove um recurso. Deve retornar o status 204 caso não exista nenhum recurso para a URI especificada.

- PUT

Atualiza um recurso na URI especificada. Caso o recurso não exista, ele pode criar um. A principal diferença entre POST e PUT é que o primeiro pode lidar não somente com recursos, mas pode fazer processamento de informações, por exemplo.

- HEAD

Retorna informações sobre um recurso. Na prática, funciona semelhante ao método GET, mas sem retornar o recurso no corpo da requisição. Também é considerado um método seguro.

3.1.1.2 Status

Toda requisição recebe um código de resposta conhecido como status. Com o status é possível saber se uma operação foi realizada com sucesso (200), se ele foi movida e agora existe em outro lugar (301) ou se não existe mais (404).

Existem muitos status divididos em diversas categorias. Na especificação você pode ver cada um deles com uma descrição bastante detalhada. Segue alguns status mais utilizados:

- 200 OK

A requisição foi bem sucedida.

- 301 Moved Permanently

O recurso foi movido permanentemente para outra URI.

- 302 Found

O recurso foi movido temporariamente para outra URI.

- 304 Not Modified

O recurso não foi alterado.

- 401 Unauthorized

A URI especificada exige autenticação do cliente. O cliente pode tentar fazer novas requisições.

- 403 Forbidden

O servidor entende a requisição, mas se recusa em atendê-la. O cliente não deve tentar fazer uma nova requisição.

- 404 Not Found

O servidor não encontrou nenhuma URI correspondente.

- 405 Method Not Allowed

O método especificado na requisição não é válido na URI. A resposta deve incluir um cabeçalho Allow com uma lista dos métodos aceitos.

- 410 Gone

O recurso solicitado está indisponível mas seu endereço atual não é conhecido.

- 500 Internal Server Error

O servidor não foi capaz de concluir a requisição devido a um erro inesperado.

- 502 Bad Gateway

O servidor, enquanto agindo como proxy ou gateway, recebeu uma resposta inválida do servidor upstream a que fez uma requisição.

- 503 Service Unavailable

O servidor não é capaz de processar a requisição pois está temporariamente indisponível.

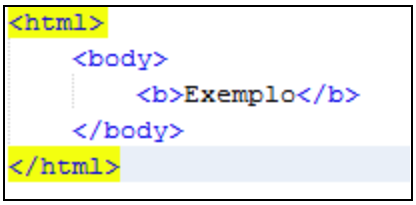
3.1.2 HTML 4

O HTML (*Hypertext Markup Language*) que significa Linguagem de Marcação de Hipertexto é uma linguagem de marcação utilizada para criar páginas web. Estas

páginas são interpretadas pelo navegador e as marcações do HTML influenciam na maneira como que o um determinado trecho do código será exibido no navegador.

Cada marcação é determinada por uma tag, que é uma estrutura de linguagem que contém uma instrução, e cada tag tem uma função que será empregada de acordo com a sua necessidade de uso. Um exemplo apresentado por Silva(2007) para deixar um trecho em negrito, Figura 1.

Figura 1 - HTML



Fonte: Silva(2007)

Visualmente a palavra “Exemplo” será exibida em negrito.

Para a criação de páginas web em HTML não é necessário o uso de IDE (*Integrated Development Environment*), que é a interface integrada para desenvolvimento, podendo ser criado em um bloco de notas e sendo salvo com a extensão.html, e executado no navegador, o próprio navegador interpreta o código do arquivo, e exibe a página de acordo com as marcações definidas (tags). A exibição do HTML pode sofrer algumas alterações dependendo do navegador, pois algumas tags podem renderizar diferentes de um navegador para outro. E com o auxílio de uma IDEs, que é um ambiente integrado para desenvolvimento de software que ajudam no desenvolvimento, o seu uso pode facilitar, já que deixam campos destacados e até auto-completam códigos, entre outras funções que podem ajudar o desenvolvedor.

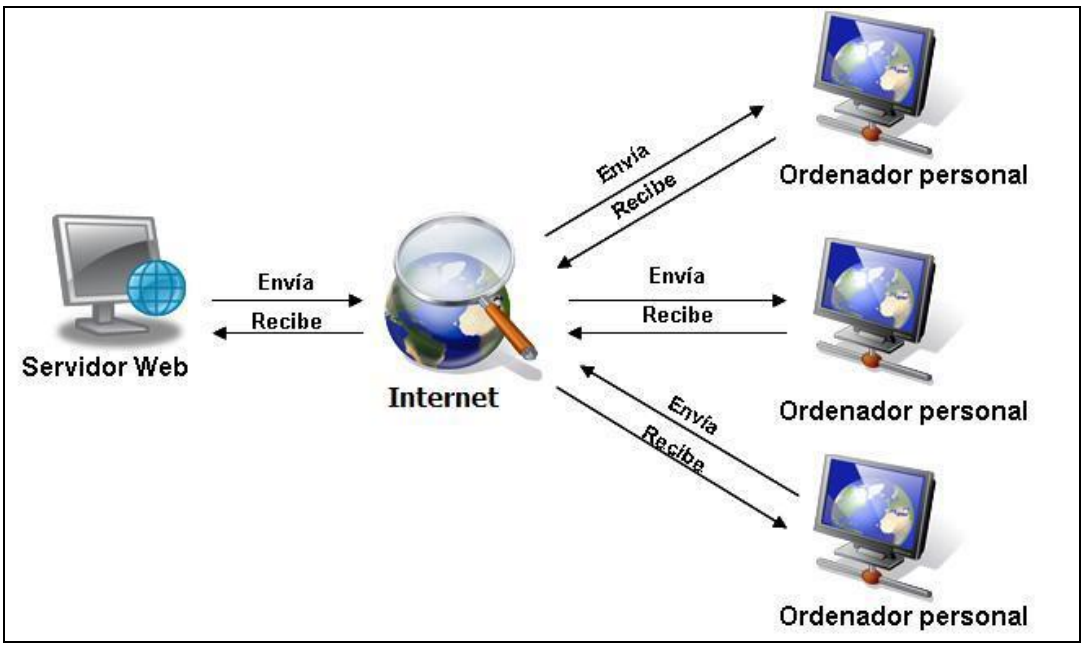
3.1.3 Servidor WEB

A função do servidor web é receber uma requisição e devolver a resposta para o cliente. Na Figura 2, é possível entender como funciona, de forma visual, as requisições que são feitas ao servidor.

Uma solicitação ocorre quando o usuário faz uma solicitação HTTP e o servidor web devolve uma resposta HTTP, sendo que o navegador verifica como

tratar esse conteúdo. Se a resposta que vem do servidor for uma página HTML, então é inserido na resposta HTTP.

Figura 2 - Servidor web



FONTE: NOVAABA[1]

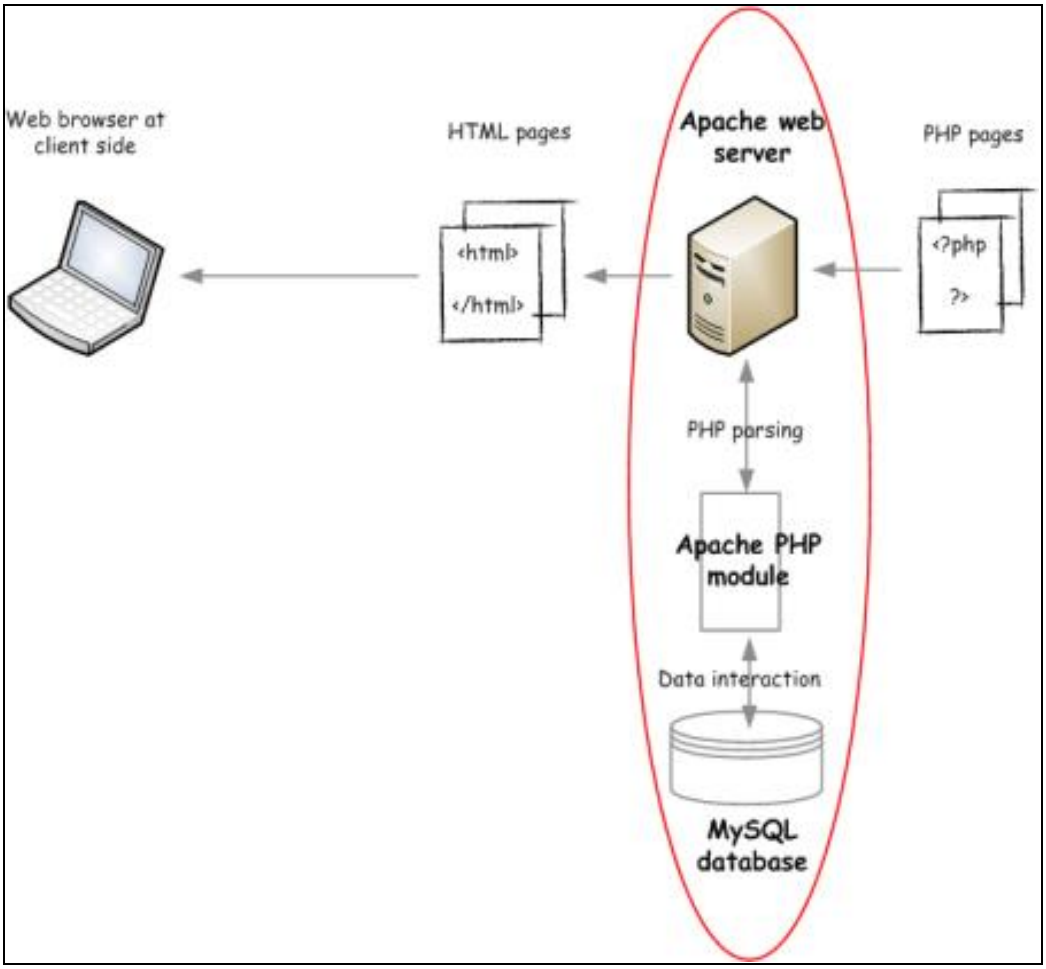
3.1.3.1 APACHE

O Apache é um servidor HTTP livre que trabalha na estrutura cliente-servidor, esse servidor web Apache, recebe as requisições do cliente e responde ao cliente em codificação HTML. O servidor web interpreta a codificação HTML e não interpreta o código PHP.

Ao receber uma solicitação .php o servidor aciona o Interpretador PHP que processa as solicitações do código PHP tais como, acessar banco de dados, sistema de arquivos, acesso ao servidor de correio eletrônico, entre outras tarefas, e retorna para o Apache em formato HTML e ele manda para o browser. O browser lê o código HTML e monta a página web para o usuário. Como pode ser observado na Figura 3 a seguir.

[1] Disponível em: <http://www.novaaba.com.br/category/servidores/>. Acesso em 15 de Nov. de 2016.

Figura 3 - Apache web Server



FONTE: SOFTWARELIVRE[2]

3.1.4 Banco De Dados

Um banco de dados é uma estrutura onde é feito o armazenamento de maneira estruturada e com a menor redundância possível. De acordo com Elmarsi e Navathe (2010, p 23) “Um banco de dados é uma coleção logicamente coerente de dados com inerente. Uma variedade aleatória de dados não pode ser corretamente chamado de banco de dados”, ou seja, os dados para serem considerados como um banco de dados precisa ter algum tipo de ligação.

Estes dados devem ser mantidos seguros, íntegros e disponíveis, para que os usuários que tenham acesso ao banco possam fazer o seu uso.

[2] Disponível em <http://softwarelivre.org/php/servidor-web-apache>. Acesso em 11 Jun. de 2017.

3.1.4.1 SGBD

Para que possa haver o controle do banco é necessário um sistema de gestão. O SGBD (sistema de gestão de banco de dados) é um conjunto de serviços que permite o gerenciamento dos bancos de dados. Permitindo acessos de maneiras mais simples, autorizando acessos simultâneos, efetuando manipulações de dados dos bancos e gerenciando permissões de acesso aos dados.

3.1.4.2 PostgreSQL

O PostgreSQL é um SGBD, muito difundido, estável e confiável. De código aberto. Contendo diversos recursos, e com capacidade de suprir a necessidade das mais diversas aplicações (MILANI, 2008). Está no ranking dos SGBD mais usados no mundo, de acordo com pesquisa feita pelo DB-ENGINES, conforme mostra a Figura 4.

