UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS CORNÉLIO PROCÓPIO
DEPARTAMENTO DE COMPUTAÇÃO
ENGENHARIA DE COMPUTAÇÃO

KLEBER CAMPOS VIANA

SISTEMA DE AVALIAÇÕES DE DESEMPENHO E SOLICITAÇÃO DE TRABALHOS PARA EMPRESA MOORE PRISMA

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
2022

KLEBER CAMPOS VIANA

SISTEMA DE AVALIAÇÕES DE DESEMPENHO E SOLICITAÇÃO DE TRABALHOS PARA EMPRESA MOORE PRISMA

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina Trabalho de Conclusão de Curso II, do curso de Engenharia de Computação da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para a obtenção do título de Bacharel.

Orientador: Prof. Dr. Antonio Carlos Fernandes da Silva

CORNÉLIO PROCÓPIO
2022

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Departamento Acadêmico de Computação
Coordenação de Engenharia de Computação
Engenharia de Computação

TERMO DE APROVAÇÃO

Sistema de avaliações de desempenho e solicitação de trabalhos para empresa Moore Prisma

por

Kleber Campos Viana

Este Trabalho de Conclusão de Curso de Engenharia de Computação foi julgado adequado para obtenção do Título de Engenheiro de Computação e aprovado em sua forma final pelo Programa de Graduação em Engenharia de Computação da Universidade Tecnológica Federal do Paraná.

Cornélio Procópio, 07 de dezembro de 2022

_____

Prof. Dr. Antonio Carlos Fernandes da Silva

_____

Prof. Dr. Silvio Ricardo Rodrigues Sanches

_____

Prof. Dr. Alexandre L'Erario

"A Folha de Aprovação assinada encontra-se na Coordenação do Curso"

Dedico este trabalho à minha família, e a todos que me apoiaram durante os anos de graduação.

AGRADECIMENTOS

Agradeço ao meu orientador Prof. Dr. Antonio Carlos Fernandes da Silva, pela confiança no meu potencial, e pelo apoio nos artigos desenvolvidos para o projeto de extensão ELLP, e em momentos difíceis da minha graduação.

Gostaria de deixar registrado também, o meu reconhecimento à minha família, pois sem o apoio deles seria muito difícil vencer esse desafio. Aos amigos, agradeço por estarem ao meu lado festejando minhas conquistas, e especialmente por me apoiarem em momentos de dificuldades.

Agradeço a todos os servidores da UTFPR-CP, pois seus esforços tornaram possível que eu apresentasse este trabalho.

RESUMO

VIANA, Kleber Campos. **Sistema de avaliações de desempenho e solicitação de trabalhos para empresa Moore Prisma.** 2022. 43 f. Trabalho de Conclusão de Curso (Graduação) – Engenharia de Computação. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2022.

Este trabalho tem como objetivo apresentar o desenvolvimento de um sistema web voltado para aumentar a produtividade e assertividade nas atividades realizadas pelas equipes de auditores da Moore Prisma, empresa de auditoria e consultoria contábil. A ferramenta substitui a utilização de planilhas e formulários de serviços em nuvem, e possibilita diretores, encarregados e assistentes realizarem suas avaliações de desempenho e solicitarem trabalhos de auditoria para o departamento interno (*BackOffice*). O sistema atende os departamentos de auditoria interna, externa, de serviços financeiros e o departamento interno.

Palavras-chave: Desenvolvimento Web. PHP. PDO. MySQL. AJAX.

ABSTRACT

VIANA, Kleber Campos. **Performance appraisal and job application system for Moore Prisma**. 2022. 43 f. Trabalho de Conclusão de Curso (Graduação) – Engenharia de Computação. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2022.

This work aims to present the development of a web system aimed at increasing productivity and assertiveness in the activities carried out by the teams of auditors at Moore Prisma, an auditing and accounting consulting firm. The tool replaces the use of cloud service spreadsheets and forms, and enables directors, supervisors and assistants to carry out their performance evaluations and request audit work for the internal department (BackOffice). The system serves the internal audit, external audit, financial services and internal departments.

Keywords: Web Development. PHP. PDO. MySQL. AJAX.

SUMÁRIO

1 INTRODUÇÃO

A Moore Prisma, localizada em Ribeirão Preto, é uma empresa de auditoria contábil que atende clientes de todo o país. Sendo o maior escritório da América Latina da rede Moore, nos últimos anos a Moore Prisma vem executando um projeto que mira o desenvolvimento de jovens talentos para as áreas de auditoria contábil e tributária, e no ano de 2021 se tornou referência na rede ao criar um departamento de TI junto ao time de auditoria contábil.

A análise quase instantânea de dados e informações são a força motriz de diversas novas tecnologias relacionadas a *data analytics*. Tal disponibilidade e o acesso a dados têm transformado a relação de trabalho de diversas profissões. Com a contabilidade, são inegáveis a agilidade e o dinamismo gerados por métodos e softwares cada vez mais responsivos e dinâmicos (COSCODAI, 2022).

Além da velocidade na análise de dados, outro objetivo desejado na criação de um departamento de TI é a utilização de tecnologias visando a otimização de funções organizacionais. Tal como afirma Morin (1992) “a tecnologia quando vista como uma função organizacional é passível de gestão e quando orientada a estratégia da empresa” portanto, tem caráter estratégico e, na base da formulação estratégica está a identificação das tecnologias relacionadas a empresa.

No contexto da Moore Prisma, ambas as frentes tecnológicas, de análise de dados e de otimização das tarefas organizacionais, foram implementadas. Foi adotada uma nova forma de realizar os trabalhos de auditoria, automatizando seus exames de auditoria contábil (através de scripts), e em paralelo foram desenvolvidos os próprios sistemas internos que substituem a utilização de planilhas ou formulários de serviços em nuvem.

Este trabalho tem como objetivo o desenvolvimento um sistema web que possibilite diretores, encarregados e assistentes realizar suas avaliações de desempenho, e para encarregados e diretores solicitarem trabalhos de auditoria para o departamento interno (*backoffice*).

1.1 Objetivos

Criar um sistema que possibilite a realização de avaliações de desempenho e solicitações de trabalhos para o Departamento Interno, com interfaces criadas para gestores e usuários finais. Com a utilização, gestores poderão acompanhar o desempenho dos membros de cada equipe selecionada para determinado trabalho, e o desenvolvimento das atividades solicitadas ao Departamento Interno conforme seus status de execução forem evoluindo.

2 FUNDAMENTAÇÃO TEÓRICA

Os conceitos relacionados ao Trabalho de Conclusão de Curso e as tecnologias utilizadas para o desenvolvimento foram escolhidos de maneira a atender o tempo estipulado para a entrega do sistema, considerando tanto o grande leque de ferramentas disponíveis quanto a simplicidade de cada uma para o desenvolvimento.

Na Seção 2.1 são apresentadas as ferramentas de software e o ambiente de desenvolvimento utilizados para a criação do sistema. Na sequência, na seção 2.2 são apresentadas as linguagens utilizadas no projeto, assim como as necessidades que cada uma atende no sistema. Na Seção 2.3 é apresentado o processo de desenvolvimento do sistema junto aos departamentos que utilizam a ferramenta.

2.1 Ferramentas de software e Ambiente de Desenvolvimento

Foram utilizadas as seguintes ferramentas para o desenvolvimento:

- Visual Studio Code (VS Code) é um editor de código de código aberto desenvolvido pela Microsoft. A ferramenta, disponível para Windows, macOS e Linux, vem com suporte integrado para JavaScript, TypeScript e Node.js e possui um rico ecossistema de extensões para outras linguagens e tempos de execução (CODE, 2022).
- PHPMyAdmin é uma ferramenta de software livre escrita em PHP utilizada para a administração de um servidor de banco de dados MySQL ou MariaDB. Seu uso é voltado para executar as tarefas de administração, criação, adição de contas de usuário, consultas, inserções e exclusões de um banco de dados (RATSCHILLER, 2022).
- Cpanel é um painel de controle baseado em Linux que permite gerenciar sites que estão dentro de uma hospedagem. Ele permite a instalação de aplicações, monitoramento de desempenho das páginas e ainda realizar configurações de ambiente. Com o Cpanel é possível realizar ações administrativas em um painel visual, sem precisar executar comandos complexos (LONGEN, 2022).
- brModelo é uma ferramenta de código aberto voltada para ensino e desenvolvimento de modelagem de banco de dados relacionais. Nele é possível a criação de diagramas e fluxogramas que abrangem as etapas conceitual, lógico e físico (NETO, 2016).

2.2 Linguagens e extensões utilizadas

Para desenvolver uma ferramenta que possibilite a criação de formulários para persistência dos dados de avaliações e solicitações, foi escolhido o PHP, linguagem que foi utilizada para a criação das páginas de cadastro, consulta, inserção e remoção de dados do sistema.

