



**INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**
Goiás

Bacharelado em Sistemas de Informação

Patrick Cavalcante Moraes

SAD - SISTEMA PARA AVALIAÇÃO DE DESEMPENHO DOCENTE DO INSTITUTO FEDERAL DE GOIÁS

Luziânia
2022

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Bacharelado em Sistemas de Informação

Patrick Cavalcante Moraes

**SAD - SISTEMA PARA AVALIAÇÃO DE DESEMPENHO DOCENTE DO
INSTITUTO FEDERAL DE GOIÁS**

Projeto apresentado ao Curso de Bacharelado em Sistemas de Informação como um dos pré-requisitos de matrícula na Disciplina de Trabalho de Conclusão de Curso.

Orientador: Wendell Bento Geraldes

Luziânia
2022

SUMÁRIO

1	INTRODUÇÃO	3
1.1	Problema	3
1.2	Justificativa	3
1.3	Objetivo	4
1.3.1	Objetivos Gerais	4
1.3.2	Objetivos Específicos	4
2	FUNDAMENTAÇÃO TEÓRICA	6
2.1	Avaliação Docente do Instituto Federal de Goiás	6
2.2	Gerenciamento de Requisitos de Software	6
2.3	Metodologia Agil	7
2.3.1	Scrum	8
2.3.2	Kamban	8
2.4	API RESTful com Java 8 e Spring Boot	9
2.5	Front-end com Angular 12	12
2.6	Banco de Dados com PHPMyAdmin	18
2.7	Abordagens de Testes de Usabilidade e Desempenho do Sistema	20
3	HIPÓTESE	21
4	METODOLOGIA	22
5	RECURSOS NECESSÁRIOS	23
6	CRONOGRAMA	24
	REFERÊNCIAS	26

1 INTRODUÇÃO

Neste trabalho descreve-se as diversas etapas do desenvolvimento de um Sistema de Avaliação Docente (SAD) para o Instituto Federal de Goiás (IFG) e discute-se sua aplicabilidade a outras organizações. Observamos a necessidade da instituição em um sistema de avaliação docente, onde foi levantado que o modelo aplicado já não agregaria resultado diante o cenário atual. Sendo que atualmente já possui uma grande quantidade de dados e com o apoio desse novo *software* assumiria uma integridade no controle desses dados gerados.

Ao desenvolver um SAD, conforme relatado neste trabalho procura-se construir um sistema que contemple as recomendações da instituição, desde as ferramentas já utilizadas, passando pela escolha das tecnologias de desenvolvimento, a preparação do sistema e a adequada da utilização dos resultados, até os métodos de implantação a ações de manutenção.

Serão também abordadas as metodologias ágeis aplicadas na criação e entrega do produto, de forma que atendam o conjunto de práticas eficazes, que se destinam a produzir o sistema demonstrando uma qualidade em seus processos, em uma forma de gerenciar o projeto sendo ele adaptável às mudanças.

Com base nisso, este projeto de pesquisa tem como objetivo apresentar a necessidade de um SAD para o apoio das demandas de avaliação de desempenho docente da Comissão Permanente de Pessoal Docente (CPPD) e entregar uma melhoria importante para a organização, podendo assim facilitar o processo de progressão dos docentes da instituição.

1.1 Problema

A inexistência de um sistema estruturado para avaliação de desempenhos, somado a alta demanda, resulta em um grande trabalho manual dos professores responsáveis, pertencentes a CPPD, o projeto apoiará o processo de avaliação dos docentes, coordenadores de curso, coordenadores acadêmicos ou chefe de departamento.

Nesse cenário, este trabalho visa atender a demanda da CPPD, e também mostra que é possível desenvolver um SAD, que seja capaz de gerir as informações e organizar o processo. O sistema será responsável pela condução do processo de avaliação docente para a efetivação da progressão funcional dos mesmos.

À vista disso, surge a necessidade da criação do *software*, onde estas avaliações serão armazenadas e servirão como histórico para esses colaboradores, e também servirão como melhoria e estruturação deste processo, tirando esses registros dos formulários *on-line* que atualmente consistem em um sistema externo, não vinculado a instituição.

Como auxiliar a equipe da CPPD na otimização de visualização e processamento dos dados da avaliação docente do Instituto Federal de Goiás?