Figura 4 - Ranking dos SGBD de acordo com popularidade

Rank			DBMS	Database Model	Score		
Mar 2017	Feb 2017	Mar 2016			Mar 2017	Feb 2017	Mar 2016
1.	1.	1.	Oracle	Relational DBMS	1399.50	-4.33	-72.51
2.	2.	2.	MySQL	Relational DBMS	1376.07	-4.23	+28.36
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1207.49	+4.04	+71.00
4.	4.	↑ 5.	PostgreSQL	Relational DBMS	357.64	+3.96	+58.01
5.	5.	↓ 4.	MongoDB	Document store	326.93	-8.57	+21.60
6.	6.	6.	DB2	Relational DBMS	184.91	-2.99	-3.02
7.	↑ 8.	7.	Microsoft Access	Relational DBMS	132.94	-0.45	-2.09
8.	↓ 7.	8.	Cassandra	Wide column store	129.19	-5.19	-1.14
9.	9.	↑ 10.	SQLite	Relational DBMS	116.19	+0.88	+10.42
10.	10.	↓ 9.	Redis	Key-value store	113.01	-1.03	+6.79

FONTE: DB-ENGINES, 2017[3]

O PostgreSQL é compatível com os maiores e principais sistemas operacionais, possuindo suporte ao SQL (*Structured Query Language*), que é a linguagem de consulta estruturada padrão que os bancos de dados relacionais utilizam. Sendo melhor em vários pontos, tendo suporte ilimitados de linhas,

[3] Disponível em <http://db-engines.com/en/ranking>. Acesso em 20 de Mar. de 2017.

permitindo bancos de dados de até 16 TB, aceitando diversos tipos de consultas e sub-consultas, contando com um bom mecanismo de segurança contra falhas, entre outros pontos (OLIVEIRA, 2006).

3.1.5 PHP 5

O PHP (*PHP: Hypertext Preprocessor*), antigamente PHP significava “Personal Home Page”, traduzido como Página Web Pessoal, mas hoje em dia significa Preprocessador de Hipertexto PHP, que é uma linguagem de programação interpretada, livre e utilizada para a criação de conteúdos dinâmicos. Estas páginas são interpretadas pelo servidor.

> O PHP como é conhecido hoje, é na verdade o sucessor para um produto chamado PHP/FI. Criado em 1994 por Rasmus Lerdof, a primeira encarnação do PHP foi um simples conjunto de binários Common Gateway Interface (CGI) escrito em linguagem de programação C. Originalmente usado para acompanhamento de visitas para seu currículo online, ele nomeou o conjunto de scripts de "Personal Home Page Tools" mais freqüentemente referenciado como "PHP Tools." Ao longo do tempo, mais funcionalidades foram desejadas, e Rasmus reescreveu o PHP Tools, produzindo uma maior e rica implementação. Este novo modelo foi capaz de interações com Banco de Dados e mais, fornecendo uma estrutura no qual os usuários poderiam desenvolver simples e dinâmicas aplicações web, como um livros de visitas. Em Junho de 1995, Rasmus, liberou o código fonte do PHP Tools para o público, o que permitiu que desenvolvedores usassem da forma como desejassem. Isso permitiu que usuários a fornecerem correções para bugs no código, e em geral, aperfeiçoá-lo.
>
> Em Setembro do mesmo ano, Rasmus expandiu o PHP e - por um breve período - mudou o nome PHP. Agora referindo-se a ferramenta como FI, abreviação para "Forms Interpreter", a nova implementação incluiu algumas funcionalidades básicas do PHP como bem conhecemos hoje. Tinha variáveis no estilo Perl, interpretação automática de variáveis de formulários, e sintaxe HTML embutida. A sintaxe em si era muito similar com a do Perl, porém muito mais limitada, simples, e um pouco inconsistente. De fato, para embutir o código em um arquivo HTML, desenvolvedores tinham que usar comentários HTML. Embora este método não sido inteiramente bem-recebido, FI continuou a desfrutar um crescimento e aceitação como uma ferramente CGI --- mas ainda não como uma linguagem. Contudo, isso começou a mudar no mês seguinte; em Outubro, 1995 Rasmus liberou um completa reescrita do código. Trazendo de volta o nome PHP, estava agora

(brevemente) nomeado "Personal Home Page Contruction Kit" e foi o primeiro lançamento a vangloriar-se que era, na época, considerado um avançado script de interface. A linguagem foi desenvolvida para, deliberadamente, ser parecida com C, tornando-a fácil para ser adotada por desenvolvedores habituados com C, Perl e linguagens similares. Tendo sido até este momento exclusiva para sistemas UNIX e sistemas compatíveis com POSIX, o potencial para uma implementação em um Windows NT começava a ser explorada.

O código tem outra reforma completa, e em Abril de 1996, combinando os nomes dos últimos lançamentos, Rasmus introduziu o PHP/FI. Esta segunda geração da implementação começou a realmente evoluir o PHP de um conjunto de ferramentas para sua própria linguagem de programação. Ele incluía suporte embutido dos banco de dados DBM, mSQL, e Postgres95, cookies, funções de apoio definidas pelo usuário, e muito mais. Em Junho, PHP/FI ganhou o status de versão 2.0. Um interessante fato sobre isso, porém, é que existia apenas um única completa versão do PHP 2.0. Quando finalmente se tornou um status beta em Novembro, 1997, o mecanismo de análise suvbjacente já estava interiramente reescrito.

Apesar de ter tido um curto período de desenvolvimento, ele continuava defrutar uma crescente popularidade em um ainda jovem mundo web desenvolvimento, Em 1997 e 1998, PHP/FI teve o apoio de milhares de usuários ao redor do mundo. Uma pesquisa Netcraft de Maio de 1998, indicou que cerca de 60.000 domínios relataram ter cabeçalhos contendo "PHP", indicando que o servidor de hospedagem de fato tinha o PHP instalado. Este número pode ser equiparado com aproximadamente 1% de todos os domínios da Internet da época. Apesar destes números impressionantes, o amadurecimento do PHP/FI foi condenado a limitações; enquanto haviam vários contribuintes menores, ainda era desenvolvido principalmente por uma única pessoa.

PHP 3.0 foi a primeira versão que se assemelha com o PHP como existe hoje. PHP/FI se encontrava ainda ineficiente e não tinha recursos que precisava para prover uma aplicação eCommerce que estavam desenvolvendo para um projeto da Universidade, Andi Gutmans e Zeev Suraski de Tel Aviv, Israel, começaram outra completa reescrita do interpretador em 1997. Abordando Rasmus online, eles discutiram vários aspectos para a corrente implementação e redesenvolvimento do PHP. Em um esforço para melhorar a engine e iniciar a construção em cima da base de usuários existentes do PHP/FI, Andi, Rasmus, e Zeev decidiram colaborar no desenvolvimento de uma nova e independente linguagem de programação. Essa nova linguagem foi lançada com um novo nome, que

removeu a impressão do limitado uso pessoal que o nome PHP/FI 2.0 tinha mantido. Foi renomeado simplesmente para 'PHP', com o significado se tornando um acrônimo recursivo - PHP: Hypertext Preprocessor.

Um dos maiores pontos fortes do PHP 3.0 foram os fortes recursos de extensibilidade. Além de fornecer a usuários finais uma interface robusta para múltiplos banco de dados, protocolos, e APIs, a facilidade de estender a sua própria linguagem atraiu dezenas de desenvolvedores que submeteram uma variedade de módulos. Indiscutivelmente esta foi a chave para o PHP 3.0 ter sido um tremendo sucesso. Outro recurso chave foi introduzido no PHP 3.0 incluindo o suporte a programação orientada a objeto e a uma mais poderosa e consistente sintaxe de linguagem.

Em junho de 1998, com muitos novos desenvolvedores ao redor do mundo unindo esforços, PHP 3.0 foi anunciado pelo novo time de desenvolvimento do PHP como o sucessor oficial para o PHP/FI 2.0. As melhorias no PHP/FI 2.0, cessaram em Novembro do ano anterior e agora foi oficialmente finalizado. Depois de nove meses de testes abertos ao público, quando o anúncio do lançamento oficial do PHP 3.0 chegou, prontamente foi instalado em 70.000 domínios em todo mundo, e já não era mais limitado ao sistemas operacionais compatíveis ao POSIX. Uma parcela relativamente pequena de domínios informaram que o PHP foi instalado em um host com servidores executando Windows 95, 98 e NT, Macintosh. E em seu pico, PHP 3.0 foi instalado em aproximadamente 10% dos servidores web da internet.

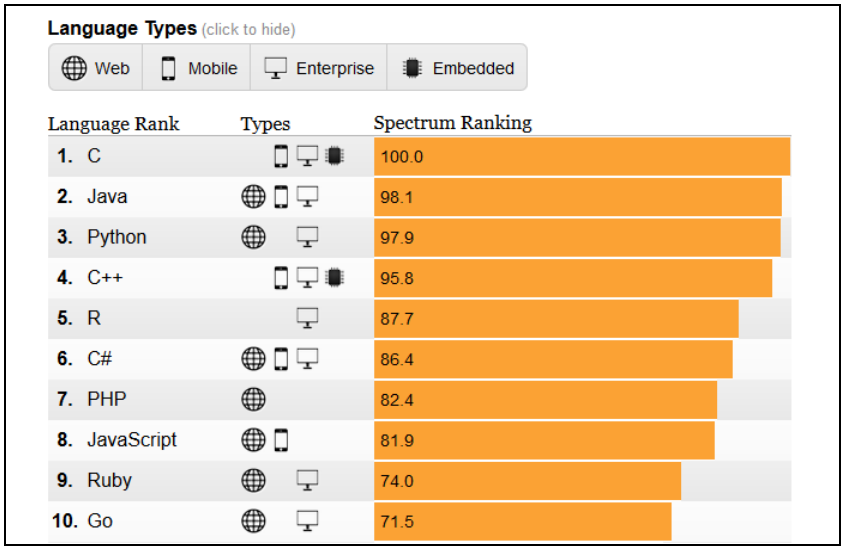
No inverno de 1998, logo após o PHP 3.0 ter sido oficialmente lançado, Andi Gutmans e Zeev Suraski começaram a trabalhar em uma reescrita do core do PHP. Os objetivos do projeto eram melhorar performance das aplicações complexas, e melhorar a modularização do código base do PHP. Tais aplicações só foram possíveis pelos novos recursos e suporte para uma ampla variedades de banco de dados de terceiros e APIs do PHP 3.0, mas o PHP 3.0 não foi projetado para trabalhar com aplicações complexas de forma eficiente.

O novo motor, chamado 'Zend Engine' (composto pelos primeiros nome, Zeev e Andi), alcançou os objetivos do projeto com sucesso, e foi introduzido em meados de 1999. O PHP 4.0 baseado neste motor, e uma variedade de novos recursos adicionais, foi oficialmente lançado em Maio de 2000, quase dois anos após seu antecessor. Além da altíssima melhoria da performance nesta versão, o PHP 4.0 incluiu outros recursos chaves, tais como suporte para maioria dos servidores web, sessões HTTP, saídas de buffering, mais maneiras seguras para manipular dados de entrada de usuários e diversas novas construções de linguagem.

> O PHP 5 foi lançado em Julho de 2004 após um longo desenvolvimento e vários pré-lançamentos. Principalmente impulsionado pelo seu core o Zend Engine 2.0 com um novo modelo de objeto e dezenas de outros novos recursos.[4]

Mesmo com o surgimento de outras linguagens de programação, o PHP vem se mantendo entre as linguagens mais utilizadas, como pode ser observado na Figura 5.

Figura 5 - Ranking Linguagens de Programação IEEE 2016



FONTE: IEEE[5]

O PHP pode ser embutido no código HTML, e sendo interpretado de acordo como as páginas vão sendo apresentadas aos usuários, e facilitando também a conexão com o banco de dados no lado servidor (CONVERSE; PARK, 2003).

A maior parte das realizações feitas com PHP, não é visível ao usuário final. O que é visual é a parte em HTML.

O principal uso do PHP é o desenvolvimento web. Sua sintaxe é semelhante ao C e C++, o que torna mais fácil a adaptação dos programadores a esta linguagem. Criando aplicações com conteúdos dinâmicos para a página web,

---

[4] Disponível em < https://secure.php.net/manual/pt_BR/history.php.php >Acesso em 29 jun. 17.

[5] Disponível em:<http://spectrum.ieee.org/static/interactive-the-top-programming-languages-2016>. Acesso em: 31 ago. 2016.

desenvolvendo rotinas, resolvendo problemas, criando regras de negócio mais complexas,

A identificação feita pelo servidor para detectar o inicio do código PHP que está embutido no HTML é procurando pelo conjunto de caracteres: <?php ?>, como Figura 6 demonstra.

Figura 6 - PHP

```
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        //código php
        ?>
    </body>
</html>
```

Fonte: Do Autor.

Qualquer comando que estiver entre essas tags será interpretado pelo servidor e seu código não será enviado para o cliente. “Os arquivos que contenham instruções PHP, devem possuir em sua extensão a sigla.php” (WELLING; THOMSON, 2003).

A manipulação de arquivos com a linguagem possibilita a criação de sistemas dinâmicos para que o usuário possa fazer diversas funções, como, alterar, excluir, realizar *upload* ou *download* de qualquer arquivo através da aplicação.