A linguagem de banco de dados escolhida foi o MySQL, administrado com o apoio da ferramenta PHPMyAdmin. Sendo assim, para a utilização em conjunto do banco de dados MySQL e o PHP foi o escolhido o *PHP Data Object* que permite que o PHP se comunique com o banco de dados relacional.

As linguagens de programação, marcação, estilo e de banco de dados utilizadas foram:

- PHP, que é uma sigla para *Hypertext Preprocessor* (Pré Processador de Hipertexto), é uma linguagem de programação *open source* de uso geral, que foi especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do HTML (PHP, 2022).
- O HTML, sigla de *Hyper Text Markup Language* (Linguagem de Marcação de HiperTexto), é uma linguagem de marcação que permite inserir o conteúdo e estabelecer a estrutura básica de um website. HTML5 é a sigla para a última versão do HTML, que trouxe novas funcionalidades para navegadores em múltiplas plataformas (MARQUES, 2022)
- Javascript é uma linguagem de programação usada por desenvolvedores para fazer páginas interativas da Internet. Sua utilização visa uma melhoria na experiência do usuário no momento da navegação em um site, através de animações, requisições sem atualização de páginas, criação de máscaras para campos, validadores de documentos, dentre outras funcionalidades (AWS, 2022).
- CSS3 é chamado de linguagem "*Cascading Style Sheet*" e é usado para estilizar elementos escritos em uma linguagem de marcação. O CSS3 separa o os elementos que compõe uma página web da representação visual do site, possibilitando personalizar a aparência das páginas mantendo tais configurações em um arquivo específico. (GONÇALVES, 2022).
- MySQL é um sistema de gerenciamento de banco de dados relacionais de código aberto. Seu funcionamento segue o modelo cliente-servidor e é utilizado

para criar e gerenciar bancos de dados baseados em um modelo de relação entre dois lados. (LONGEN, 2022).

- PDO, sigla de *PHP Data Objects* (Objetos de Dados PHP), é uma extensão do PHP que possibilita a aplicação se comunicar com diferentes tipos de bancos de dados. A extensão possibilita uma abstração da forma de acessar às informações contidas em um banco de dados, sendo os seus métodos independentes para cada um dos tipos suportados. (NOLETO, 2022).
- AJAX (*Asynchronous JavaScript and XML*) é um conjunto de técnicas de desenvolvimento voltado para a web que permite que aplicações trabalhem de modo assíncrono (o “Asynchronous”, da sigla), processando qualquer requisição ao servidor em segundo plano (GONÇALVES, 2022).
- JQuery é uma biblioteca JavaScript que possui recursos voltados para facilitar a manipulação de elementos HTML, manipulação de eventos, animações, a utilização do AJAX e demais ações que trazem versatilidade e rapidez ao carregar o conteúdo de uma página web. Suas linhas de código simplificam os scripts interpretados pelo navegador do lado cliente (JQUERY, 2022).

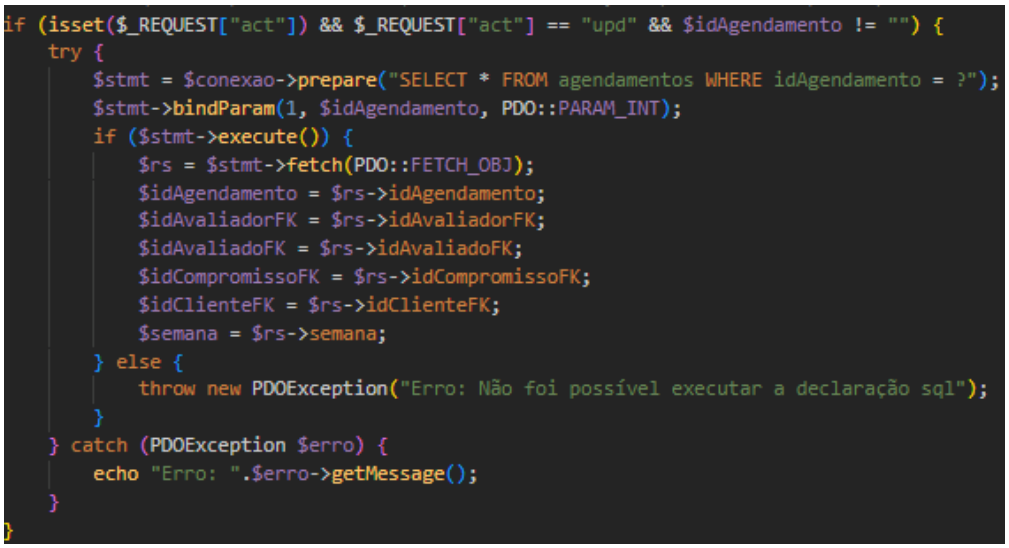
2.2.1 PHP Data Objects

O PDO define uma interface de conexão a banco de dados leve e consistente para PHP. Há a possibilidade de utilização de diversos drivers de conexão que implementam a interface do PDO para vários tipos de bancos de dados. Ele fornece uma camada de abstração de acesso a dados, isso significa que independentemente do sistema de banco de dados escolhido, é possível utilizar as mesmas funções para realizar consultas, inserções, modificações ou exclusões dos dados persistidos (PHP, 2022).

Prepared Statements

O PDO suporta *prepared statements*, que são comandos SQL pré-compilados que permitem a criação de comandos SQL de forma genérica e a execução dos mesmos várias vezes, com os parâmetros reais sendo substituídos pelos valores específicos da execução (NOLETO, 2022). Um exemplo da utilização dos *prepared statements* pode ser conferida na Figura 1.

Figura 1 - Exemplo de utilização dos *prepared statements*



```php
if (isset($_REQUEST["act"]) && $_REQUEST["act"] == "upd" && $idAgendamento != "") {
    try {
        $stmt = $conexao->prepare("SELECT * FROM agendamentos WHERE idAgendamento = ?");
        $stmt->bindParam(1, $idAgendamento, PDO::PARAM_INT);
        if ($stmt->execute()) {
            $rs = $stmt->fetch(PDO::FETCH_OBJ);
            $idAgendamento = $rs->idAgendamento;
            $idAvaliadorFK = $rs->idAvaliadorFK;
            $idAvaliadoFK = $rs->idAvaliadoFK;
            $idCompromissoFK = $rs->idCompromissoFK;
            $idClienteFK = $rs->idClienteFK;
            $semana = $rs->semana;
        } else {
            throw new PDOException("Erro: Não foi possível executar a declaração sql");
        }
    } catch (PDOException $erro) {
        echo "Erro: ".$erro->getMessage();
    }
}
```

Fonte: Autoria Própria.

No trecho apresentado na Figura 1, onde as informações são recuperadas do banco para que posteriormente seja realizado um update, nota-se que a função *bindParam* é responsável por passar a variável e o seu tipo de parâmetro para a query. Assim, caso não dê erro ao ser executada a função *execute(),* são passados os valores de cada campo da tabela do banco de dados, através do objeto $rs, para suas respectivas variáveis.

Tratamento de transações

O tratamento de transações é útil para garantir que um conjunto de operações em determinado banco de dados sejam concluídas como um todo. Sendo assim, caso todas as operações não tiverem sido concluídas com sucesso, a transação não será considerada bem-sucedida pelo sistema ou aplicação. Se uma dessas transações for interrompida, todas as outras são revertidas para o estado inicial (NOLETO, 2022).

Parametrização das queries

Possibilita criar uma query com parâmetros genéricos e, depois, substituir esses parâmetros pelos valores reais quando for executá-la. Com a parametrização de queries a aplicação passa a ser mais segura, pois evita que ataques de injeção SQL ocorram na aplicação, pois os valores reais dos parâmetros passados são enviados para o banco de dados separadamente da query (NOLETO, 2022).

Tratamento de exceções

Para o tratamento de exceções o PDO possui a classe *PDOException*, que estende a *RuntimeException,* e é responsável por capturar os erros gerados no PDO (PHP, 2022). Uma forma de tratar erros utilizando o tratamento de exceções do PDO pode ser conferido na Figura 2.

Figura 2 – Tratamento de erros utilizando a classe *PDOException*

```
7   // Cria a conexão com o banco de dados
8   try {
9       $conexao = new PDO("mysql:host=localhost;dbname=a******_avaliacoes", "a******_kviana", "***************");
10      $conexao->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
11      $conexao->exec("set names utf8");
12  } catch (PDOException $erro) {
13      echo "Erro na conexão:".$erro->getMessage();
14  }
```

Fonte: Autoria Própria.