1.2 Justificativa

De acordo [XAVIER P. PIRES \(2005\)](#) as Instituições de Ensino Superior, de um modo geral, vem sendo alvo de inúmeras questões sobre sua atuação no contexto social, e a ausência de subsídios que apresentem respostas concretas às questões constantes tem provocado o descrédito quanto à responsabi-

lidade social. Desta forma, surge no seu bojo uma latente questão: As Instituições de Ensino Superior vem atendendo à demanda e expectativas da sociedade brasileira, enquanto entidade responsável pela disseminação do conhecimento?

Conforme abordado por OLIVEIRA-CASTRO G. LIMA (1996) os sistemas de avaliação devem ser justos e imparciais, baseados em padrões de desempenho atingíveis, objetivos e claros, apoiados na realidade dos cargos ou postos de trabalho. Para tal, é necessário pesquisar os padrões desejáveis de desempenho junto aos ocupantes dos cargos e às respectivas chefias.

Implantar um sistema de avaliação visa melhorar alguns pilares importantes para uma organização, podendo colocar como os principais: planejamento, produtividade, ética, desenvolvimento de carreira, excelência, melhoria no processo de trabalho, responsabilidade e comprometimento.

Portanto, em suma, é importante destacar que as instituições estão cientes de seu valor no processo de desenvolvimento e crescimento institucional, considerando que os profissionais estão sendo exigidos cada vez mais se capacitar para o mercado. Logo a avaliação de desempenho apresenta-se como instrumento e ação capaz de sinalizar o desempenho e detectar alterações entre o planejado e o que está sendo executado, oferecendo, desta forma, subsídios para correção.

Diante a este cenário, o IFG não apresenta nenhuma aplicação para a automatização dos processos de avaliação docentes. A CPPD possui uma necessidade de um *software* que seja capaz de efetuar as avaliações dos docentes pertencentes a instituição, sendo ele apto de gerir as informações e consiga atender os requisitos conforme os parágrafos anteriores. O sistema visará entregar um processo integrado que seja preparado para criar e gerir as avaliações de desempenho para fins de progressão e promoção funcional.

1.3 Objetivo

Esta seção tem como finalidade apresentar os objetivos que precisam ser alcançados para este projeto.

1.3.1 Objetivos Gerais

O objetivo deste *software* é melhorar o processo de avaliação de desempenho docente, contribuindo com a melhoria no processo de avaliação docente e posterior progressão funcional. Essa abordagem irá ajudar a descobrir os pontos fortes e fracos, além do fato de que é uma forma importante para que os docentes entendam o processo de avaliação no trabalho. O sistema tem como intuito fornecer informações para CPPD, subsidiando assim as ações da comissão.

1.3.2 Objetivos Específicos

Com a finalidade de atingir o objetivo geral, este projeto tem os seguintes objetivos específicos:

- Fazer o levantamento de requisitos;
- Modelar os requisitos na forma de diagramas;
- Elaborar o projeto de software;
- Implementar o software;
- Apresentar o protótipo a CPPD;

-
- Validar o protótipo;
 - Efetuar os testes;
 - Avaliação do *software* junto a CPPD;

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Avaliação Docente do Instituto Federal de Goiás

O Instituto Federal de Goiás dispõe da Comissão Permanente de Pessoal Docente, como responsável pelas avaliações de desempenho para fins de progressão e promoção funcional, formada por 9 membros pertencentes ao quadro de servidores do IFG (Atualizado em 19 de agosto de 2021). Atualmente a CPPD é responsável por cerca de 1300 docentes que são avaliados periodicamente (GOIAS, 2021).

A solução será importante para o processo de avaliação docente, pois essas avaliações impactam diretamente na progressão de carreira do docente. No cenário atual o SAD será necessário na automatização do processo diminuindo a margem de erros, pois atualmente as avaliações são feitas em sistemas distintos e vinculados a uma planilha, tornando-se um procedimento manual passível a falhas.

Logo ao gerar os dados no SAD existirá um processo de armazenamento centralizado de dados, uma forma de consulta simples, pois estará disponível para visualização em um sistema Web, sendo um formato mais atualizado. Atualmente quando a necessidade por diversos motivos, a consulta de uma nota atribuída anteriormente, é exigido um trabalho maior, pois é necessário a solicitação para a CPPD da planilha que está armazenada a nota. Este processo requer tempo, e um trabalho manual dos membros da comissão para que consigam disponibilizar essa nota.