3.1.6 Estilos, Funções e Visual

Para que a aparência e usabilidade da aplicação fiquem melhores e mais apresentáveis, são necessários implementar alguns estilos e funções, em seguida serão apresentados alguns paradigmas utilizados para que o sistema opere de forma mais apresentável, e com funções mais simplificadas.

3.1.6.1 CSS 3

Conforme que o HTML foi se popularizando e evoluindo, foram incluídas em suas características, o domínio de controlar algumas aparências para o documento. Håkon Wium Lie, criou jeito mais fácil para formatar a informação. E em 1992 propôs a criação do CSS ou Cascading Style Sheets. Juntamente com Bert Bos começaram a trabalhar no projeto do desenvolvimento do CSS. Em 1995 eles apresentaram sua proposta e o W3C – World Wide Web Consortium, se interessou pelo projeto e resolveu criar uma equipe. O resultado apareceu logo, em 1996, eles lançaram a recomendação oficial pelo W3C do CSS Level 1 (CSS 1). Dois anos depois, no dia 12 de Maio de 1998, eles lançaram a recomendação do CSS de nível 2. A segunda versão das Folhas de Estilo para web. O nível 3 do CSS é a versão mais atual.

Quando é feito um código em HTML é necessário o uso de tags, que definem como determinado trecho do código irá se apresentar, como será a sua aparência. Antigamente a customização de cada tag era feito no próprio HTML, como a Figura 7 exibe.

Figura 7 - Definir cor a trecho em HTML

```
<html>
    <body>
        <font color="blue"> Exemplo</font>
    </body>
</html>
```

Fonte: Do Autor.

Porém, hoje em dia a estilização diretamente nas tags no HTML é considerada má práticas de desenvolvimento.

Com o CSS foi possível uma maior flexibilidade e controle na especificação de como as características serão exibidas, permitiu um compartilhamento de formato, reutilização de estilizações e reduziu a repetição no conteúdo estrutural de uma página.

Surgiu o CSS (*Cascading Style Sheets*) que é uma linguagem de folhas de estilo e tem por objetivo tratar da estilização da página, como cor, tamanho, fonte, dimensões, entre outros estilos.

Para exemplificar, deve ser criado em um arquivo separado do HTML, denominado extensão.css e dentro deste arquivo defini-se um padrão para uma tag, a Figura 8 exemplifica.

Figura 8 - CSS

```
p
{
font-family:Arial;
font-size:14px;
color: blue;
}
```

Fonte: Do Autor.

Que define que onde a tag <p> for usada, o texto será da com a fonte arial, tamanho 14 e a cor azul.

O arquivo HTML então deve fazer um link para poder ter acesso a este arquivo CSS da seguinte forma:

<link rel=”stylesheet” href”estilo.css”>

Após os arquivos estarem interligados, onde o arquivo HTML fazer o uso da tag <p> o texto será mostrado em azul, conforme determinado no arquivo CSS (SILVA, 2007).

Assim como o HTML, o CSS não necessita de IDE para seu desenvolvimento, podendo ser criado em um bloco de notas e sendo salvo com a extensão.css, e do mesmo modo, é possível fazer o uso de IDE para facilitar o desenvolvimento.

3.1.6.2 JavaScript

O HTML não possui funcionalidades que permitam interatividades avançadas às páginas, ficando a cargo das linguagens de programação a tarefa de efetuar funções. Com o HTML é possível criar um formulário com rótulos e campos para serem preenchidos pelo usuário, mas esse não é capaz de fazer o processamento das informações, nem enviá-los a outra máquina ou servidor (SILVA, 2010).

O JavaScript foi criado com a finalidade de fornecer um meio de adicionar interatividade a uma página web e que roda do lado cliente, ou seja, a interpretação

e funcionamento dependem das funcionalidades do navegador do usuário. O autor (SILVA, 2010) ainda complementa que com o JavaScript pode-se controlar o comportamento dos navegadores, gerarem janelas de pop-up, apresentar mensagens ao usuário, redimensionar a janela do navegador, entre outras coisas.

Seguindo os padrões Web de desenvolvimento, uma página web deve ser dividida em três camadas, a estruturação do conteúdo em HTML, o estilo e apresentação em CSS, e os scripts de comportamento em JavaScript. Além de tornar os códigos mais organizados, implica na facilidade para reaproveitamento de códigos. É possível colocar a função dentro do HTML, mas também se caracteriza como má prática de desenvolvimento.

Para exemplificar, deve ser criado em um arquivo separado do HTML e CSS, fazendo-se a criação de um arquivo, por exemplo, denominado script.js e determinar as funções dentro deste arquivo. Muitas funções são pré definidas como, por exemplo, o alert(), que abre uma janela contendo informações, as quais o desenvolvedor definir, basta o desenvolvedor modelar de acordo com a necessidade. O arquivo HTML então deve fazer um link para poder ter acesso a este arquivo JS da seguinte forma:

<script type=”text/javascript” src=”script.js”></script>

Após os arquivos estarem interligados, o arquivo HTML pode fazer o uso das funções estabelecidas no arquivo JS.

Assim como o HTML e CSS, o JavaScript não necessita de IDE para seu desenvolvimento, podendo ser criado em um bloco de notas e sendo salvo com a extensão.js, como nas linguagens anteriores, é possível fazer o uso de IDE para facilitar o desenvolvimento.

3.1.6.3 jQuery

O jQuery é uma biblioteca JavaScript, com o lema “*write less, do more*.”, que significa, escrever menos e fazer mais. “O foco principal da biblioteca jQuery é a simplicidade. Por que submeter os desenvolvedores ao martírio de escrever longos e complexos códigos para criar simples efeitos?” (RESIG, *apud* SILVA, 2010, p. 23). Segundo o autor o jQuery torna fácil a escrita de códigos em JavaScript, tornando a

criação de códigos mais simples, e que até mesmo alguém com pouca experiência, consiga fazer o seu uso sem maiores dificuldades.

De acordo com Silva (2010, p. 24) “Simplicidade é a palavra-chave que resume e norteia o desenvolvimento com jQuery”, pois, com poucas linhas de código em jQuery, é possível criar uma tarefa que em JavaScript seriam necessárias diversas linhas escritas, e este mesmo código em JavaScript, além de longo, poderia ser de complexo entendimento.

Com o uso do jQuery é possível criar páginas web dinâmicas e interativas, como por exemplo, implementando animações, alterando conteúdos, entre outros, e assim, possibilitando a facilidade em implementar usabilidade, acessibilidade e design ao sistema web, tornando a experiência do usuário mais rica.

A biblioteca jQuery é um arquivo Javascript, e o arquivo deve ter a extensão .js, e também deve ser referenciado na página que fará o seu uso, do mesmo jeito como é feito com o JavaScript. Deve ser feito o download do arquivo jQuery direto do site do fornecedor, *jquery.com*, recomenda-se sempre usar a versão mais atual da biblioteca. Dentro deste pacote estão as diversas funções habilitadas da biblioteca, mas é possível implementar de forma manual alguma função que não exista e seja necessária para o desenvolvedor.

3.1.6.4 jQuery UI

A biblioteca jQuery UI foi criada baseada em jQuery, seguindo os mesmos princípios do jQuery, e serve para a implementação de elementos visuais ricos a experiência do usuário. Como Silva (2012) descreve, possui diversas funções que tornam a aplicação com uma aparência melhor, e mais fácil de usar. A seguir podem-se observar através de uma tabela 1 os componentes existentes nesta biblioteca.

Tabela 1– Classificação e finalidade dos elementos de interface

Grupo	Nome	Descrição
	Draggable (Arrastar)	Cria elementos capazes de serem arrastados pela interface
	Droppable (Soltar)	Cria elementos para receber os elementos arrastáveis

Componentes	Resizable (Dimensionar)	Cria elementos cujas dimensões podem ser controladas pelo usuário.
	Selectable (Selecionar)	Cria elementos que possam ser selecionados pelo usuário, quer individualmente quer em grupos.
	Sortable (Ordenar)	Cria elementos que possam ser ordenados pelo usuário com ação de arrastar e soltar.
Widgets	Accordion (Acordeão)	Cria o efeito acordeão, para ocultar/mostrar conteúdos.
	Autocomplete (Autocompletamento)	Apresenta ao usuário uma lista de sugestões de palavras à medida que ele digita em um campo de texto.
	Button (Botão)	Cria de vários tipos de botão
	Datepicker (Seletor de datas)	Cria uma janela pop-up para seleção da data a ser digitada em um campo destinado a coletar uma data.
	Dialog (Janela de diálogo)	Cria vários tipos de janela de diálogo, entre elas janelas modais.
	Progressbar (Barra de progresso)	Cria uma barra indicativa do andamento de uma tarefa.
	Slider (Controle deslizante)	Cria um botão arrastável em uma guia para seleção de um valor compreendido em determinada faixa.
	Tabs (Abas)	Cria uma interface cuja navegação é feita com o uso de abas.
	Color Animation	Esse efeito destina-se a animar as cores de um elemento.
	Toggle class	Esses efeitos destinam-se a animar elementos baseados na manipulação dos seus atributos classe.
	Add class	
	Remove class	
	Switch class	
	Effect	Aplica em um elemento uma série de

Efeitos		efeitos de animação padrão da biblioteca jQuery, tais como os efeitos de esmaecimento, pulsação, balanço, sacudir etc.
	Toggle	Aplica um dos efeitos padrão da biblioteca jQuery com a finalidade de alternar entre ocultar e esconder um conteúdo.
	Hide	Efeito destinado a ocultar um conteúdo.
	Show	Efeito destinado a mostrar um conteúdo.
Utilidades	Position	Destina-se a controlar e manipular o posicionamento dos elementos de interface.
	Widgets	Destina-se à criação de widgets personalizados.

FONTE: SILVA, 2012, p. 24

Quando o desenvolvedor for fazer o *download* da biblioteca, deve escolher quais os componentes que serão úteis na sua aplicação, personalizando de acordo com a necessidade do seu projeto. A biblioteca é em módulos, que só deverá ser baixado o que for necessário, e tem por objetivo tornar mais rápido o carregamento das páginas (SILVA, 2012). É possível simular e modelar os temas na página do jQuery UI[6], onde é possível a criação de um tema personalizado, ou fazer o uso de algum dos temas prontos, quem além de permitir que o desenvolvedor possa ver como ficará a forma visual a aplicação, gera o arquivo de download de acordo com o tema que foi definido.

Para ser usado, o jQuery UI segue os mesmo princípios do jQuery, tanto que para chamar uma função, deve ser escrita da mesma forma que seria com jQuery. Porém esta biblioteca necessita de mais alguns componentes além dos arquivos JavaScript, como os widgets, CSS, entre outros. Que são artefatos necessários para seja possível o uso do tema qual foi previamente escolhido.

[6] Disponível em <http://jqueryui.com/themeroller>. Acesso em 27 de Mar. de 2017.

3.1.7 XAMPP

O XAMPP é um pacote de aplicativos que facilita a configuração e configuração de plataforma. É uma ferramenta de desenvolvimento, que permite aos programadores testar seus trabalhos em seu próprio computador, sem a necessidade de acesso à Internet. Possui vários aplicativos como: Apache, MySQL, phpMyAdmin, FileZilla FTP Server, OpenSSL. Como a sua própria inicial já significa:

- X- para diferentes sistemas operacionais
- A - Apache- servidor web
- M- MySQL- base de dados
- P- PHP- interpretador para linguagens de script
- P- Perl- interpretador para linguagens de script

A principal função da qual será feito o uso, nesta aplicação, do XAMMP é o Apache, para simular um servidor web local, permitindo testar as páginas que fizerem o uso de PHP.

3.2 RECURSOS PARA ESPECIFICAÇÃO

Para a modelagem do sistema, necessita-se de uma boa especificação e descrição da aplicação. Existem ferramentas capazes de modelar o sistema de formas mais visuais.

3.2.1 Modelagem Relacional

O conceito principal vem da teoria de conjuntos atrelado a ideia de que não é relevante ao usuário saber onde os dados estão ou como eles se encontram, representado por uma coleção de tabelas, é um conjunto de linhas uma lista de valores de atributos.

3.2.1.1 Modelo Entidade Relacionamento

O Modelo entidade relacionamento proposto por Peter P. Chen são as relações entre as coisas. Como as mesmas irão relaciona-se entre elementos

individualizados de diferentes conjuntos ou entre elementos de um mesmo conjunto. A forma de comunicação entre as coisas ou um conjunto delas.

Entidade é a representação genérica de um componente do mundo real, sobre o qual desejamos armazenar informações, uma representação de quase todas as informações com varias propriedades que devem ser compreendidas pelo sistema de informação, qualquer coisa que produza ou consuma informações. Entidade são coisas significativas sobre a qual a organização deseja guarda, ou seja, dados podendo ser algo tangível ou intangível.