A Figura 2 apresenta a utilização da classe *PDOException* na criação de uma conexão com um banco de dados. No exemplo, caso as credenciais informadas não estejam corretas, a classe é acionada, informado o erro, e em seguida ele é exibido através do método *getMessage*().

2.2.2 AJAX e JQuery

O AJAX, acrônimo de *Asynchronous JavaScript and XML*, é um conjunto de técnicas de desenvolvimento web que possibilita a criação de páginas web mais interativas. Um dos principais objetivos é permitir que aplicações trabalhem de modo assíncrono, processando requisições ao servidor em segundo plano, sem que seja necessária a atualização da página como um todo (GONÇALVES, 2022).

Para a realização das realizar requisições assíncronas é possível utilizar o *jQuery em conjunto com o AJAX. JQuery é* uma das principais bibliotecas *JavaScript,* que provê funções com estrutura simplificada para trabalhar com Ajax. Na Figura 3 é apresentado um exemplo de criação de uma requisição utilizando a função $.ajax().

Figura 3 – Estrutura que realiza uma requisição AJAX com JQuery

```
826  $("#idAvaliadorFK").on("change", function() {
827      $.ajax({
828          url: 'https://auditoriamooreprisma.com.br/controllers/pegaValores.php',
829          type: 'POST',
830          data: {
831              idAvaliados: $("#idAvaliadorFK").val()
832          },
833          beforeSend: function() {
834
835              $("#labelAgendamento").css({
836                  'display': 'none'
837              });
838          },
839          success: function(data) {
840              $("#idAvaliadoFK").css({
841                  'display': 'block'
842
843              });
844          },
845          error: function(data) {
846              $("#idAvaliadoFK").css({
847                  'display': 'block'
848              });
849              $("#idAvaliadoFK").html("Houve um erro ao carregar");
850          }
851      });
852  });
```

Fonte: Autoria Própria

Na linha 828 é informada a URL para a qual é enviada a requisição. Na linha 829 é informado qual método/verbo HTTP é utilizado na requisição e na linha 830 é enviada a informação, no corpo da requisição.

Linhas 833 a 838: o argumento *beforeSend* recebe uma função que é executada antes da requisição ser enviada. Entre as linhas 839 e 843, o argumento *success* recebe, na variável data, a informação após a informação ser recebida do lado servidor. No caso do argumento *error*, das linhas 845 a 850, é exibida uma mensagem caso os dados não tenham sido recebidos após a execução da requisição.

2.3 Padrão *Model-View-Controller*

Utilizado no início do desenvolvimento da aplicação, o padrão MVC é um padrão de arquitetura de software. O MVC, sigla de Model View Controller, que significa Modelo, Visão e Controle, tem como característica separar uma aplicação em três grupos de componentes, os modelos, as exibições e os componentes. As solicitações dos usuários são encaminhadas para um controlador, que trabalha em conjunto com o modelo para executar as ações do usuário. Fica a cargo do controlador

definir qual exibição será exibida para o usuário, de modo que ela apresente os dados solicitados do modelo. Isto permite que sejam criadas diversas interfaces, que podem ser modificadas sem a necessidade de alterar as regras de negócios (SMITH, 2022).

Modelo

O responsável por acessar os dados da aplicação é o modelo, que ainda pode ser dividido em classes de domínio, que representam o estado dos objetos e as regras do negócio definidas para eles. É o modelo quem recebe as informações do controle e as valida caso estejam corretas. (GONÇALVES, 2022).

Visão

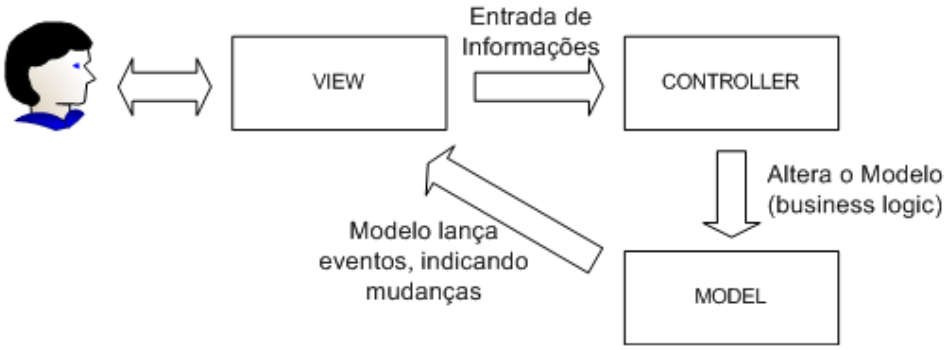
É responsável por apresentar as informações para o usuário. Sem contato direto com as regras de negócio da aplicação, a *visão* recebe instruções do controle e informações do modelo para exibi-las para o usuário. Para a visão é necessário que haja o mínimo de lógica referente as regras de negócio envolvida, e qualquer lógica contida na visão deve se relacionar exclusivamente à apresentação do conteúdo (SMITH, 2022).

Controle

É quem interpreta as entradas de informações enviadas pelo usuário através da visão, com as respostas fornecidas pelo modelo, processando os dados e os enviando para as demais camadas. É o controle que permite a comunicação entre o banco de dados e o restante da aplicação (SMITH, 2022).

Uma representação da comunicação entre os grupos de componentes do MVC pode ser conferida na Figura 4.

Figura 4 – Ilustração da comunicação entre os grupos no Padrão MVC.



Fluxo de eventos e informações em uma arquitetura MVC (ALMEIDA, 2022)

2.4 Trabalhos relacionados

Durante o desenvolvimento, duas ferramentas foram estudadas e tidas como referência de eficiência para o gerenciamento de solicitações.

2.4.1 Pipefy

Pipefy é uma ferramenta que possibilita a implementação de fluxos de trabalho e a personalização de entregas. A aplicação se faz presente desde a criação da demanda à execução e relatórios para análise de resultados. A principal utilização do Pipety é para o gerenciamento de processos como gestão de RH, compras e acompanhamento de chamados de clientes.

2.4.2 Sankhya ERP

O ERP Sankhya é uma ferramenta que possui soluções integradas que auxiliam a gestão de equipes. A aplicação possibilita conferir em tempo real informações sobre desempenho dos colaboradores, além de possuir interfaces para geração de relatórios e oferecer serviços de gestão de compras/estoque, CRM, cotação eletrônica, dentre outros.

Embora proporcione diferentes visualizações das informações relacionadas aos colaboradores e também a ordens de serviço, o ERP não possui uma funcionalidade que permita realizar avaliações de desempenho ou solicitações de trabalhos.

3 DESENVOLVIMENTO DO PROJETO

Para se dar início a etapa de desenvolvimento foi desenhado, com o acompanhamento dos responsáveis de cada um dos departamentos envolvidos, o diagrama de banco de dados. Em cerca de 5 reuniões forem definidas as principais necessidades e, com o diagrama de banco como apoio, deu-se início ao desenvolvimento.

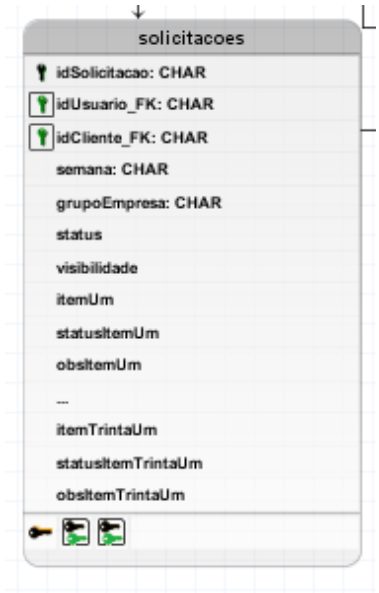
3.1 Levantamento de requisitos

As principais necessidades do sistema foram elencadas a partir de entrevistas com a equipe do departamento interno, para o projeto de solicitações, e com a diretoria da Auditoria, para o projeto de avaliações. O processo se estendeu até as primeiras etapas do desenvolvimento, onde eram apresentadas prévias das telas de navegação com exemplos das atividades a serem realizadas, de maneira que houvesse um alinhamento entre as partes.

Para as solicitações, as informações necessárias para que um colaborador possa realizar uma solicitação de trabalhos para o departamento interno são colhidas em um formulário que registra qual é o encarregado do trabalho, o cliente de qual serão enviados os arquivos, a semana da visita (em que a equipe realizará os trabalhos), nomes de possíveis filiais e a lista dos trabalhos de auditoria solicitados. Como os trabalhos de auditoria são divididos em três etapas, sendo elas preliminar, controle interno e final, também foi solicitado que o sistema apontasse em qual etapa o solicitante pretende receber os trabalhos prontos.