A solução atenderá a princípio toda a organização do IFG que contempla 14 campus, que atualmente possui como método a Ficha de Avaliação de Desempenho. A avaliação é feita em dois períodos ao ano, de acordo com a Res. Consup IFG nº 040/2018, Regimento Geral do IFG, Art. 190-193, 198-201, avaliando a atuação dos Docente, Coordenadores de Curso, Coordenador Acadêmico e Chefe de Departamento, atribuindo-lhe nota numa escala de 0(zero) à 10(dez). São feitas três avaliações a avaliação do aluno, a avaliação do chefe departamento ou coordenador de curso e a auto avaliação, gerando uma média aritmética com a nota do docente, que é necessário a soma das três notas e dividivisão pela soma das quantidades, um processo simples, porém a aplicação se torna bastante necessário e modificará o processo para melhor.

2.2 Gerenciamento de Requisitos de Software

Conforme reuniões com os atuais membros pertencentes da comissão da CPPD, foram levantados os pontos necessários para o projeto do sistema para avaliação de desempenho docente. Serão realizadas reuniões para se possa ser feito as interações e troca de informações que poderão ser importantes para as decisões do projeto em questão. A abordagem visará que a CPPD colabore durante a etapa de análise de requisitos, mantendo todos alinhados com a solução, sendo uma maneira fácil para adoção, garantido a organização e eficiência.

Na sequência foram elaborados alguns diagramas, a fim de descrever e modelar o entendimento do sistema e facilitar o desenvolvimento. Pode-se visualizar no diagrama de caso de uso, os atores representados, identificando os usuários do sistema, que diferenciam seus papéis e níveis de acesso, servindo como um unificador em todo o desenvolvimento do sistema.

Segundo CENTENARO (2014) o diagrama de caso de uso documenta o que o sistema faz da perspectiva do usuário. Em outras palavras, descreve as principais funções do sistema e as interações dessas funções com os usuários. Neste diagrama, não será aprofundando nos detalhes técnicos, informando

como o sistema funciona.

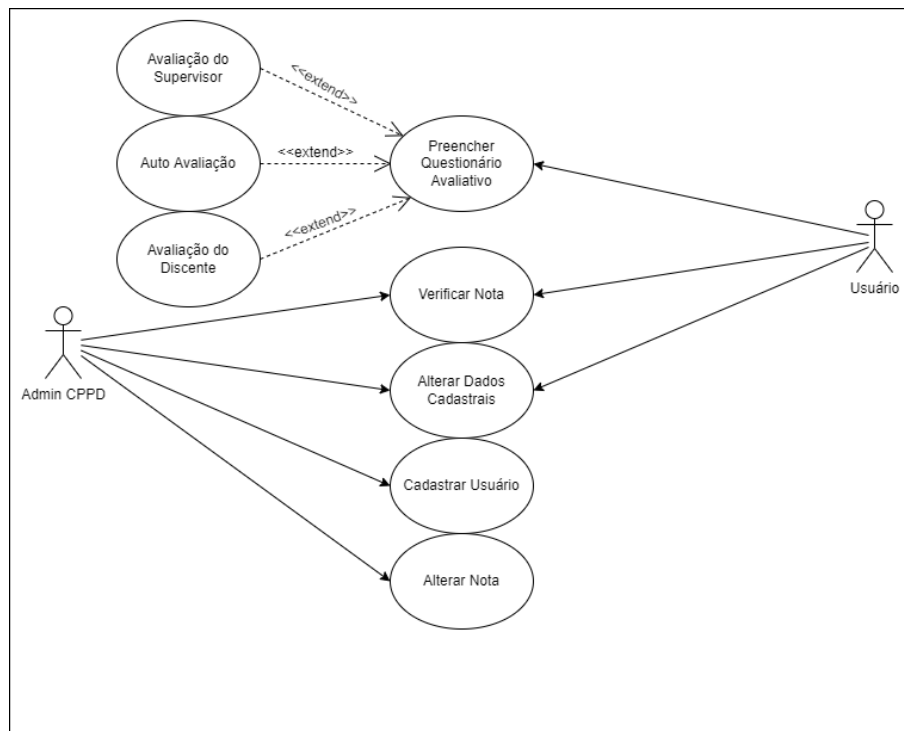


Figura 1 – Diagrama de Caso de Uso

Baseado no que está sendo demonstrado no diagrama de caso de uso acima, o sistema possui funções que atenderam o cadastro de usuário, diferenciando seus papéis. As ações apresentadas pelos caso de uso demonstra como está representado cada função do sistema. As extensões atribuídas no caso de uso Preencher Questionário Avaliativo, consiste no comportamento que pode variar de acordo com o perfil do usuário.