Atributo é tudo o que se pode relacionar como próprio da entidade que de alguma maneira a qualifique e a distinga de outras.

Relacionamento é a relação existente entre entidades, isto é, a ligação lógica entre duas entidades que representa uma regra ou restrição de negócio, possibilitando entender como uma entidade se comporta em relação às demais, qual o seu grau de dependência de outras entidades e qual a associação de dados existentes entre elas.

Cardinalidade indica a quantidade de ocorrências de determinado relacionamento, sua representação é variável de acordo com as seguintes notações,

- N : várias vezes
- 1 : apenas uma vez
- 0: não acontece

Os Relacionamentos também devem ser destacados, uma vez que as entidades são identificadas, deve-se então definir como se dá o relacionamento entre elas. De acordo com a quantidade de objetos envolvidos em cada lado do relacionamento, pode-se classificar de três formas:

- Relacionamento 1..1 (um para um): cada uma das duas entidades envolvidas referencia obrigatoriamente apenas uma unidade da outra.
- Relacionamento 1..n ou 1..* (um para muitos): uma das entidades envolvidas pode referenciar várias unidades da outra, porém, do outro lado cada uma das várias unidades referenciadas só pode estar ligada uma unidade da outra entidade.
- Relacionamento n..n ou *..* (muitos para muitos): neste tipo de relacionamento cada entidade, de ambos os lados, podem referenciar múltiplas unidades da outra.

3.3 UML

A UML (*Unified Modeling Language*) Linguagem de Modelagem Unificada é uma ferramenta que auxilia na modelagem de sistemas. Segundo Medeiros (2004) a UML nos auxilia em como representar um sistema em diversos estágios de desenvolvimento, tornando possível visualizar, por meio dos diagramas, como o software deve funcionar.

Existem vários tipos de diagramas na UML, de acordo com Guedes (2007) “A utilização de diversos diagramas permite que falhas possam ser descobertas nos diagramas anteriores, diminuindo a possibilidade da ocorrência de erros durante a fase de desenvolvimento do software”. Ainda segundo o autor, cada diagrama tem a sua utilidade em especial, e não é necessário o uso de todos os diagramas para a modelagem de um sistema, alguns são específicos a determinados tipos de implementações.

3.3.1 Diagrama de Casos de Uso

O diagrama de casos de uso é um dos mais utilizados, segundo Guedes (2007). Sendo muito empregado para o levantamento e análise de requisitos, momento em que são apontadas as necessidades dos usuários, e no entendimento geral do sistema, sendo geralmente usado como base para a criação dos outros diagramas. Diagrama esse que apresenta linguagens de fácil compreensão para que as idéias de comportamento do sistema sejam entendidas pelos usuários.

Para a criação do diagrama, se faz necessário que em primeiro momento seja identificado os Atores, esses são entidades externas, ou seja, pode ser um usuário, um outro sistema que interaja com o sistema que está sendo implementado, algum equipamento que solicite ações. Estes atores utilizarão de alguma funcionalidade do software, tais funções são chamadas de Casos de Uso. Os casos de uso conforme designa Medeiros (2004) pode ser a representação de uma macro atividade, e para que a ação sugerida seja efetivada, necessita de diversas outras atividades que não sejam descritas no diagrama, como por exemplo, um caso de uso para que seja efetuado o pagamento, será validado o cartão, informado os valores, entre outras informações. Só que no diagrama não fica explícito cada atividade que o caso de uso vai fazer, se tornando desse modo uma macro atividade que engloba todas as

atividades que se fará necessário, ao ser chamado, por exemplo, Efetuar pagamento.

Possuem também tipos de iterações, que são as ligações entre um caso de uso e outro. Tem a Inclusão, que é quando os casos de uso estiverem interligados, quando o primeiro for chamado, o caso de uso que estiver ligado por inclusão, automaticamente executará a sua função. E tem também a extensão, que é quando os casos de uso estiverem interligados, quando o primeiro for chamado, o caso de uso que estiver ligado por extensão, é opcional a sua execução.

3.4 ESTUDO DE CASO

Para realizar a análise das necessidades que o sistema necessita cumprir, foram observados e analisados os documentos fornecidos pelo IFSUL e pelo professor responsável (documentos disponíveis em lista de ANEXOS). A partir deles foi percebido a necessidade de que a aplicação realize o armazenamento das informações que se farão necessárias para os cadastros dos docentes e das atividades que os mesmo realizarão. Tais como: classe (nível em qual o docente encontra-se em relação ao plano de carreira); campus (onde o docente está vinculado); titulação (nível acadêmico do docente); regime (de horas de contrato); professor (os dados próprios do docente); avaliador (os dados do responsável pelas avaliações); item (o título principal de cada atividade); atividade (a atividade única a ser exercida); avaliação (conjunto da avaliação que fica dentro dos 24 meses para a progressão); item avaliação (atividade exercida individualmente e que já pode ser avaliada).

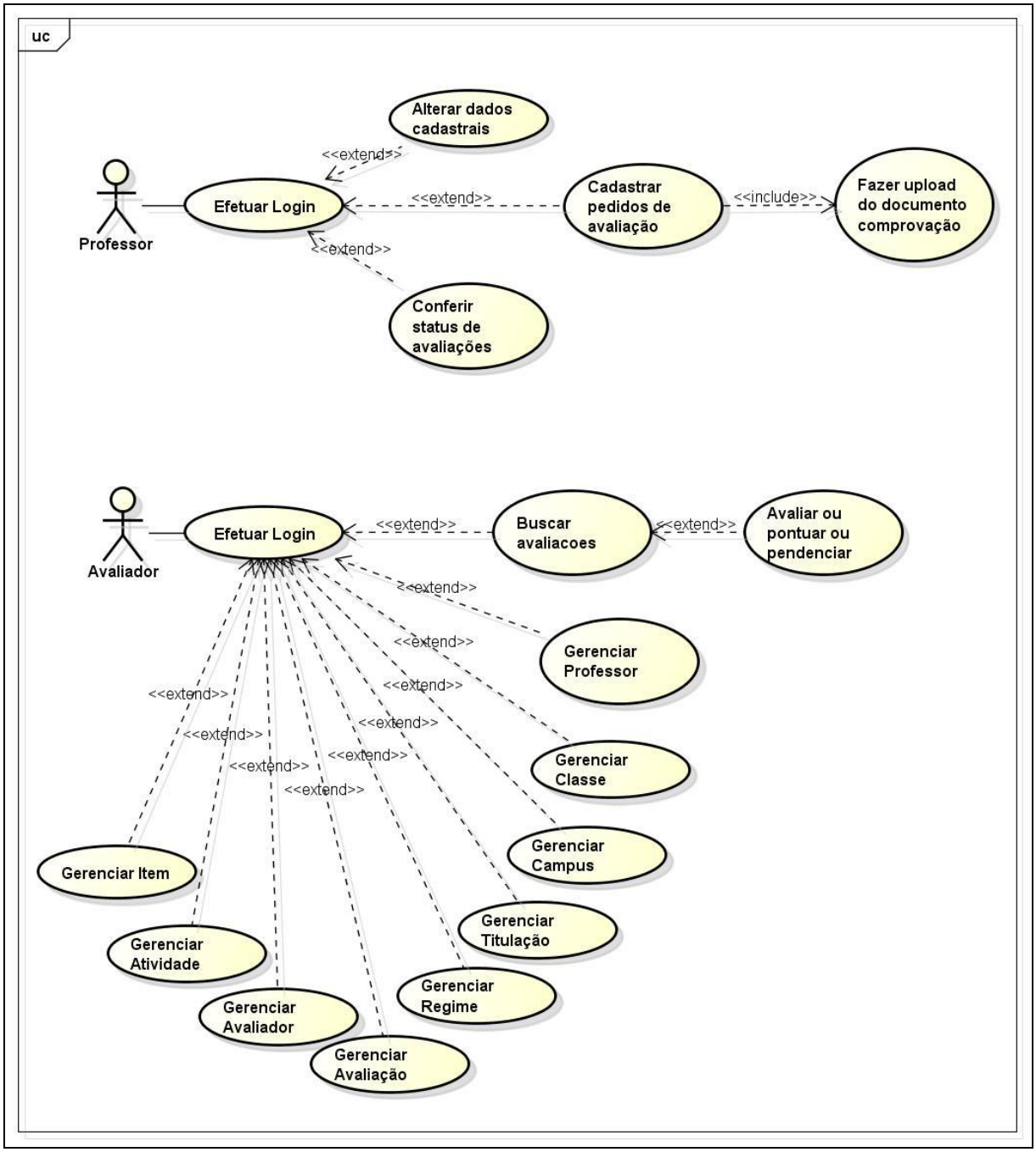
3.4.1 Diagrama de Casos de Uso

Com a modelagem do diagrama de casos de uso, se torna mais visual a as funcionalidades necessárias da aplicação.

No caso descrito na Figura 9, o ator Professor terá de efetivar o *login* para ter acesso aos dados cadastrais, cadastro de pedidos de avaliação com upload de documentos e conferência de status de avaliações. O ator Avaliador terá de efetivar o *login* para poder ter acesso ao gerenciamento de Classe, Campus, Titulação,

Regime, Professor, Item, Atividade, Avaliador e Avaliação, busca de avaliações e pontuação de avaliações.

Figura 9- Diagrama de Casos de Uso



Fonte: Do Autor.

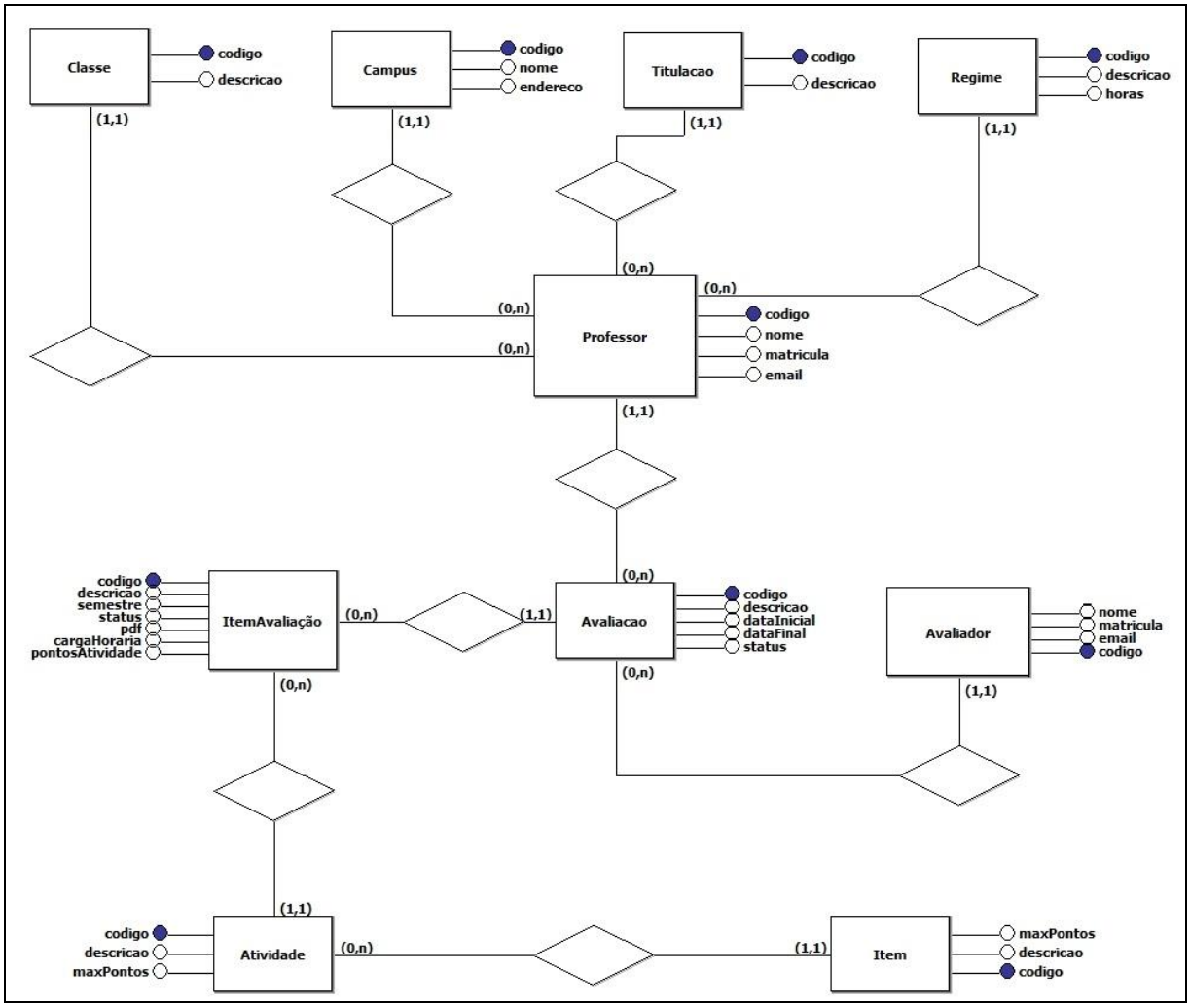
3.4.2 Modelo Entidade Relacionamento

Com base no levantamento de requisitos, tornou-se possível a identificação das partes e das suas principais funções. A partir deste estudo foi possível construir um modelo entidade relacionamento, que descreve as entidades com seus atributos e a maneira que se relacionam. O modelo representa a estrutura que serviu de base para o banco de dados da aplicação.