Ainda para o projeto de solicitações, também foi solicitado que pudessem ser acompanhados os status de progresso dos trabalhos de uma solicitação individualmente, e o status geral da avaliação. Para que os administradores das demandas de solicitações (departamento interno) pudessem controlar quais solicitações ficariam visíveis para os demais auditores, foi criado o campo visibilidade para tal controle. Os campos e tipos de dados dos campos da tabela solicitações, após as diversas etapas de entrevistas, podem ser conferidos na Figura 5.

Figura 5 – Tabela de solicitações



Fonte: Autoria Própria

O levantamento de requisitos para o projeto de avaliações de desempenho apresentou de início a necessidade de haver não apenas uma tabela para o recebimento das avaliações, mas também uma para o agendamento dessas avaliações. Uma vez que as visitas das equipes nos clientes são agendadas previamente no sistema Sankhya, a profissional responsável pelo agendamento de avaliações de desempenho cria os agendamentos com duas semanas de antecedência, de maneira que além de um controle de agendamento de avaliações, reúne também as informações de qual cliente, semana e equipe cada auditor trabalhará nas próximas semanas.

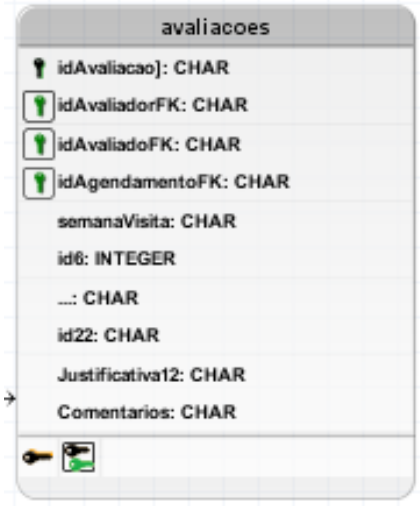
Na Figura 6 é possível observar a tabela agendamentos como os agendamentos de trabalho, reunindo as informações nome do avaliador, nome do avaliado, cliente, semana da visita forma:

Figura 6 – Tabela de agendamentos

Fonte: Autoria Própria

Para que uma avaliação seja realizada, é necessário haver previamente um agendamento criado. A tabela de avaliações, que pode ser vista na Figura 7, também armazena as informações semana da visita, e listados como id6, id7...até o id 22, os itens da avaliação de desempenho. Os itens são listados com a mesma numeração dos itens no antigo formulário utilizado para a empresa, e durante o desenvolvimento foi considerado apropriado para a equipe utilizar essa numeração. Cada item compõe uma competência exigida, com desempenho mensurado em uma escala de 1 a 5.

Figura 7 – Tabela de avaliações



Fonte: Autoria Própria

3.2 Metodologia

O desenvolvimento seguiu uma divisão de arquivos e páginas onde as páginas se encontram na raiz do projeto e os demais arquivos, responsáveis pela comunicação do sistema com o banco de dados, folhas de estilo, imagens do sistema e scripts fossem guardados em pastas deparadas. A página index.php é a responsável pelo redirecionamento dos usuários que acessam o sistema para a tela de login e foi a primeira a ser implementada, bem como as bases para cada página que o usuário deve ser redirecionado, de acordo com o cargo e departamento do qual faz parte.

Para um desenvolvimento estruturado em camadas, inicialmente o sistema foi projetado para que seguisse o padrão MVC (*Model-View-Controller*) de arquitetura, entretanto devido ao curto prazo para a entrega do sistema e a necessidade de revisar estudos do padrão para a implementação, o sistema se baseia no conceito de

controller do MVC, mas não possui um padrão arquitetura dentre o rol dos listados como recomendados para o desenvolvimento de software, tendo os arquivos de códigos das principais páginas de navegação e do sistema de avaliações na raiz do projeto, e os demais separados nas pastas solicitações, *controllers* e *assets*.

Na raiz do projeto encontram-se os arquivos de páginas iniciais dos diversos grupos de usuários, o formulário de avaliação, as páginas de agendamento, index.php, redefinição de senha e os CRUD's das demais tabelas. Na pasta controllers estão os arquivos que contém as funções que realizam as criações, inserções, alterações e exclusões no banco de dados.

A pasta *controllers* também contém os arquivos que recebem as requisições AJAX das páginas que possuem a atualização das informações sem a necessidade da atualização da página completa. No diretório *assets* são armazenados os arquivos de folhas de estilo, imagens utilizadas no sistema e *scripts* Javascript utilizados na funcionalidade de dividir um formulário em múltiplos passos, referente ao projeto de solicitações

3.2.1 Avaliações

Com a implementação do acesso funcionando, o primeiro projeto a ter o desenvolvimento iniciado foi o das Avaliações de Desempenho. De início, foram criadas as telas responsáveis pelos CRUD's das tabelas mais simples, de até no máximo 1 chave estrangeira. As tabelas cliente, cargo e compromisso possuem suas respectivas páginas para realização das quatro operações com a exibição dos dados de clientes já cadastrados, e seus respectivos responsáveis relacionados através de chave estrangeira. As tabelas cargo e compromisso não possuem chaves estrangeiras.

As páginas de agendamento e avaliações foram criadas na sequência, seguindo as necessidades de relacionar os dados das tabelas Avaliador, Avaliado, Cliente e Compromisso de forma que permitem um agendamento prévio a ser realizado antes da efetuação de uma avaliação através do formulário. Cada departamento possui um encarregado de criar os agendamentos de avaliações, que são baseados nos trabalhos de auditoria realizados semanalmente em diversos clientes.

Para a visualização das avaliações pendentes, feitas e recebidas, o sistema possui páginas que exibem para os usuários as próprias avaliações recebidas e feitas. Também é possível visualizar todas as avaliações pendentes dos demais usuários, devido a utilidade de verificar as agendas de trabalho das equipes antes das visitas acontecerem, uma vez que os agendamentos são cadastrados com antecedência.

Diretores possuem o privilégio de acessar todas as avaliações feitas do banco de dados, para isso possuem suas páginas com acesso único.

3.2.2 Solicitações

A criação das páginas voltadas para o controle das solicitações de trabalhos dos auditores externos deu-se em conjunto com a equipe responsável pela execução e monitoramento dos trabalhos. Foram utilizados os dados de tabelas previamente criadas para as avaliações, como a tabela avaliador, que é utilizada para persistir a informação do solicitante dos trabalhos.

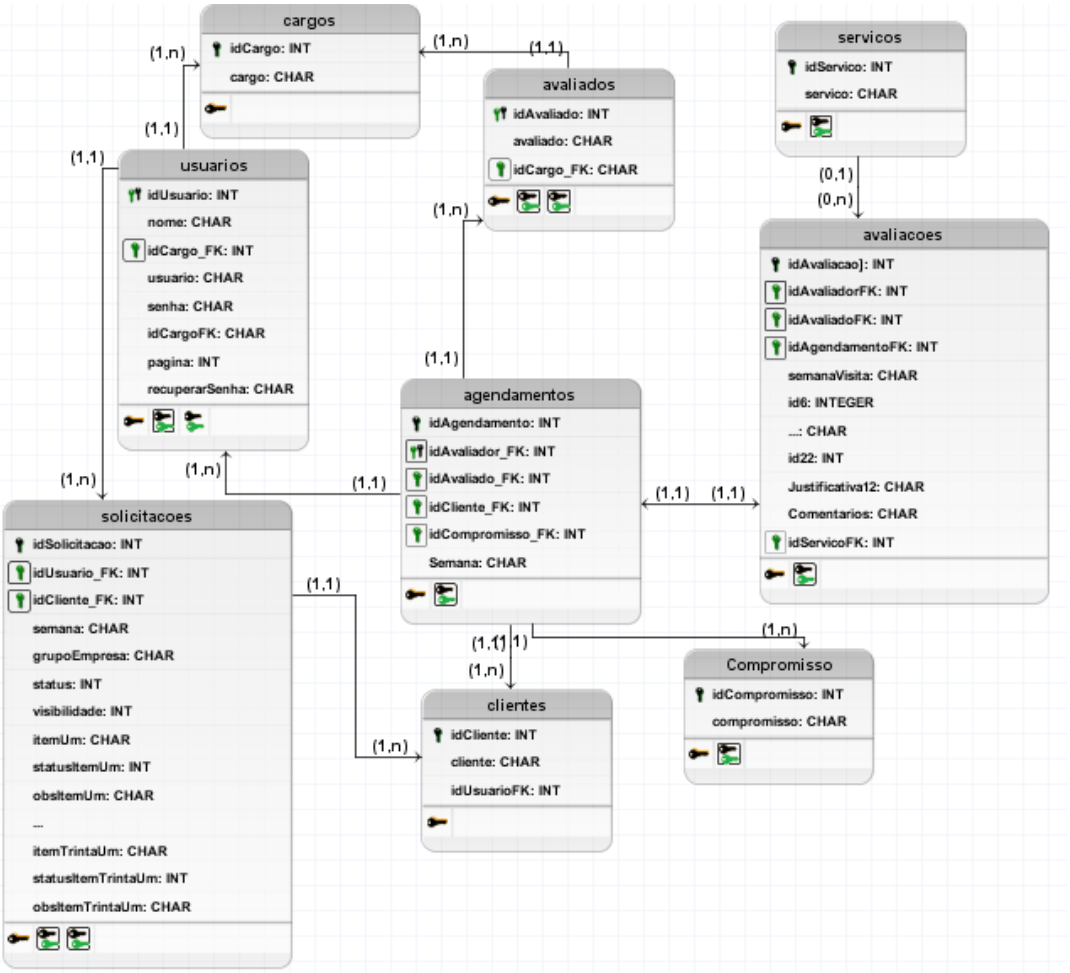
Ao receber as informações dos trabalhos solicitados, a equipe do Departamento Interno conta com uma página de controle de solicitações, onde sinaliza quais itens de cada solicitação estão completos, e o status geral da solicitação em específico. Além disso, é de poder dos membros do departamento interno alterar a visibilidade de uma solicitação nas páginas de exibição.