2.3 Metodologia Ágil

Baseando-se na metodologia ágil, no *framework Scrum*, onde as tarefas são reunidas em um *backlog*, e o projeto é dividido em blocos fixos de tempo com suas próprias tarefas e metas, ou seja, os sprints, serão divididas as etapas de desenvolvimento do projeto. Outro ponto bastante utilizado da metodologia ágil aplicada, onde será suficiente apenas ter um *backlog* com as entregas pendentes do projeto, uma forma simples, pois o foco será em antecipar o máximo possível o trabalho (UCHOA, 2022).

Durante o desenvolvimento do produto, serão definidas as etapas divididas em *sprint*, conforme abordado por CRONAPP (2021) “Um *sprint backlog* é um tempo predeterminado que define o ciclo de desenvolvimento de um *software*”. Cada *sprint* precisará ser validado, a parti da validação será iniciado uma novo *backlog* do sistema.

Também está sendo utilizado no desenvolvimento do sistema o *Time Box* do *Scrum*, na tradução literal significa “caixa de tempo”. Ou seja, é ter o tempo, para fazer um trabalho, limitado, executando da melhor forma que puder nessa janela de tempo. É uma técnica simples usada no desenvolvimento de software para rastrear o progresso e para, simplesmente, ter o trabalho feito e obter uma entrega contínua.

2.3.1 Scrum

Segundo [Drumond \(2022\)](#) o *Scrum* é um método ágil para gestão de projetos. Ele é muito utilizado em equipes de desenvolvimento de software porque reúne um conjunto de boas práticas que facilitam o trabalho em equipes dessa natureza, como reuniões periódicas, lista de requisitos a serem atendidos, *feedbacks* constantes sobre o produto, entre outros.

O *Scrum* é extremamente prescritivo, ou seja, para que um projeto dê certo dentro desse método é preciso seguir suas principais recomendações à risca, já que o *Scrum* define desde os papéis dentro da equipe de trabalho até a duração ideal das reuniões ([ESPINHA, 2010](#)).

No *Scrum* exige que seja identificada uma lista de funcionalidades que precisam ser desenvolvidas para que o produto chegue ao objetivo esperado, o chamado *Product Backlog*. Essa lista é traduzida em *Sprints*: ciclos de tempo onde pequenas partes do produto são planejadas, executadas e entregues, e é justamente aí que o *kanban* pode entrar como um facilitador.

2.3.2 Kamban

De acordo com [Digite \(2022\)](#) o *Kanban* é um sistema de gestão visual para controle de tarefas e fluxos de trabalho através da utilização de colunas e cartões, facilitando a gestão de atividades. Sendo muito comum confundir o *kanban* com o *Scrum*, ou achar que o *Scrum* necessariamente precisa do *kanban* para funcionar, mas não é bem assim.

O *kanban* é um sistema, ou seja, uma ferramenta para auxiliar no trabalho da equipe, como uma linguagem de programação, por exemplo. Ele não é prescritivo ou impõe regras para que o trabalho seja feito corretamente, apenas possibilita que o time de trabalho execute suas tarefas com mais clareza e colaboração. E é por isso que, obviamente, o *kanban* não funciona como um substituto para o *Scrum*, no entanto, se utilizados juntos podem formar um casamento perfeito ([ESPINHA, 2010](#)).