As entidades, como podem ser observadas na Figura 10, foram criadas da seguinte forma:

- **Classe** – com os atributos: código (chave primária) e descrição;
- **Campus** – com os atributos: código (chave primária), nome e descrição;
- **Titulação** – com os atributos: código (chave primária) e descrição;
- **Regime** – com os atributos: código (chave primária), descrição e horas;
- **Item**– com os atributos: código (chave primária), descrição e maxPontos;
- **Avaliador** – com os atributos: código (chave primária), nome, matricula, email;
- **Professor** – com os atributos: código (chave primária), nome, matricula, email, classe (chave estrangeira, de Classe), campus (chave estrangeira, de Campus), titulação (chave estrangeira, de Titulação) e regime (chave estrangeira, de Regime);
- **Avaliacao** – com os atributos: código (chave primária), descrição, dataInicio, dataFinal, professor (chave estrangeira, de Professor) e avaliador (chave estrangeira, de Avaliador);
- **Atividade** – com os atributos: código (chave primária), descrição, maxPontos e item (chave estrangeira, de Item);
- **ItemAvaliacao** – com os atributos: código (chave primária), descrição, semestre, status, pdf, cargaHoraria, pontosAtividades, avaliacao (chave estrangeira, de Avaliacao) e atividades (chave estrangeira, de Atividade).

Figura 10 - Modelo Entidade Relacionamento



Fonte: Do Autor.

3.4.3 Modelagem Do Bando De Dados

Após a modelagem de entidade relacionamento ter sido findada, a modelagem de banco de dados foi desenvolvida com maior facilidade por meio do uso do pgAdmin, que é o SGBD usado, o banco de dados foi com as sqls a seguir:

- **Criação da tabela Classe:**

  create table classe (
  codigo serial not null primary key,
  descricao varchar(10) not null);

- **Criação da tabela Campus:**

```
create table campus (
codigo serial not null primary key,
nome varchar(50) not null,
endereco varchar(100) not null );
```

- **Criação da tabela Titulacao:**

```
create table titulacao (
codigo serial not null primary key,
descricao varchar(10) not null );
```

- **Criação da tabela Regime:**

```
create table regime (
codigo serial not null primary key,
descricao varchar(20) not null,
horas int not null );
```

- **Criação da tabela Professor:**

```
create table professor (
codigo serial not null primary key,
nome varchar(50) not null,
matricula varchar(20) not null,
email varchar(50) not null,
classe int not null,
campus int not null,
titulacao int not null,
regime int not null,
foreign key (classe) references classe(codigo),
foreign key (campus) references campus(codigo),
foreign key (titulacao) references titulacao(codigo),
foreign key (regime) references regime(codigo) );
```

- **Criação da tabela Avaliador:**

```
create table avaliador (
codigo serial not null primary key,
```

```
nome varchar(50) not null,
matricula varchar(20) not null,
email varchar(50) not null);
```

- **Criação da tabela Avaliacao:**

```
create table avaliacao (
codigo serial not null primary key,
descricao varchar(70) not null,
dataInicio date not null,
dataFinal date not null,
status varchar(50) not null,
professor int not null,
avaliador int not null,
foreign key (professor) references professor(codigo),
foreign key (avaliador) references avaliador(codigo) );
```

- **Criação da tabela Item:**

```
create table item (
codigo serial not null primary key,
descricao varchar(150) not null,
maxpontos int not null );
```

- **Criação da tabela Atividade:**

```
create table atividade (
codigo serial not null primary key,
descricao varchar(150) not null,
maxpontos int not null,
item int not null,
foreign key (item) references item(codigo) );
```

- **Criação da tabela ItemAvaliacao:**

```
create table itemAvaliacao (
codigo serial not null primary key,
descricao varchar(200) not null,
```

```
semestre varchar(15)  not null,
cargahoraria int not null,
pontosatividade int not null,
status varchar(20) not null,
pdf varchar(10) not null,
atividade int not null,
avaliacao int not null,
foreign key (atividade) references atividade(codigo),
foreign key (avaliacao) references avaliacao(codigo) );
```

3.4.4 Conexão Com Banco De Dados

Para que o sistema PHP se conecte com o banco de dados postgresql é necessário que seja feito uma função de conexão conforme a Figura 11 demonstra, devem ser colocados usuário e senha do banco de dados, o local onde está o banco de dados, o nome do banco que será usado e a porta que o banco de dados usa. A função utiliza métodos prontos para fazer a conexão ou fechar a conexão com postgresql.

Figura 11- Conexão com Banco de Dados

```php
<?php
class Banco {
    private $conexao;
    function abreConexao() {
        $usuario = 'postgres';
        $senha = 'postgres';
        $host = 'localhost';
        $banco = 'CPPD';
        $porta = '5432';
        $this->conexao = pg_connect("
        host = '$host'
        dbname = '$banco'
        user = '$usuario'
        password = '$senha'
        port = $porta
        ") or die("Erro na conexao"); //echo "Conectado";
    }
    function fechaConexao() {
        pg_close($this->conexao); //echo "Desconectado";
    }
    function executaSql($sql) {
        $status = pg_connection_status($this->conexao);
        if ($status != 0)
            die("Conexao caiu");
        pg_send_query($this->conexao, $sql);
        return pg_get_result($this->conexao);
    }
    function retornaErro() {
        return pg_errormessage($this->conexao);
    }
}
?>
```

Fonte: Do Autor.

3.4.5 MVC

Model-view-controller (MVC), em português modelo-visão-controlador, é um padrão de arquitetura de software que separa a representação da informação da interação do usuário. Separa a aplicação em três camadas. O modelo (*model*) consiste nos dados da aplicação, regras de negócios, lógica e funções. Uma visão (*view*) pode ser qualquer saída de representação dos dados, como uma tabela ou um diagrama. É possível ter várias visões do mesmo dado, como um gráfico de barras para gerenciamento e uma visão tabular para contadores. O controlador (*controller*) faz a mediação da entrada, convertendo-a em comandos

para o modelo ou visão. As idéias centrais por trás do MVC são a reusabilidade de código e separação de conceitos.

3.4.5.1 Modelo

É responsável pela leitura e escrita de dados, e também por suas validações. Será usada uma classe com menos atributos para o fim de demonstração do MVC. A Figura 12 mostra a camada modelo se ligando ao arquivo do banco que fez a conexão com o banco de dados, a criação da classe e seus atributos e encapsulamento dos atributos.

Figura 12 - Modelo Parte 1

```php
<?php

include_once '../Banco/Banco.php';

class Classe {

    private $codigo;
    private $descricao;

    public function getCodigo() {
        return $this->codigo;
    }

    public function getDescricao() {
        return $this->descricao;
    }

    public function setCodigo($codigo) {
        $this->codigo = $codigo;
    }

    public function setDescricao($descricao) {
        $this->descricao = $descricao;
    }
```

Fonte: Do Autor.

Nas Figuras 13 e 14 são demonstradas algumas das funções criadas para que os dados possam ser manipulados: buscados, listados, salvos e excluídos do banco de dados.

Figura 13 - Modelo Parte 2

```
    public function buscaClasse($codigo) {
        $sql = "select * from classe where codigo=$codigo";
        $b = new Banco();
        $b->abreConexao();
        $b->executaSql($sql);
        $resultado = $b->executaSql($sql);
        if ($b->retornaErro() != "")
            echo "$b->retornaErro()";
        else {
            $linha = pg_fetch_array($resultado);
            $this->setCodigo($linha['codigo']);
            $this->setDescricao($linha['descricao']);
        }
        $b->fechaConexao();
    }

    public function salvaClasse() {
        if ($this->getCodigo() == "") {
            $sql = "insert into classe(descricao) values ('$this->descricao')";
        } else {
            $sql = "update classe set ";
            $sql = $sql . " descricao = '$this->descricao'";
            $sql = $sql . "where codigo = $this->codigo";
        }
        echo $sql;
        $b = new Banco();
        $b->abreConexao();
        $b->executaSql($sql);
        $erro = $b->retornaErro();
        $b->fechaConexao();
        return $erro;
    }
```

Fonte: Do Autor.

Figura 14 - Modelo Parte 3

```
    function listarClasse($ordem) {
        $sql = "select * from classe order by $ordem";
        $b = new Banco();
        $b->abreConexao();
        $result = $b->executaSql($sql);
        $erro = $b->retornaErro();
        if ($erro != "")
            echo "$erro";
        else {
            while ($linha = pg_fetch_array($result)) {
                $c = new Classe();
                $c->setCodigo($linha['codigo']);
                $c->setDescricao($linha['descricao']);
                $res[] = $c;
            }
            return $res;
        }
    }

    function excluiClasse($codigo) {
        $sql = "delete from classe where codigo = $codigo";
        $b = new Banco();
        $b->abreConexao();
        $b->executaSql($sql);
        $erro = $b->executaSql($sql);
        if ($erro != "")
            echo "$erro";
        else
            echo "Exclusão bem sucedida";
    }
}
?>
```

Fonte: Do Autor.

3.4.1 Controle

É o responsável por receber todas as requisições do usuário, controlando qual modelo usar.

A Figura 15 mostra a ligação entre a camada controle e a camada modelo. Também são criados as funções que serão usadas na camada visão. Por exemplo, se alguém clicar no salvar na camada visão, será acionado a função salvar que foi

descrita na camada controle. Foram implementadas as funções alterar, salvar, excluir e ordenar.

Figura 15 - Controle

```php
<?php
include_once '../Modelo/ModeloClasses.php';
if(@$_REQUEST['al']){
    $c = new Classe();
    $c->buscaClasse($_REQUEST['al']);
    $codigo = $c->getCodigo();
    $descricao = $c->getDescricao();
}
if(@$_POST['salvar']){
    $c = new Classe();
    $c->setCodigo($_POST['codigo']);
    $c->setDescricao($_POST['descricao']);

    $erro = $c->salvaClasse();
    if($erro==""){
    $codigo = "";
    $descricao = "";

    $mostrar = "nao";
    echo "  Salvo com sucesso!";
    } else echo $erro;
}
if(@$_REQUEST['ex']){

    $c = new Classe();
    $c->excluiClasse($_REQUEST['ex']);
}
//ORDENAÇÃO
if(@$_REQUEST['ord'])
    $ordem = $_REQUEST['ord'];
    else
        $ordem="descricao";
$c = new Classe();
$listaclasse = $c->listarClasse($ordem);
?>
```

Fonte: Do Autor.

3.4.2 Visão

É a camada de interação com o usuário, ela faz a exibição dos dados em tela.

Figura 16 - Visão Parte 1

```
OCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitior
ml xmlns="http://www.w3.org/1999/xhtml">
<head>

    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>CPPD</title>
    <link href="../estiloVisao.css" rel="stylesheet" type="text/css" />
    <link rel="stylesheet" type="text/css" href="../Scripts/superfish/superfish.css" />
    <script type="text/javascript" src="../js/jquery-1.9.1.min.js"></script>
    <script type="text/javascript" src="../Scripts/superfish/superfish.js"></script>
    <script type="text/javascript" src="../js/jquery.mousewheel.js"></script>
    <link rel="stylesheet" type="text/css" href="../Scripts/slidedeck/slidedeck.skin.css" media="screen" />

    <script type="text/javascript">
        //Executa a funcao quando o documento estiver carregado
        $(document).ready(function () {
            $('.navigation').superfish();
        });</script>

    <!-- Incluir aqui o SlideDeck jQuery script. -->
    <script type="text/javascript" src="../js/slidedeck.jquery.lite.pack.js"></script>
    <style type="text/css">
        #slidedeck_frame {
            width: 850px;
            height: 300px;
        }
    </style>

</head>
<body>
    <div id="container">
```

Fonte: Do Autor.