Para que os auditores externos consigam acompanhar a evolução do desenvolvimento dos trabalhos solicitados, foi criada uma página para a visualização dos status das solicitações, com a possibilidade de visualizar os status dos itens presentes em cada uma.

3.3 Diagrama de banco de dados

A criação do diagrama de banco de dados utilizado em produção foi concluída após os diversos testes realizados durante o desenvolvimento, e posteriormente com os envolvidos dos diferentes setores da empresa. Para que houvesse o mínimo possível de informações repetidas em diferentes tabelas, foram utilizadas chaves estrangeiras para que informações como clientes, avaliadores, avaliados, compromissos, cargos e serviços pudessem ser referenciadas pelas principais tabelas do banco. O diagrama pode ser conferido na figura 8.

Figura 8 – Diagrama de banco de dados



Fonte: Autoria Própria

3.4 Análise e implementação

As principais páginas do sistema, são o formulário de avaliação de desempenho, as páginas de agendamento e de controle de solicitações. Todas foram implementadas em PHP utilizando dos recursos disponibilizados pela extensão PDO e pelos scripts em Javascript adicionados para adicionar as funcionalidades necessárias.

A conexão com o banco de dados e as operações realizadas foram desenvolvidas utilizando PDO. Com sua utilização, as informações capturadas nos formulários são enviadas para os arquivos onde as funções responsáveis por criar, selecionar, atualizar e deletar os dados das respectivas tabelas são implementas.

Para a atualização das informações nas páginas, mediante as alterações nos *selects* e *radio buttons*, bem como para a verificação dos itens determinados como

required, foi utilizada a linguagem *Javascript*. Os scripts responsáveis pela verificação dos itens preenchidos foram armazenados em um arquivo separado. Para a atualização de parte das informações sem atualizar as páginas por completo, foi utilizado o recurso AJAX, com seus trechos de código implementados juntamente com os arquivos de *front-end* das páginas referidas.

3.5 Interfaces do sistema

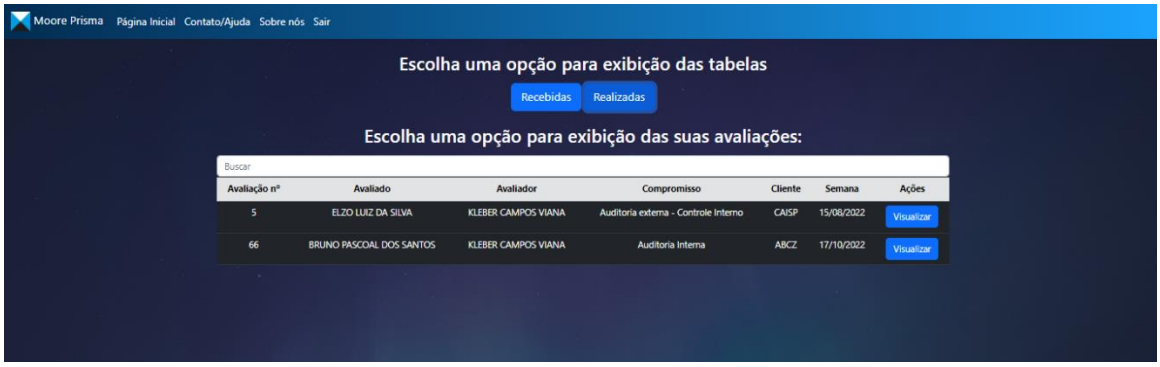
Os diferentes níveis de privilégios dos sistemas exigem a exibição de conteúdos específicos para cada grupo, e para os grupos com maiores privilégios de acesso (diretores, gerentes e encarregados) as informações são divididas em diferentes slides de um *carousel*, com finalidades dos itens apresentados devidamente identificadas. As imagens utilizadas, tons de cor dos botões, *selects* e demais itens seguem o padrão de cores da empresa, e suas escolhas foram devidamente acertadas com a equipe de comunicação da Moore Prisma, que também revisou todos os textos apresentados no sistema. Na figura 9 pode-se conferir um modelo página inicial.

Figura 9 – Modelo de página inicial do sistema

Fonte: Autoria Própria

As páginas onde são listadas avaliações agendadas e finalizadas, e as de visualização das solicitações possuem formatos de exibição em lista, com um botão para abrir a visualização completa da avaliação ou solicitação em questão, como pode ser visto na figura 10.

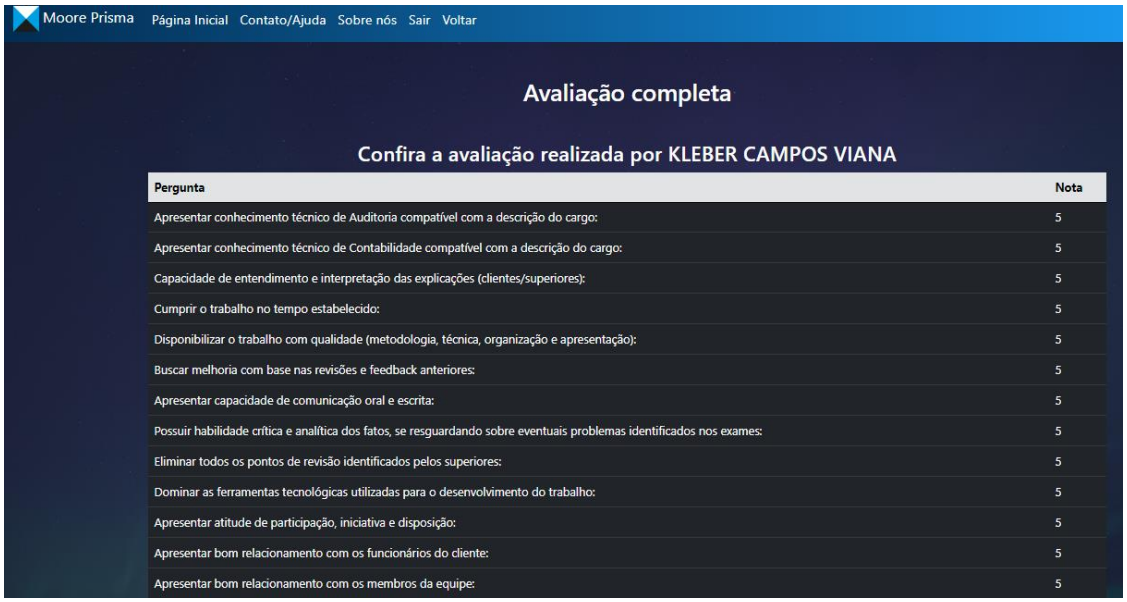
Figura 10 – Modelo de visualização prévia das avaliações



Fonte: Autoria Própria

Quando se é clicado no botão visualizar, os itens são exibidos, como apresentado na Figura 11.