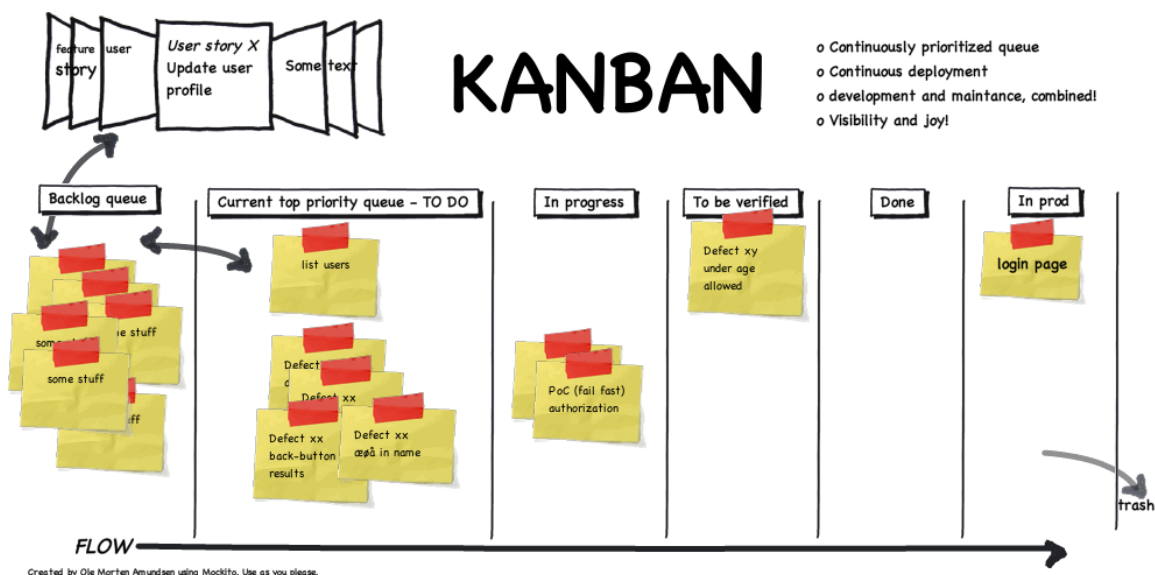


Figura 2 – Modelo de Quadro Kanban

No *kanban* pode incorporar as *Sprints* e traduzir todo o trabalho que precisa ser executado em cartões, facilitando a gestão das tarefas para agilizar as entregas e garantir autonomia, que pode atribuir as próprias tarefas, princípios tão importantes para que o *Scrum* funcione da melhor maneira. Já no que se refere a controlar as atividades, o quadro *kanban* se torna eficiente no controle acompanhando a

produtividade com o esperado. Assim está sendo aplicado as recomendações ágil e as práticas e técnicas que são suportadas, usufruindo as melhores abordagens para um cenário de um desenvolvedor.

2.4 API RESTful com Java 8 e Spring Boot

O mercado de desenvolvimento tem avançado e apresentado inúmeros recursos e padrões novos, que devem ser seguidos para atender a demanda de acessos e dados que precisam manipular nos dias de hoje. *APIs Restful* se tornaram peça chave para criação de aplicações robustas, seguindo padrões de micro serviços, além de trabalharem de modo *standalone*, ou seja, sem que uma requisição dependa de outra para executar uma determinada operação (SOUZA, 2021).

No sistema SAD, foi utilizado para a criação da *API Restful* as tecnologias *Java 8* e *Spring Boot*, projetado para consumir informações da *Web*. Segundo Azevedo (2019) essas aplicações tem por objetivo consumir informação por meio de interfaces que implementam uma série rotinas e padrões que chamamos de API. O acrônimo API vem da expressão em inglês *Application Programming Interface* (em português, Interface de Programação de Aplicações). Uma API é um conjunto de padrões e regras documentadas para que uma aplicação X possa utilizar funcionalidades de uma aplicação Y sem precisar conhecer os detalhes da implementação dessa aplicação X.

Também foi utilização dos recursos disponíveis do *Spring Boot* e foi desenvolvido a criação de tabelas do banco de dados de modo automático. Com o *Flyway* que é um *framework* que permite o controle de versão e automação durante a criação do banco de dados, foi configurado a criação das tabelas, e os dados iniciais que devem estar na base de dados. O *Flyway* também possui um utilitário de linha de comando que permite criar, atualizar e até mesmo limpar bancos de dados, tornando o gerenciamento de bancos de dados simples e intuitivo.

Conforme (VALE, 2020) ao armazenar as senhas dos usuários na *API Rest* a aplicação utiliza um modo seguro no banco de dados, utilizando o *BCrypt* que permite encriptar irreversivelmente um certo valor, o que é ideal para armazenar informações com segurança em um banco de dados. O mais interessante é que, se chamado várias vezes, o *BCrypt* criará diferentes valores de *hash* para o mesmo valor, o que torna sua criptografia muito eficiente e segura, sendo a função *hash* um algoritmo matemático para a criptografia, na qual ocorre uma transformação do dado (como um arquivo, senha ou informações) em um conjunto alfanumérico com comprimento fixo de caracteres.