Figura 17 - Visão Parte 2

```
            <div id="topo">
                <a href="../index.php"> <img src="../images/cabecalhoIFSUL.jpg"/> </a>
            </div>    <!--fim da div topo -->

            <ul class="navigation">
                <li> <a href="../index.php"> Início </a> </li>
                <li> <a href="VisaoClasse.php">Classe </a></li>
                <li> <a href="VisaoCampus.php">Campus </a></li>
                <li> <a href="VisaoTitulacao.php">Titulação </a></li>
                <li> <a href="VisaoRegime.php">Regime </a></li>
                <li> <a href="VisaoProfessor.php">Professor </a></li>
                <li> <a href="VisaoItem.php">Item </a></li>
                <li> <a href="VisaoAtividade.php">Atividade </a></li>
                <li> <a href="VisaoAvaliador.php">Avaliador </a></li>
                <li> <a href="VisaoAvaliacao.php">Avaliacao </a></li>
                <li> <a href="VisaoItemavaliacao.php">Registros</a></li>
            </ul> <!-- fim da ul class navigation -->

            <div id="conteudo">

                <?php
                include_once '../Controle/ControleClasse.php';
                ?>
                <?php
                if ((@$_REQUEST['mostrar']) && (@$mostrar == ""))
                    $mostrar = $_REQUEST['mostrar'];
                if (@$mostrar == "sim") {
                    ?>
                    <form method="POST" action="">
                        </br>
```

Fonte: Do Autor.

Figura 18 - Visão Parte 3

```
                <table border="0" align="center" >
                    <tr align="left" >
                        <input type="hidden" name="codigo" value="<?= @$codigo ?>">

                            <td>Descrição</td>
                            <td><input type="text" name="descricao" value="<?= @$nome ?>"></td>
                    </tr>
                </table>

                <br><input type="submit" name="salvar" value="Salvar">

                        </form>

                    <?php } ?>
                    <br><br>
                            <a href="?mostrar=sim">Incluir</a>
                            <table border="3" align="center" >
                                <tr>
                                    <td><a href="?ord=descricao"><b> Descricao</b></a></td>
                                    <td><b>Operação</b></td>
                                </tr>
                                <?php
                                foreach ($listaclasse as $c) {
                                    echo "<tr >";
                                    echo "<td> <b>" . $c->getDescricao() . "</b></td>";
                                    echo "<td><a href=?ex=" . $c->getCodigo() . ">Excluir</a>
      <a href=?al=" . $c->getCodigo() . "&mostrar=sim>Alterar</a>
</td>";
                                    echo "</tr>";
                                }
                                ?>
                            </table>
                            </div> <!--fim da div conteudo -->
                            </div>  <!--fim da div container -->
                            </body>
                            </html>
```

Fonte: Do Autor.

Como pode ser observados nas Figuras 16, 17 e 18, é feito toda a parte visual em HTML, e a ligação da camada visão com a camada controle. Na Figura 16 nas primeiras linhas são feitas as conexões e chamando as funções de JavaScript, por exemplo, a linha com o código abaixo, está habilitando a folha de estilo denominada estiloVisão.css, que contém as descrições para as tags, para ser utilizada nesta camada.

<link href="../estiloVisao.css" rel="stylesheet" type="text/css" />

Abaixo seguem também mais alguns trechos do código referentes a utilização do JavaScript, jQuery e jQuery UI, também demonstrada na Figura 16.

<link rel="stylesheet"type="text/css"href="../Scripts/superfish/superfish.css"/>
<script type="text/javascript" src="../js/jquery-1.9.1.min.js"></script>
<script type="text/javascript" src="../Scripts/superfish/superfish.js"></script>
<script type="text/javascript" src="../js/jquery.mousewheel.js"></script>

```
<link rel= "stylesheet"type= "text/css"
href="../Scripts/slidedeck/slidedeck.skin.css" media="screen" />
<script type="text/javascript" src="../js/slidedeck.jquery.lite.pack.js"></script>
```

Pode ser observado que em alguns momentos, quando necessário, são chamadas funções que estão descritas na Figura 15, camada de controle, que são as funções alterar, salvar, excluir e ordenar.

Na Figura 19 é a camada visão em execução no navegador. Expondo todas as suas funcionalidades disponíveis.

Figura 19 - Visão em Tela

Fonte: Do Autor.

Foram criadas as camadas modelos, controles e visões para cada entidade que precisam ser salvas da aplicação, com disponibilidade de manipulação dos dados.

3.5 TIPOS DE ACESSO

De acordo com o levantamento de requisitos, são necessários dois tipos de acesso ao sistema, o acesso do avaliador com permissão para manipulação de todos os dados, e o acesso do professor que poderá manipular os dados de suas próprias avaliações.

3.5.1 Usuário Avaliador

Com permissão para a manipulação de todos os dados, ele poderá criar, salvar, editar e excluir todos os tipos de entidades existentes na aplicação.

Conforme a Figura 20 expõe é possível observar a disponibilidade de todos os atributos para a manipulação. Contando também com uma lista dos itens que estão aguardando a avaliação.

Figura 20 - Tela Avaliador

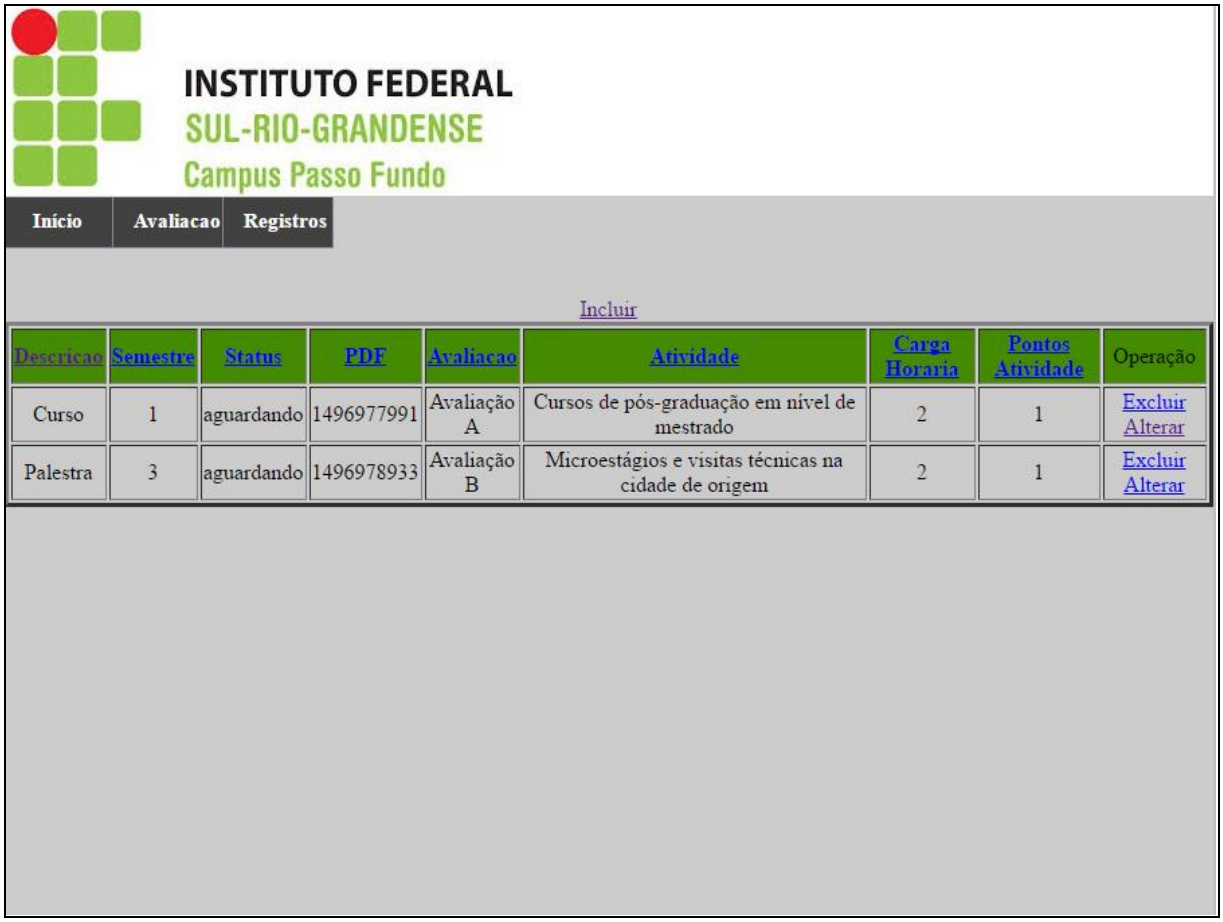
Fonte: Do Autor.

3.5.2 Usuário Professor

Com permissão apenas para a manipulação dos seus dados, o usuário professor poderá criar, salvar, editar e excluir as entidades avaliação e registro (item Atividade) na aplicação.

Conforme a Figura 21 expõe é possível observar a disponibilidade dos atributos avaliação e registros para a manipulação. Contando também com uma lista dos itens que foram cadastrados.

Figura 21 - Tela Professor



Fonte: Do Autor.

3.5.3 Login

Como este trabalho trata-se de um protótipo, o *login* foi desenvolvido de forma mais simplificada, sem o uso de banco de dados. Em primeiro momento, Figura 22, foi criada uma classe PHP, designada uma variável e inserido manualmente o usuário e senha, que será a base de dados.

Figura 22 - Base de Dados PHP

```php
<?php
$valida[admin] = "admin";
?>
```

Fonte: Do Autor.

Em seguida, foi implementada outra classe PHP chamada login.php, Figura 23, onde se conecta a base de dados da Figura 22, que recebe usuário e senha e valida se o usuário e senha estão cadastrados na base de dados.

Figura 23 - login.php

```php
<?php
error_reporting(0);
ini_set(display_errors, 0 );
?>
<?php
$user = $_POST['user'];
$pass = $_POST['pass'];
 include("bd.php");
if($valida[$user]==$pass){
setcookie('completo', '1');
 echo "<script>location.href='index_1.php'</script>";
 }
 else{
 echo "<font face=verdana size=2>";
 echo "Usuário ou senha incorretos!";
 echo "<br>";
 echo "<a href=login.html>";
 echo "Clique aqui</a> para tentar novamente.";
 echo "</a></font>";
 }
?>
```

Fonte: Do Autor.

Após, foi criado então um arquivo HTML chamado login.html, Figura 24 e 25, onde na Figura 24, apenas para critérios de visualização do arquivo por completo está sendo exposto, pois são as bases da página de exibição.

Figura 24 - login.html 1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <title>CPPD</title>
        <link href="estilo.css" rel="stylesheet" type="text/css" />
        <link rel="stylesheet" type="text/css" href="Scripts/superfish/superfish.css" />
        <script type="text/javascript" src="js/jquery-1.9.1.min.js"></script>
        <script type="text/javascript" src="Scripts/superfish/superfish.js"></script>
        <script type="text/javascript" src="js/jquery.mousewheel.js"></script>
        <link rel="stylesheet" type="text/css" href="Scripts/slidedeck/slidedeck.skin.css" media="screen" />

        <script type="text/javascript">
            //Executa a funcao quando o documento estiver carregado
            $(document).ready(function () {
                $('.navigation').superfish();
            });
        </script>

        <!-- Incluir aqui o SlideDeck jQuery script. -->
        <script type="text/javascript" src="js/slidedeck.jquery.lite.pack.js"></script>
        <style type="text/css">
            #slidedeck_frame {
                width: 850px;
                height: 300px;
            }
        </style>
```

Fonte: Do Autor.

Já na Figura 25 segue uma validação de usuário e senha via JavaScript, e após o formulário, que ao ser executado chama a classe login.php, Figura 23, para que seja feita toda a execução do login, com a validação na base de dados. Se o *login* for correto será direcionado para a página inicial da aplicação, e caso este incorreto, aparecerá a mensagem: Usuário ou senha incorretos!

Figura 25 - login.html 2

```
    <script type="text/javascript">
        function validaCampo()
        {
            if (document.login.user.value == "")
            {
                alert("Insira seu nome de usuário.");
                return false;
            }
            else
            if (document.login.pass.value == "")
            {
                alert("Insira sua senha.");
                return false;
            }
            else
                return true;
        }
    </script>
</head>
<body>
    <div id="container">
        <div id="topo">
            <img src="images/cabecalhoIFSUL.jpg"/>
        </div>    <!--fim da div topo -->
        <div align="center">
            <form action="login.php" method="post" name="login" id="login" class="login" onsubmit="return validaCampo();
                    return false;">
                <h1>Login</h1>
                Usuário: <br>
                <input type="text" name="user" id="user" class="user"> <br> <br>
                Senha: <br>
                <input type="password" name="pass" id="pass" class="user"> <br> <br>
                <input type="submit" value="Entrar">
            </form>
        </div>
    </div>
</body></html>
```

Fonte: Do Autor.

3.5.4 Controle De Acesso

Em cada tela que cada usuário terá permissão para visualizar, será necessário inserir o código de trecho descrito na Figura 26 e 27.

Figura 26 - Cookie Completo

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
<?php
if(IsSet($_COOKIE["completo"])){}
else{
echo '<meta http-equiv="refresh" content="0;url=login.html">';
exit;

}
?>
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
```

Fonte: Do Autor.