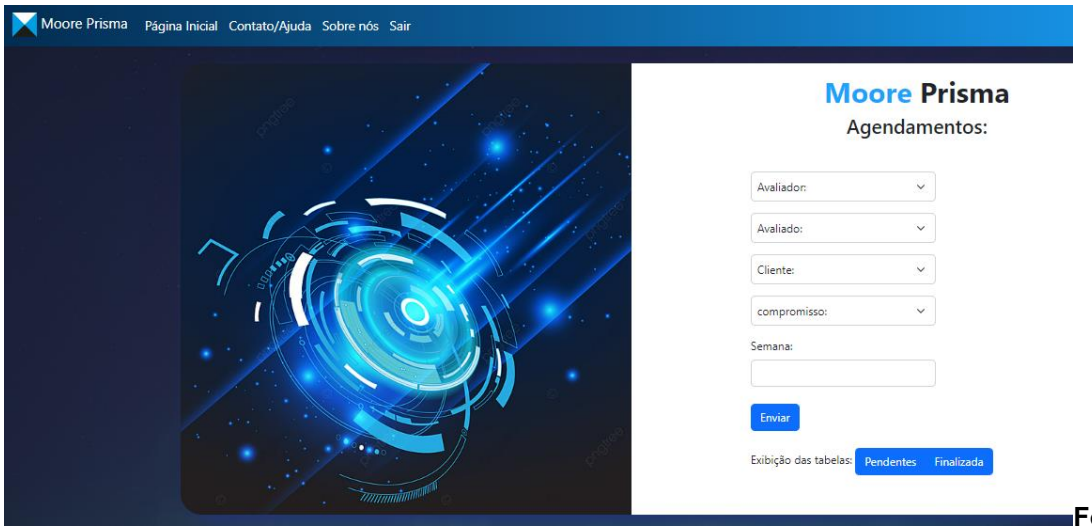
Figura 11 – Exibição do resultado uma avaliação



Fonte: Autoria Própria.

Os demais CRUD’s são realizados em páginas com os campos dispostos, de acordo com o modelo da Figura 12.

Figura 12 – Modelo de página que realiza as operações CRUD



Fonte: Autoria Própria.

3.6 Comunicação com o banco de dados

Como citado, para a execução das consultas e demais operações que o sistema exige, foi escolhida a extensão para acesso a banco de dados do PHP, o PDO (*PHP Data Object*). A extensão possui suporte para a comunicação com diversos bancos de dados, entre eles o MySQL, escolhido para o sistema. Na Figura 13 pode ser conferido o trecho de código em que é realizada a conexão com o banco de dados:

Figura 13 – Código para conexão ao banco de dados

```
7   // Cria a conexão com o banco de dados
8   try {
9       $conexao = new PDO("mysql:host=localhost;dbname=a******_avaliacoes", "a******_kviana", "***************");
10      $conexao->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
11      $conexao->exec("set names utf8");
12  } catch (PDOException $erro) {
13      echo "Erro na conexão:".$erro->getMessage();
14  }
```

Fonte: Autoria Própria.

A criação do objeto que é associado a variável $conexão é do tipo PDO, e representa a conexão com o banco e que será utilizada nos momentos em que for necessária a conexão com o banco. Como parâmetros, a criação do objeto $conexão contém uma fonte de dados onde são informados o driver do banco, o nome do host e o nome do banco de dados, um nome de usuário para conexão com o banco de dados e a senha para o usuário informado.

Nos arquivos *controllers*, que realizam as operações de comunicação entre o sistema web e o banco de dados, a primeira etapa executada é a verificação do envio

de dados via POST, e atribuindo *null* para as variáveis que vierem sem dados relacionados do front-end. Na Figura 14 pode ser observado um exemplo da verificação, do trecho extraído do arquivo controllerAgendamentos.php:

Figura 14 – Verificando se os dados foram enviados via POST

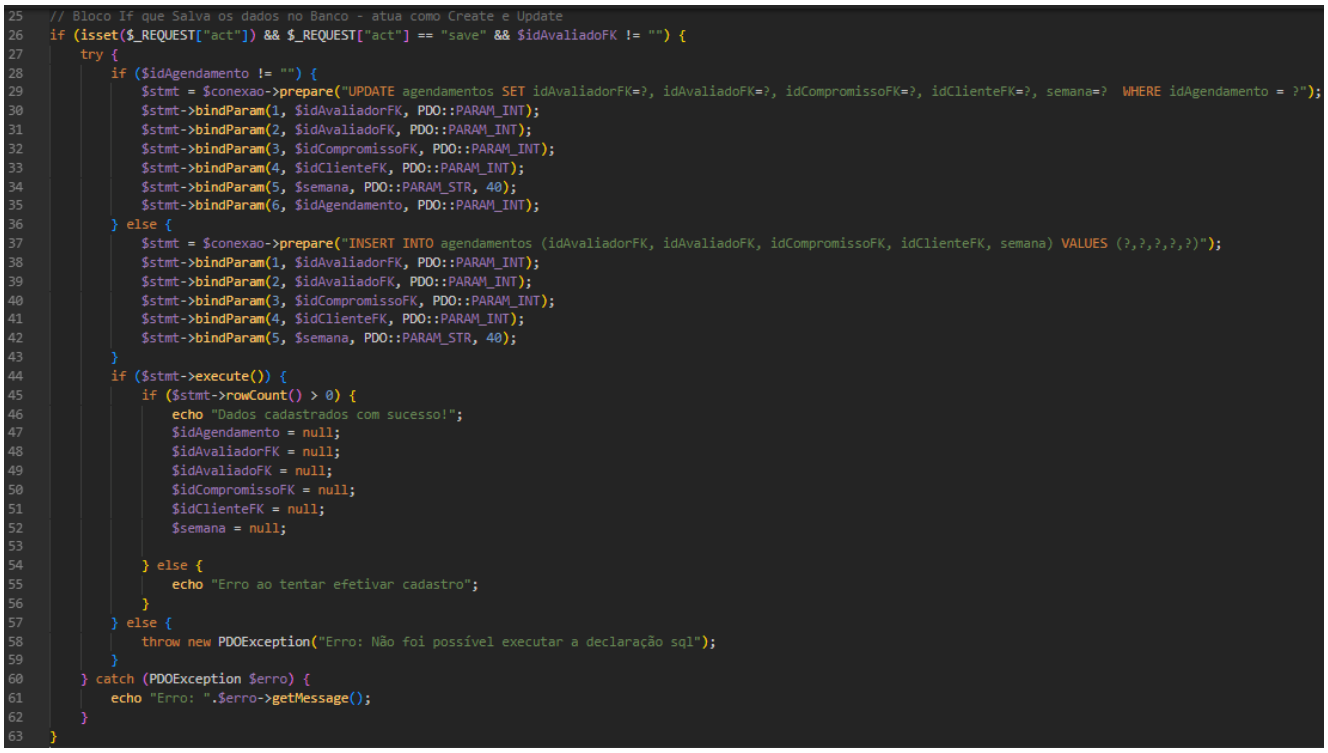
```php
// Verificar se foi enviando dados via POST
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $idAgendamento = (isset($_POST["idAgendamento"]) && $_POST["idAgendamento"] != null) ? $_POST["idAgendamento"] : "";
    $idAvaliadorFK = (isset($_POST["idAvaliadorFK"]) && $_POST["idAvaliadorFK"] != null) ? $_POST["idAvaliadorFK"] : "";
    $idAvaliadoFK = (isset($_POST["idAvaliadoFK"]) && $_POST["idAvaliadoFK"] != null) ? $_POST["idAvaliadoFK"] : "";
    $idCompromissoFK = (isset($_POST["idCompromissoFK"]) && $_POST["idCompromissoFK"] != null) ? $_POST["idCompromissoFK"] : "";
    $idClienteFK = (isset($_POST["idClienteFK"]) && $_POST["idClienteFK"] != null) ? $_POST["idClienteFK"] : "";
    $semana = (isset($_POST["semana"]) && $_POST["semana"] != null) ? $_POST["semana"] : "";


} else if (!isset($idAgendamento)) {
    // Se não se não foi setado nenhum valor para variável $idAgendamento
    $idAgendamento = (isset($_GET["idAgendamento"]) && $_GET["idAgendamento"] != null) ? $_GET["idAgendamento"] : "";
    $idAvaliadorFK = NULL;
    $idAvaliadoFK = NULL;
    $idCompromissoFK = NULL;
    $idClienteFK= NULL;
    $semana= NULL;
}
```

Fonte: Autoria Própria.

As informações do formulário que envia as informações para o controllerAgendamentos.php são persistidas através de inicialmente um *try-catch* criado para a conexão. Com os dados recebidos, o sistema possui um bloco que realiza as operações de create e update no banco. Para isso, existe uma instrução *if* para descobrir se há uma ação *act* para salvar *save*, caso houver então o bloco direciona a um novo *try-catch*, que utiliza o objeto de conexão da variável $conexao. A estrutura montada para criação e atualização dos dados pode ser conferida na Figura 15.