Ao desenvolver a *API Restful* foi implementado o *BCrypt*, para o armazenamento correto das senhas no banco de dados, para que as senhas não fiquem claramente visíveis nas tabelas, se tornando um problema de segurança. O *Spring Security* que é uma estrutura que se concentra em fornecer autenticação e autorização para aplicativos JAVA e que disponibiliza o mecanismos de codificação *BCrypt*, possui alguns outros mecanismos como *MD5PasswordEncoder* e o *ShaPasswordEncoder*, entretanto atualmente são considerados como obsoletos, por este motivo a escolha da implementação do *BCrypt*.

Desenvolvendo os componentes *Spring* como serviços *Restful* da API, o *Spring Rest* possui a anotação *Controller* que uma vez adicionada a uma classe Java, aceitará um *path* como parâmetro. Um objeto *path* contém o nome do arquivo e a lista de diretórios usados, para construir o caminho e é usado para examinar, localizar e manipular arquivos e também tornará esse componente disponível para acesso HTTP para o *path* adicionado. Com os *controllers*, é possível gerenciar os verbos HTTP (GET, POST, PUT, DELETE,...) para cada método da classe, permitindo criar todos os acessos Restful para a API.

Conforme Azevedo (2019) o protocolo HTTP tem sido usado desde 1990 e a versão atual do protocolo é o HTTP/3. O protocolo define oito métodos que determinam ações a serem efetuadas no momento da requisição de algum recurso ao servidor. Desses oito, os 4 mais utilizados são:

GET: método utilizado para ler e recuperar dados. Requisita uma representação do recurso especificado e retorna essa representação.

POST: método utilizado para criar um novo recurso. Envia dados ao servidor. O tipo do corpo da solicitação é indicado pelo cabeçalho Content-Type.

PUT: cria um novo recurso ou substitui uma representação do recurso de destino com os novos dados. A diferença entre PUT e POST é que PUT é idempotente: ao chamá-lo uma ou várias vezes sucessivamente o efeito é o mesmo, enquanto se chamar o POST repetidamente pode ter efeitos adicionais. Por exemplo, se criarmos um produto com POST, se a URL definida na API for chamada 20 vezes, 20 produtos serão criados e cada um deles terá um ID diferente. Já o com o PUT se você executar a URL definida na API 20 vezes, o resultado tem que ser o mesmo: o mesmo produto atualizado 20 vezes.

DELETE: exclui o recurso.

Para expor uma API com o Spring Boot, a primeira coisa a fazer é adicionar a dependência do *Spring Boot Web*, para isso foi adicionado ao arquivo pom.xml:

```
<dependency>
<groupId> org.springframework.boot </groupId>
<artifactId> spring-boot-starter-web </artifactId>
</dependency>
```

Figura 3 – Dependência Adicionadas

Após salvar o arquivo para instalar as dependências, que incluem o *Tomcat*, *Jackson*, entre outras. Depois, foi criada uma classe Java com o seguinte código:

```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
@RequestMapping ( "/api/exemplo" )
public class ExemploController {

    @GetMapping ( value = "{nome}" )
    public String exemplo ( @PathVariable ( "nome" ) String nome ) {
        return "Olá " + nome ;
    }
}
```

Figura 4 – Criação da Classe

No código acima, *@RestController* será o responsável por criar a *API Rest*, seguido do *@RequestMapping*, que indicará o path do serviço. Após isso, basta mapear os métodos dos *controllers* com a anotação *@GetMapping*, seguido de um valor opcional como parâmetro. A *@GetMapping* se refere a requisições HTTP GET, para outras como POST, PUT e DELETE, basta mudar a anotação para o formato desejado, como: *@PostMapping*, *@PutMapping* e *@DeleteMapping*, respectivamente.

Como demonstrado por [SOUZA \(2021\)](#) a *@PathVariable* serve para obter um valor passado na URL nos serviços RESTful, tanto os dados quanto as funcionalidades são considerados recursos e ficam acessíveis através da utilização de URIs. Essas URI's normalmente são endereços na web que identificam tanto o servidor no qual a aplicação está hospedada quanto a própria aplicação e qual dos recursos oferecidos pela mesma está sendo solicitado. Seguindo o mapeamento do exemplo abaixo, foi executado a aplicação e acessado com a seguinte URL para testar o controller.

```
http://localhost:8080/api/emplo/NOME
```

Figura 5 – URL de Acesso

Para que possa se entender o que foi desenvolvido na API RESTful do SAD, o modelo abaixo demonstrará a arquitetura implantada. Primeiro mostra o armazenamento no banco de dados utilizando PHPMyAdmin e o Flyway para o gerenciamento do banco de dados, que será também responsável pelas migrações e controle do MySQL. Após o JPA Repository, que faz parte da API Spring Data e possui a função de fornecer o acesso ao banco e a algumas entidades JAVA padrão.