Figura 27 - Cookie Limitado

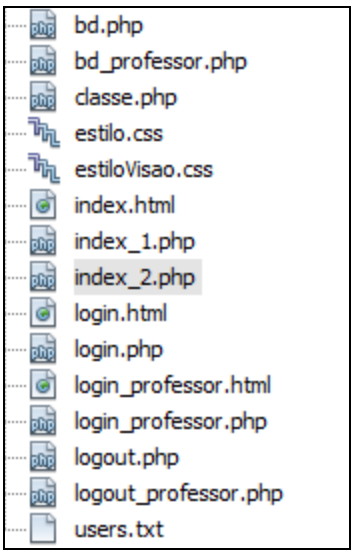
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
<?php
if(IsSet($_COOKIE["limitado"])){}
else{
echo '<meta http-equiv="refresh" content="0;url=login_professor.html">';
exit;

}
?>
```

Fonte: Do Autor.

Para cada tipo de usuário, foi criada uma tela separada. Para os usuários com acesso completo foi denominado ao Cookie como completo, e aos com acessos limitados como limitado. Cookie é um grupo de dados trocados entre o navegador e o servidor de páginas. A sua função principal é a de manter a persistência de sessões HTTP.

A organização das páginas iniciais foram todas duplicadas, Figura 28.

Figura 28 - Organização de arquivos



Fonte: Do Autor.

A tela inicial do sistema com *login* implementado, Figura 29, como a Figura 28 também descreve, ficou com campos separados para a efetivação do acesso.

Figura 29 - Tele inicial com login

Fonte: Do Autor.

3.5.5 Logout

Para que seja feito o *logout*, quando o usuário clicar em sair, será chamado a classe logout.php, Figura 30, e esta acionará uma função que deleta o Cookie e encerra a conexão, redirecionando para a tela inicial de *logins*.

Figura 30 – logout.php

```
k?php
setcookie("completo", "");
?>
<HTML>
<HEAD>
<TITLE>CPPD</TITLE>
<meta http-equiv="refresh" content="3;url=index.html">
<script language="JavaScript">
  function deleteCookie(nome){
    var exdate = new Date();
    exdate.setTime(exdate.getTime() + (-1 * 24 * 3600
       * 1000));
    document.cookie = nome + "=" + escape("")+ ((-1
       == null) ? "" : "; expires=" + exdate);
  }
</script>

</HEAD>
<BODY>
    <div align="center">
<FONT FACE="Verdana" SIZE="2">Você foi deslogado, será redirecionado automaticamente para a tela inicial!</FONT>
    </div>
<script language="JavaScript">
  deleteCookie("completo");
</script>

</BODY>
</HTML>
```

Fonte: Do Autor.

3.6 UPLOAD DO ARQUIVO

Para que o upload do arquivo fosse realizado, foi necessário usar alguns métodos próprios do PHP, conforme pode ser observado na Figura 30.

- No item 1: na *action* do *form* chamamos o arquivo onde se encontra a função para fazer o upload do arquivo, neste caso, vem a ser o mesmo arquivo onde o código está sendo escrito;
- No item 2: dentro do formulário se faz uma função para mudar o nome do arquivo para o tipo *time*(), pois assim, nunca terá um valor repetido no banco, e se atribui este nome ao dado que será repassado ao banco de dados;
- No item 3: é a função para que seja pego este arquivo via formulário e enviado para a pasta denominada arquivos.

Figura 31 - Upload de Arquivo

1
```
<form method="POST" action="VisaoItemavaliacao.php" enctype="multipart/form-data">
```

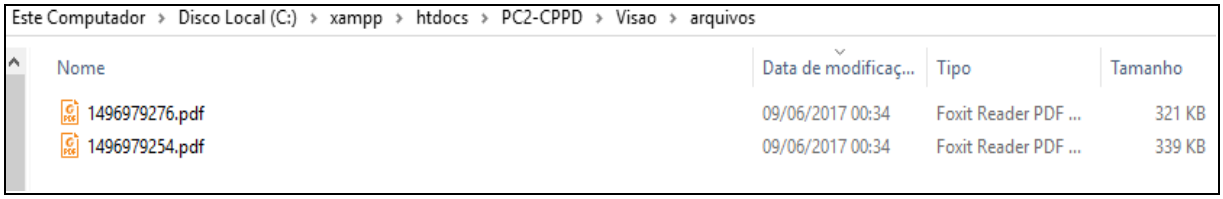
2
```
<tr align="left">
    <td>Documento</td>

    <?php
    // O nome original do arquivo no computador do usuário
    $arqName = $_FILES['pdf']['name'];

    $nome = time();
    $pdf = $nome;
    ?>

    <td><input type="file" name="pdf"  value="<?= @$pdf ?>" size="20"></td>
    <td><input type="text" name="pdf" value="<?= @$pdf ?>"></td>
</tr>
```

3
```
<?php
$uploaddir = 'arquivos/';
// O nome original do arquivo no computador do usuário
$arqName = $_FILES['pdf']['name'];
// Pega a extensão do arquivo enviado
$extensao = strtolower(end(explode('.', $arqName)));
// Define o novo nome do arquivo usando um UNIX TIMESTAMP
$nome = time() . '.' . $extensao;

if (move_uploaded_file($_FILES['pdf']['tmp_name'], $uploaddir . $nome)) {

    echo "Arquivo Enviado";
} else {
    echo "Houve um problema no upload do arquivo.";
}
?>
```

Fonte: Do Autor.

A Figura 31 ilustra a pasta arquivos com os documentos recebidos via formulário e já renomeados.

Figura 32 - Pasta Upload



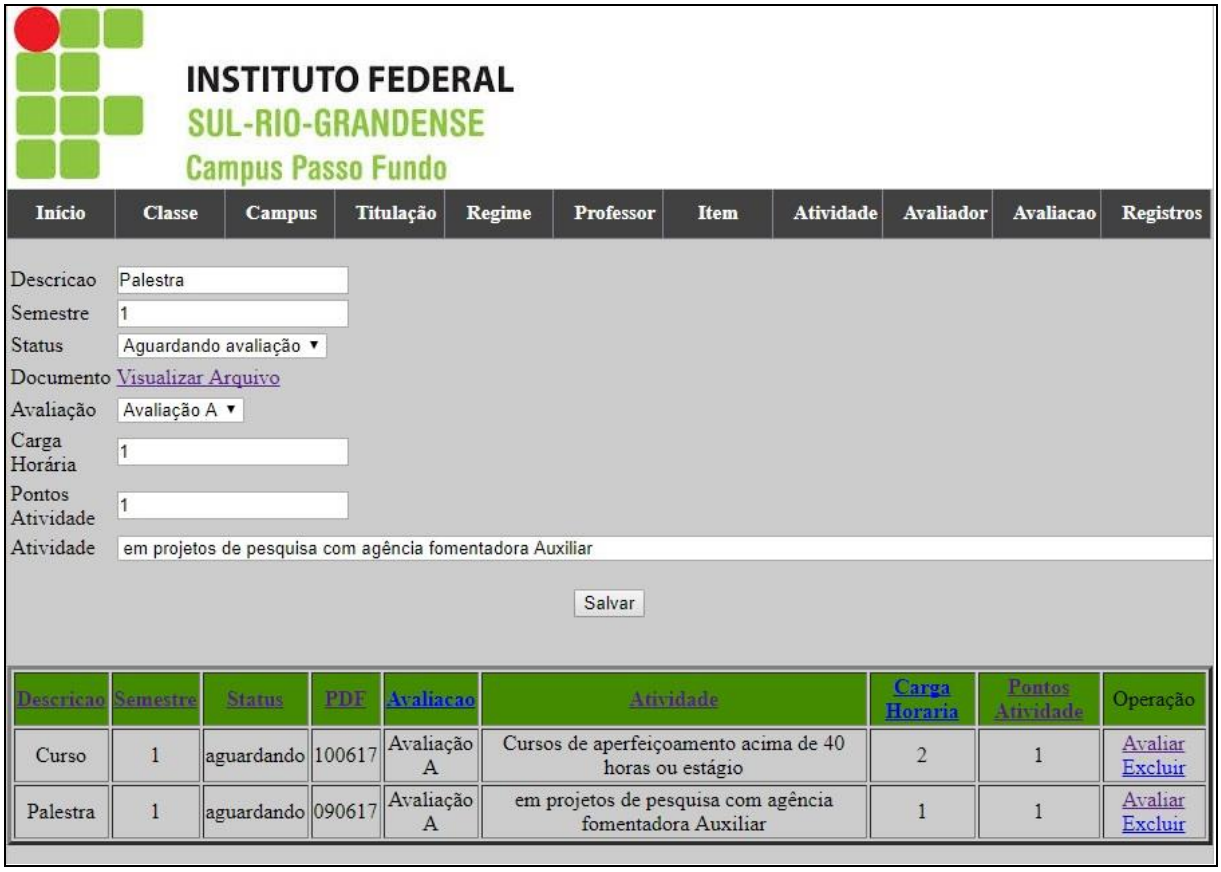
Fonte: Do Autor.

3.7 DOWNLOAD DO ARQUIVO

O avaliador poderá visualizar todos os registros pendentes de avaliação e avaliar, conforme ilustrado na Figura 32, podendo aceitar, recusar ou pendenciar o

registro. Os dados de cada registro, irão ser acessados pelo link Avaliar, que os mostrará na tela juntamente com o link disponibilizando o download do arquivo.

Figura 33 - Tela de Avaliação



Fonte: Do Autor.

Para viabilizar o arquivo para download, foi necessário selecionar o nome do arquivo pelo banco de dados e permitir que o usuário ao clicar no link, abra uma página no navegador exibindo o arquivo, Figura 33.

Figura 34 - Link para mostrar arquivo

```
<tr align="left">
    <td>Documento</td>

    <td> <a href="arquivos/<?= @$pdf ?>.pdf" target="blank">Visualizar Arquivo</a> </td>