Figura 15 – Estrutura responsável pela criação e atualização dos dados



```php
// Bloco If que Salva os dados no Banco - atua como Create e Update
if (isset($_REQUEST["act"]) && $_REQUEST["act"] == "save" && $idAvaliadoFK != "") {
    try {
        if ($idAgendamento != "") {
            $stmt = $conexao->prepare("UPDATE agendamentos SET idAvaliadorFK=?, idAvaliadoFK=?, idCompromissoFK=?, idClienteFK=?, semana=?  WHERE idAgendamento = ?");
            $stmt->bindParam(1, $idAvaliadorFK, PDO::PARAM_INT);
            $stmt->bindParam(2, $idAvaliadoFK, PDO::PARAM_INT);
            $stmt->bindParam(3, $idCompromissoFK, PDO::PARAM_INT);
            $stmt->bindParam(4, $idClienteFK, PDO::PARAM_INT);
            $stmt->bindParam(5, $semana, PDO::PARAM_STR, 40);
            $stmt->bindParam(6, $idAgendamento, PDO::PARAM_INT);
        } else {
            $stmt = $conexao->prepare("INSERT INTO agendamentos (idAvaliadorFK, idAvaliadoFK, idCompromissoFK, idClienteFK, semana) VALUES (?,?,?,?,?)");
            $stmt->bindParam(1, $idAvaliadorFK, PDO::PARAM_INT);
            $stmt->bindParam(2, $idAvaliadoFK, PDO::PARAM_INT);
            $stmt->bindParam(3, $idCompromissoFK, PDO::PARAM_INT);
            $stmt->bindParam(4, $idClienteFK, PDO::PARAM_INT);
            $stmt->bindParam(5, $semana, PDO::PARAM_STR, 40);
        }
        if ($stmt->execute()) {
            if ($stmt->rowCount() > 0) {
                echo "Dados cadastrados com sucesso!";
                $idAgendamento = null;
                $idAvaliadorFK = null;
                $idAvaliadoFK = null;
                $idCompromissoFK = null;
                $idClienteFK = null;
                $semana = null;

            } else {
                echo "Erro ao tentar efetivar cadastro";
            }
        } else {
            throw new PDOException("Erro: Não foi possível executar a declaração sql");
        }
    } catch (PDOException $erro) {
        echo "Erro: ".$erro->getMessage();
    }
}
```

Fonte: Autoria Própria.

Para que os dados sejam atualizados nos formulários e nas páginas de consulta sem que seja necessária a atualização completa da página, foi escolhido o AJAX, que é um conjunto de tecnologias de desenvolvimento que permite que aplicativos funcionem de maneira síncrona, processando solicitações ao servidor em segundo plano. A interatividade necessária é para que, por exemplo, na página de avaliações só apareçam como opções para a escolha de avaliado aquele que estiver previamente agendado para um cliente. Caso haja a atualização do nome do cliente, ou do avaliador, a lista de avaliados é atualizada automaticamente.

Na Figura 16 pode ser conferido o trecho de código em Javascript que permite a utilização do AJAX, presente no arquivo avaliacao.php:

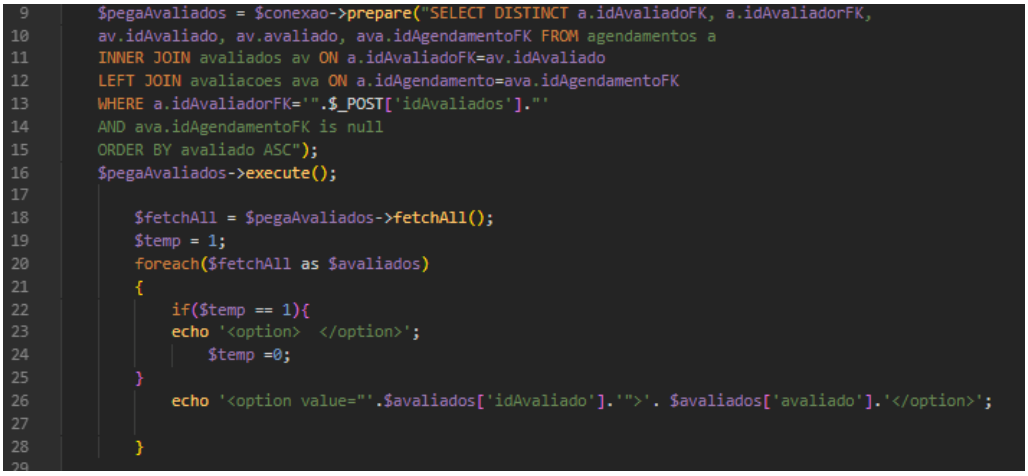
Figura 16 – Requisição AJAX

```
826     $("#idAvaliadorFK").on("change", function() {
827         $.ajax({
828             url: 'https://auditoriamooreprisma.com.br/controllers/pegaValores.php',
829             type: 'POST',
830             data: {
831                 idAvaliados: $("#idAvaliadorFK").val()
832             },
833             beforeSend: function() {
834       //...
835                 $("#labelAgendamento").css({
836                     'display': 'none'
837                 });
838             },
839             success: function(data) {
840                 $("#idAvaliadoFK").css({
841                     'display': 'block'
842   //...
843                 });
844             },
845             error: function(data) {
846                 $("#idAvaliadoFK").css({
847                     'display': 'block'
848                 });
849                 $("#idAvaliadoFK").html("Houve um erro ao carregar");
850             }
851         });
852     });
```

Fonte: Autoria Própria.

Na Figura 17 está a parte complementar da requisição, o trecho de consulta ao banco que é realizada em segundo plano, presente no arquivo pegaValores.php.

Figura 17 – Código que realiza a consulta ao banco em segundo plano



```
9     $pegaAvaliados = $conexao->prepare("SELECT DISTINCT a.idAvaliadoFK, a.idAvaliadorFK,
10    av.idAvaliado, av.avaliado, ava.idAgendamentoFK FROM agendamentos a
11    INNER JOIN avaliados av ON a.idAvaliadoFK=av.idAvaliado
12    LEFT JOIN avaliacoes ava ON a.idAgendamento=ava.idAgendamentoFK
13    WHERE a.idAvaliadorFK='".$_POST['idAvaliados']."'
14    AND ava.idAgendamentoFK is null
15    ORDER BY avaliado ASC");
16    $pegaAvaliados->execute();
17
18        $fetchAll = $pegaAvaliados->fetchAll();
19        $temp = 1;
20        foreach($fetchAll as $avaliados)
21        {
22            if($temp == 1){
23            echo '<option>  </option>';
24                $temp =0;
25        }
26            echo '<option value="'.$avaliados['idAvaliado'].'">'. $avaliados['avaliado'].'</option>';
27
28        }
```

Fonte: Autoria Própria.

4 RESULTADOS E DISCUSSÕES

As composições dos departamentos atendidos definiram as regras iniciais para o desenvolvimento do projeto. Os colaboradores envolvidos são os assistentes, sênior, encarregados, gerentes e diretores dos departamentos: Departamento Interno, Auditoria Externa, Auditoria Externa e Auditoria de Serviços Financeiros. Para que o modelo de projeto escolhido pudesse atender as demandas de todos os departamentos envolvidos, um responsável de cada departamento esteve acompanhando o desenvolvimento do sistema e validando o funcionamento correto das funcionalidades requeridas.

Para os integrantes do Departamento Interno em específico, o sistema atende a demanda de solicitações de trabalhos, feitas pelos auditores do departamento de Auditoria Externa, que trabalham em campo nas diversas cidades onde os clientes da Moore Prisma possuem seus empreendimentos. O controle permite que eles recebam as solicitações através do formulário que é preenchido pelos auditores da auditoria externa e atualizem os status de cada item das solicitações para o acompanhamento dos auditores solicitantes, bem como os status de progresso geral das atividades solicitadas.

Todos os departamentos utilizam da outra demanda que fora solicitada em sua implementação, que é o agendamento e realização das avaliações de desempenho dos membros de todas as equipes de trabalho, com a realização da avaliação sendo feita pelo responsável pelo trabalho de auditoria aos membros de suas equipes, e as autoavaliações de cada colaborador. O sistema permite que sejam visualizadas as avaliações pendentes e as finalizadas, através de consultas ao banco de dados.

5 CONCLUSÃO

O sistema atendeu as necessidades elencadas pelos diretores no momento da solicitação do projeto. Dentre as funcionalidades apresentadas, algumas não haviam sido citadas previamente, mas ao longo do desenvolvimento observou-se que são indispensáveis para as equipes.

5.1 Limitações

O formato de controle das solicitações apresentado para os gestores do Departamento Interno possui uma usabilidade limitada, de maneira que com poucas alterações são necessárias várias operações de busca no banco.

O sistema de *login* atende o esperado quanto ao formato escolhido para recuperação e criação de senha, porém o modelo atual não permite que o usuário crie seu perfil inserindo suas informações como cargo, nome completo e clientes dos quais possivelmente sejam encarregados.

5.2 Atualizações e trabalhos futuros

Dentre as atualizações necessárias no sistema, a principal consiste na atualização dos códigos e arquivos para que sigam uma arquitetura de software que possibilite um melhor desempenho, através de melhores práticas que também tragam melhor manutenibilidade na estrutura do *front-end* e do *back-end*. Para isso, uma das medidas a serem tomadas é a utilização de um framework PHP como Laravel, Symfony, dentre outros.

Com uma estrutura definida para o projeto, outro ponto a ser evoluído é a experiência do usuário, adicionando elementos como, por exemplo, modais, que tragam maior facilidade de interpretação das informações pelo usuário.

REFERÊNCIAS

MORIN, J. de. **Des technologies, des marchés et des hommes:** pratiques et perspectivas du management des ressources technologiques. Paris: Les Éditions d'Organisation, 1992.

COSCODAI, V. Artigo Ibracon: **Ética em tempos de tecnologia,** Rio de Janeiro. Disponível em: <http://www.http://www.ibracon.com.br/ibracon/Portugues/detNoticia.php?cod=8762/>. Acesso em: 28 nov. 2022.

PHP, Documentation Group. PHP: **PHP Manual.** Disponível em: <https://www.php.net/manual/en/index.php/>. Acesso em: 28 nov. 2022.

MARQUES, R. Homehost: **O que é HTML? Entenda de forma descomplicada.** Disponível em: <https://www.homehost.com.br/blog/tutoriais/o-que-e-html/>. Acesso em: 28 nov. 2022.

AWS, Amazon Web Services. AWS: **O que é Javascript?** Disponível em: <https://aws.amazon.com/pt/what-is/javascript/>. Acesso em: 28 nov. 2022.

GONÇALVES, A. Hostinger: **O que é CSS? Guia Básico para Iniciantes.** Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-css-guia-basico-de-css/>. Acesso em: 28 nov. 2022.

LONGEN, A. S. Weblink: **O Que é MySQL – Guia para Iniciantes.** Disponível em: <https://www.weblink.com.br/blog/o-que-e-mysql/>. Acesso em: 28 nov. 2022.

SMITH, S. Learn Microsoft: **Padrão MVC.** Disponível em: <https://learn.microsoft.com/pt-br/aspnet/core/mvc/>. Acesso em: 28 nov. 2022.

PHP, Documentation Group. PHP: **PDO Introdução.** Disponível em: <https://www.php.net/manual/pt_BR/intro.pdo.php/>. Acesso em: 28 nov. 2022.

GONÇALVES, A. Hostinger: **O Que é AJAX e Como Funciona.** Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-ajax/>. Acesso em: 28 nov. 2022.

NOLETO, C. BeTrybe: **PDO(PHP Data Objects): o que é, como funciona e como usar?** Disponível em: < https://blog.betrybe.com/php/pdo-php/>. Acesso em: 28 nov. 2022.

JQUERY, OpenJS Foundation. JQuery: **What is jQuery?** Disponível em: <https://jquery.com/>. Acesso em: 28 nov. 2022.

NOLETO, C. BeTrybe: **MVC PHP: o guia inicial dessa arquitetura de desenvolvimento.** Disponível em: < https://blog.betrybe.com/php/mvc-php/>. Acesso em: 28 nov. 2022.

CODE, Visual Studio, R. Visual Studio Code: **Getting Started.** Disponível em: <https://code.visualstudio.com/docs/>. Acesso em: 28 nov. 2022.

RATSCHILLER, T. The phpMyAdmin: **phpMyAdmin Introdução.** Disponível em: <https://docs.phpmyadmin.net/pt_BR/latest/intro.html/>. Acesso em: 28 nov. 2022.

LONGEN, A. S. Hostinger: **O Que é cPanel? Prós e Contras + Como Usá-lo.** Disponível em: < https://www.hostinger.com.br/tutoriais/cpanel-o-que-e-painel-de-hospedagem/>. Acesso em: 28 nov. 2022.

NETO, M. B. Repositório UFSC: **brModeloWeb: Ferramenta Web para Ensino e Modelagem de Banco de Dados.** Disponível em: <https://repositorio.ufsc.br/xmlui/handle/123456789/171508/>. Acesso em: 28 nov. 2022.

Presidência da República
Casa Civil
Subchefia para Assuntos Jurídicos

LEI Nº 9.610, DE 19 DE FEVEREIRO DE 1998.

Mensagem de veto

Altera, atualiza e consolida a legislação sobre direitos autorais e dá outras providências.

O PRESIDENTE DA REPÚBLICA Faço saber que o Congresso Nacional decreta e eu sanciono a seguinte Lei:

Título I

Disposições Preliminares

Art. 1º Esta Lei regula os direitos autorais, entendendo-se sob esta denominação os direitos de autor e os que lhes são conexos.

Art. 2º Os estrangeiros domiciliados no exterior gozarão da proteção assegurada nos acordos, convenções e tratados em vigor no Brasil.

Parágrafo único. Aplica-se o disposto nesta Lei aos nacionais ou pessoas domiciliadas em país que assegure aos brasileiros ou pessoas domiciliadas no Brasil a reciprocidade na proteção aos direitos autorais ou equivalentes.

Art. 3º Os direitos autorais reputam-se, para os efeitos legais, bens móveis.

Art. 4º Interpretam-se restritivamente os negócios jurídicos sobre os direitos autorais.

Art. 5º Para os efeitos desta Lei, considera-se:

I - publicação - o oferecimento de obra literária, artística ou científica ao conhecimento do público, com o consentimento do autor, ou de qualquer outro titular de direito de autor, por qualquer forma ou processo;

II - transmissão ou emissão - a difusão de sons ou de sons e imagens, por meio de ondas radioelétricas; sinais de satélite; fio, cabo ou outro condutor; meios óticos ou qualquer outro processo eletromagnético;

III - retransmissão - a emissão simultânea da transmissão de uma empresa por outra;

IV - distribuição - a colocação à disposição do público do original ou cópia de obras literárias, artísticas ou científicas, interpretações ou execuções fixadas e fonogramas, mediante a venda, locação ou qualquer outra forma de transferência de propriedade ou posse;

V - comunicação ao público - ato mediante o qual a obra é colocada ao alcance do público, por qualquer meio ou procedimento e que não consista na distribuição de exemplares;

VI - reprodução - a cópia de um ou vários exemplares de uma obra literária, artística ou científica ou de um fonograma, de qualquer forma tangível, incluindo qualquer armazenamento permanente ou temporário por meios eletrônicos ou qualquer outro meio de fixação que venha a ser desenvolvido;

VII - contrafação - a reprodução não autorizada;

VIII - obra:

a) em co-autoria - quando é criada em comum, por dois ou mais autores;

b) anônima - quando não se indica o nome do autor, por sua vontade ou por ser desconhecido;

c) pseudônima - quando o autor se oculta sob nome suposto;

d) inédita - a que não haja sido objeto de publicação;

e) póstuma - a que se publique após a morte do autor;

f) originária - a criação primígena;

g) derivada - a que, constituindo criação intelectual nova, resulta da transformação de obra originária;

h) coletiva - a criada por iniciativa, organização e responsabilidade de uma pessoa física ou jurídica, que a publica sob seu nome ou marca e que é constituída pela participação de diferentes autores, cujas contribuições se fundem numa criação autônoma;

i) audiovisual - a que resulta da fixação de imagens com ou sem som, que tenha a finalidade de criar, por meio de sua reprodução, a impressão de movimento, independentemente dos processos de sua captação, do suporte usado inicial ou posteriormente para fixá-lo, bem como dos meios utilizados para sua veiculação;

IX - fonograma - toda fixação de sons de uma execução ou interpretação ou de outros sons, ou de uma representação de sons que não seja uma fixação incluída em uma obra audiovisual;

X - editor - a pessoa física ou jurídica à qual se atribui o direito exclusivo de reprodução da obra e o dever de divulgá-la, nos limites previstos no contrato de edição;

XI - produtor - a pessoa física ou jurídica que toma a iniciativa e tem a responsabilidade econômica da primeira fixação do fonograma ou da obra audiovisual, qualquer que seja a natureza do suporte utilizado*;*

XII - radiodifusão - a transmissão sem fio, inclusive por satélites, de sons ou imagens e sons ou das representações desses, para recepção ao público e a transmissão de sinais codificados, quando os meios de decodificação sejam oferecidos ao público pelo organismo de radiodifusão ou com seu consentimento;

XIII - artistas intérpretes ou executantes - todos os atores, cantores, músicos, bailarinos ou outras pessoas que representem um papel, cantem, recitem, declamem, interpretem ou executem em qualquer forma obras literárias ou artísticas ou expressões do folclore.

Art. 6º Não serão de domínio da União, dos Estados, do Distrito Federal ou dos Municípios as obras por eles simplesmente subvencionadas.