    <td><input type="file" name="pdf"  value="<?= @$pdf ?>" size="20"></td>
    <td><input type="text" name="pdf" value="<?= @$pdf ?>"></td>
</tr>
```

Fonte: Do Autor.

4. CONSIDERAÇÕES FINAIS

Com o desenvolvimento desta aplicação, destaca-se o quão importante e o quanto uma ferramenta web pode ajudar nas soluções de alguns problemas.

Por se tratar de um protótipo, algumas funções não haviam sido consideradas no início do projeto, ocasionando a necessidade de reprogramação dos códigos no decorrer do desenvolvimento da aplicação.

A forma simplificada de implementação do controle de login supriu as necessidades de limitar os acessos validando o perfil do usuário.

Com a implementação deste protótipo foi atingido os objetivos inicialmente propostos, foram realizados testes nas funções implementadas que seria o cadastro e envio de informações, com armazenamento em banco de dados, controle de acesso por perfil de avaliador ou professor e centralização dos documentos enviados ao servidor, em ambos os testes o sistema se comportou de forma esperada.

5. TRABALHOS FUTUROS

A próxima etapa deste trabalho será a implementação de login e validação de perfil utilizando informações contidas no banco de dados, controle dos pontos obtidos, permitindo apenas a inserção dos valores máximos permitidos por cada atividade, modificação a forma como o professor insere o avaliador, que no momento, permite que o usuário professor insira o avaliador, na próxima alteração deve ficar este campo em aberto e o avaliador selecionar o registro para avaliação, e aí então sim sem atrelado o avaliador ao registro. Efetivar testes de maior escala e determinar quais os ajustes e melhorias a serem feitas. Gerar relatórios, implementar tabelas mais complexas, tornando a aplicação tão completa, que ao findar não seja mais necessário o uso da planilha do Excel que é usada atualmente.

REFERÊNCIAS

CONVERSE, Tim; PARK, Joyce. PHP a Bíblia. 2. ed. Rio de Janeiro: Campus, 2003.

ELMASRI, Ramez; NAVATHE, Shamkant B. Sistemas de banco de dados. 6. Ed. São Paulo. Person, 2010.

GUEDES, Gilleanes. UML 2 Guia prático. 2. Ed. São Paulo. Novatec. 2007.

GOURLEY, David; TOTTY, Brian. HTTP: The Definitive Guide. Sebastopol, Ucrânia: O'Reilly Media, 2002.

MEDEIROS, Ernani. Desenvolvendo Software com UML definitivo 2.0. São Paulo. Person. 2004.

MILANI, André. PostgreSQL Guia do Programador. São Paulo. Novatec, 2008.

OLIVEIRA, Kaui Aires. PostgreSQL x MySQL. Qual Escolher?. Devmedia. 2006. Disponível em: < http://www.devmedia.com.br/postgresql-x-mysql-qual-escolher/3923>. Acesso em: 16 de Nov. de 2014.

SILVA, Maurício Samy. Construindo Sites com CSS e (X)HTML. São Paulo. Novatec, 2007.

SILVA, Maurício Samy. JavaScript Guia do Programador. São Paulo. Novatec, 2010.

SILVA, Maurício Samy. JQuery A Biblioteca do Programador JavaScript. São Paulo. Novatec, 2010.

SILVA, Maurício Samy. jQuery UI Componentes de Interface Rica para Aplicações web. São Paulo. Novatec, 2012.

WELLING, Luke; THOMSON, Laura. PHP e MySQL Desenvolvimento Web. Rio de Janeiro:Elsevier,/2003.

ANEXOS

ANEXO A: Normas para a Avaliação de desempenho docente para fins de progressão funcional.

COMISSÃO PERMANENTE DE PESSOAL DOCENTE (CPPD)

NORMAS PARA AVALIAÇÃO DE DESEMPENHO DOCENTE PARA FINS DE PROGRESSÃO FUNCIONAL.

DA PROGRESSÃO FUNCIONAL DE DOCENTES.

Art. 1º - A progressão na carreira do magistério de 1º e 2º graus, do ensino médio, técnico, tecnológico e superior, far-se-á de um nível para outro imediatamente superior dentro da mesma classe, ou, de uma classe para a imediatamente superiora, após o cumprimento, pelo docente, do interstício de dezoito meses no nível respectivo, mediante avaliação de desempenho ou interstício de três anos de atividade em órgão público.

Art. 2º - Na contagem de interstício, para efeito da avaliação mencionada no artigo 1º, será descontado o tempo correspondente a:

A) faltas não justificadas;

B) suspensão disciplinar, inclusive a preventiva, quando dela resultar pena mais grave que a de repreensão;

C) período excedente a dois anos de licença ou suspensão de contrato, para tratamento de saúde, no caso de acidente de trabalho ou de doenças especificadas em lei;

D) licença para acompanhar cônjuge, para prestar assistência a familiar doente ou tratar de interesse particular;

E) cumprimento de pena privativa da liberdade, exclusivamente nos casos de crime comum;

Parágrafo único - Nas hipóteses das alíneas "B" e "E", se constatada a improcedência da penalidade ou da condenação, a contagem será restabelecida, computando-se o período correspondente ao afastamento.

Art. 3º - A avaliação do desempenho docente, para efeito de promoção na carreira do magistério, será baseada nos seguintes itens:

I desempenho didático-pedagógico;

II formação, aperfeiçoamento e atualização;

III produção intelectual;

IV prestação de serviços e outras atividades;

V administração.

Art. 4º - As atividades desenvolvidas em cada item serão pontuadas, conforme descrição a seguir;

I – DESEMPENHO DIDÁTICO PEDAGÓGICO - ATÉ 60 PONTOS

Serão consideradas atividades que configurem o envolvimento do docente nos processos de ensino-aprendizagem, tais como:

Aulas ministradas (40 horas) .... 2,5 pontos/aula

Aulas ministradas (20 horas) ..... 5 pontos/aula

Participação nas reuniões regulares de quartas-feiras ..... 5 pontos

II – FORMAÇÃO, APERFEIÇOAMENTO E ATUALIZAÇÃO DOCENTE

ATÉ 60 PONTOS

Serão consideradas atividades que configurem o envolvimento do docente em processos de formação continuada, tais como:

a) Microestágios e visitas técnicas na cidade de origem ..... 2 pontos

Microestágios e visitas técnicas fora da cidade de origem ..... 3 pontos/dia

b) Cursos de curta duração, equivalente a aperfeiçoamento ou atualização de conhecimentos (até 20 horas) ..... 4 pontos

c) Cursos de curta duração, equivalente a aperfeiçoamento ou atualização de conhecimentos (entre 20 e 40 horas) ..... 6 pontos

d) Cursos de aperfeiçoamento acima de 40 horas ou estágio ..... 8 pontos

e) Cursos de pós-graduação em nível de aperfeiçoamento com 180horas ..... 20 pontos

Cursos de pós-graduação em nível de especialização com 360 horas.....25 pontos

Cursos de pós-graduação em nível de mestrado ..... 50 pontos

Cursos de pós-graduação em nível de doutorado ..... 50 pontos

f) Participação como ouvinte em palestras, encontros, simpósios, congressos, eventos científicos, culturais, esportivos, artísticos ou similares ..... 2 pontos/dia

g) Participação em grupos de estudo, relacionados à formação continuada

Até 40 horas ..... 3 pontos

Acima de 40 horas ..... 6 pontos

Obs. Docentes matriculados como aluno especial em cursos de pós-graduação, pontuarão de acordo com os itens B, C e D.

III – PRODUÇÃO INTELECTUAL - ATÉ 60 PONTOS

Serão consideradas as atividades que configurem o envolvimento do docente em processos de produção de conhecimento relativo a sua área de atuação, assim como apresentação e divulgação de resultados, conforme segue:

a) Participação em projetos de pesquisa com agência fomentadora..... 20 pontos (coord.)

Participação em projetos de pesquisa com agência fomentadora..... 15 pontos (aux.)

Participação em projetos de pesquisa sem agência fomentadora..... 8 pontos (coord.)

Participação em projetos de pesquisa sem agência fomentadora..... 5 pontos (aux)

b) Apresentação de palestras, trabalhos em congresso, minicursos ou similares...10 pontos /atividade

c) Cursos igual ou acima de 40 horas como ministrante..... 12 pontos/atividade

Cursos igual ou acima de 40 horas como coordenador..... 5 pontos/atividade

Cursos igual ou acima de 40 horas como monitor..... 2 pontos/atividade

d) Elaboração de trabalhos técnicos, relatórios técnicos, por determinação da área e/ou para atender convênios, pesquisa ou extensão..... 15 pontos (coord.)

Idem para auxiliar ..... 10 pontos

e) Tradução de obras literárias, técnicas ou científicas, editadas, desde que no âmbito de sua atuação ou na área da educação (completas) ..... 15 pontos

Idem para capítulo de um livro ..... 7 pontos

Idem para apenas um documento ..... 2 pontos

f) Participação em evento artístico-cultural, científico e esportivo, relativo à atividade docente, com exposição ou divulgação ..... 5 pontos

g) Publicação de artigos científico, resumos de Anais ..... 20 pontos (internacional)

Publicação de artigos científico, resumos de Anais ..... 15 pontos (nacional)

Publicação de artigos científico, resumos de Anais .....10 pontos (local)

h) Publicação de artigos em jornais ..... 2 pontos

i) Participação em projetos inovadores na área do ensino ..... 5 pontos/projeto (coord.)

Participação em projetos inovadores na área do ensino.....3 pontos/projeto (aux.)

j) Publicações técnicas, didáticas, científicas e literárias em sua área de atuação ou na área da educação. Livro ..... 40 pontos (autor ou co-autor)

Idem para capítulo de livro ..... 20 pontos (autor ou co-autor)

Elaboração de apostila, avaliada pela coordenadoria ..... 5 pontos

k) Revisão de artigos técnicos ou científicos .... 2 pontos

l) Orientação de trabalho de mestrado ou doutorado ..... 15 pontos

Orientação de trabalho de especialização e pesquisa TCC ..... 8 pontos

IV – PRESTAÇÃO DE SERVIÇOS E OUTRAS ATIVIDADES – ATÉ 60 PONTOS

Como prestação de serviços serão consideradas as atividades de envolvimento em processos de aplicação de conhecimentos ou extensão dos serviços do IFSUL à comunidade, devidamente autorizadas, conforme segue:

a) Participação em projeto de extensão de caráter científico, artístico-cultural,

esportivo ou de prestação de serviços ..... 10 pontos (coord.)

Idem ..... 8 pontos (aux.)

b) Assessoria e/ou consultoria na área de atuação ou na de educação como representante do IFSUL. (exemplo: portarias do MEC) ..... 10 pontos

c) Participação em comissão de coordenação/organização de congressos ou similares (internacional) ..... 15 pontos (coord.) .....12 pontos (aux.)

Participação em comissão de coordenação/organização de congressos ou similares (nacional) ..... 12 pontos (coord.) ..... 8 pontos (aux.)

Participação em comissão de coordenação/organização de congressos ou similares (regional) ..... 8 pontos (coord.) ..... 6 pontos (aux.)

Participação em comissão de coordenação/organização de congressos ou similares (local) ..... 6 pontos (coord.) ..... 5 pontos (aux.)

d) Partic. em bancas de defesa de mestrado ou doutorado .....15 pontos

Partic. em bancas de qualificação de mestrado ou doutorado ..... 10 pontos

Participação em bancas de exames de defesa de TCC ..... 5 pontos

e) Participação em bancas de concurso público, comissões de avaliação de projetos, ou de organização de semana de curso ..... 3 pontos

f) Coordenação de grupos de trabalho em congressos ou afins ..... 5 pontos

g) Representação do IFSUL em reuniões, eventos, etc ..... 2 pontos

h) Orientação de projeto final de avaliação (cursos técnicos ou graduação)..... 2 pontos

V – ADMINISTRAÇÃO – ATÉ 60 PONTOS

Serão assim classificadas as atividades do docente na administração do Instituto, envolvendo:

a) Docentes em atividade administrativa (CD) ..... 50 pontos (12 meses)

Docentes em atividade administrativa (CD) ..... 4 pontos por mês

Docentes em atividade administrativa (CD) ..... 2 pontos até 15 dias

b) Docentes afastados para prestar serviço no Ministério da Educação, entidades de classe e em outras situações previstas na legislação vigente ..... 50 pontos

c) Docentes em atividade administrativa (FG) ..... 30 pontos

d) Docentes em cargo de coordenação de áreas ou cursos (técnico, tecnológico, graduação ou especialização) ..... 20 pontos

e) Docentes em atividade de secretaria de áreas ..... 10 pontos

Art. 5º Será descontado um ponto por falta não justificada em cada turno e um ponto a

cada duas impontualidades comprovadas e não justificadas.

Art. 6º O docente será avaliado, automaticamente, pela CPPD, a cada dezoito meses de efetivo exercício no nível em que se encontra.

Art. 7º Para efeito de progressão, será calculada a média aritmética dos pontos obtidos em cada semestre do interstício.

Art. 8º Estará habilitado a progredir para o nível seguinte o docente que obtiver no mínimo média igual ou superior a 50 pontos.

Art. 9º O docente deverá enviar à CPPD os comprovantes das atividades, objeto de avaliação, referentes ao interstício de dezoito meses, conforme artigo 4º, com uma antecedência mínima de trinta dias da sua avaliação.

Art. 10º O docente terá acesso a pontuação obtida em sua avaliação.

Parágrafo único – No caso de não concordância com a pontuação recebida, caberá recurso ao Conselho Superior.

Art. 11º No período em que o docente estiver afastado para licença especial ou licença de saúde, fará jus a pontuação mínima para progressão.

Art. 12º No período em que o docente estiver afastado para licença capacitação, fará jus à pontuação mínima para progressão, desde que apresente certificado do curso realizado.

Art. 13º Os casos omissos serão resolvidos em reunião da CPPD.

Art. 14º Este sistema de avaliação entra em vigor na data de sua aprovação pelo Conselho Superior.

ANEXO B – Planilha de Registro de Atividades

CPPD	FICHA DE AVALIAÇÃO	Interst. De				
Nome:				Cla. / Nív.=		
Lotação: Campus P. Fundo		Titulação:				
Regime de trabalho (coloque x):		**40 horas**		**20 horas**		
Itens	**Denominação**	**Pont.**	**1° sem.**	**2°sem.**	**3°sem.**	**4°sem.**
I	Desempenho didático pedagógico	60	**0**	**0**	**0**	**0**
	Número de aulas semanais					
	Horas de reunião pedagógica regular (até 2h)					
I I	Formação, aperfeiçoamento e atualização	60	**0**	**0**	**0**	**0**
I I I	Produção intelectual	60	**0**	**0**	**0**	**0**
I V	Prestação de serviços e outras atividades	60	**0**	**0**	**0**	**0**
V	Administração	60	**0**	**0**	**0**	**0**
VI	Outras atividades	30	**0**	**0**	**0**	**0**
VII	Portarias	30	**0**	**0**	**0**	**0**
	Sub-total		**0**	**0**	**0**	**0**
VIII	Descontos					
	Faltas não justificadas					
	Impontualidades					
	TOTAL		**0**	**0**	**0**	**0**
	MÉDIA DOS SEMESTRES		**0**			