Para comunicação entre o JPA Repository e o banco de dados, existe uma camada de serviços que possui como função as regras de negócio da aplicação, em Services também possui uma unidade de cache, para fazer o cache de serviços evitando ou melhorando a performance, utilizando da distribuição JAVA Ehcache.

Após quem será responsável pelas chamadas ao Services, serão os Controllers. Os Controles farão a interfaces com Services e fará o uso dos *Data Transfer Object* (DTOs) para a comunicação com o cliente. Todos os dados recebidos e enviados serão convertidos em DTOs.

A comunicação feita pelo usuário através da interface gráfica que é chamada de UI (Interface de Usuário), acessada pela internet, farão o uso dos Controllers. Porém entre essa comunicação existe uma API de Segurança chamada Spring Security com a autenticação via tokens em JWT. E para armazenamento do código-fonte está sendo utilizado o GitHub, que permite a hospedagem prática na nuvem, podendo compartilhar o código-fonte.

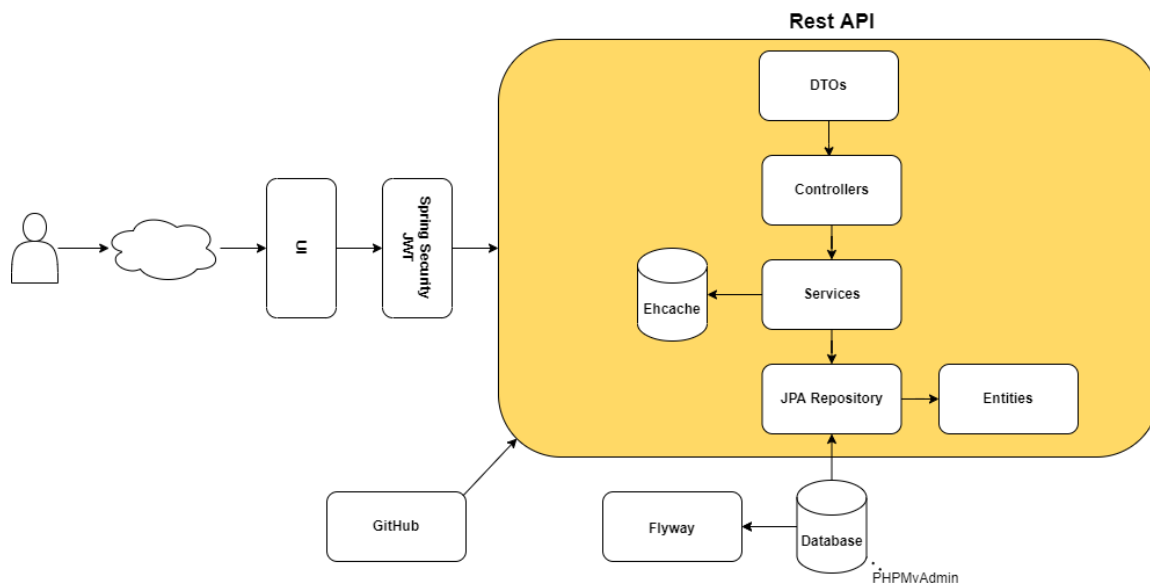


Figura 6 – Arquitetura da API RESTful

Documentar uma aplicação é um ponto essencial de qualquer projeto, muitas vezes negligenciado. A partir da necessidade de documentar a API RESTful, o recurso utilizado é o *framework* Swagger, que é composto por diversas ferramentas que, independente da linguagem, auxilia a descrição, consumo e visualização dos serviços da API.

O Swagger é a solução ideal para documentar e criar ambientes para testes de uma API RESTful, ele pode ser facilmente integrado com o Spring Boot, e de modo automático extrairá todas as informações

da API do código-fonte. Em conjunto com o Swagger UI, uma interface funcional também foi disponibilizada para a API, abaixo a interface do Swagger UI da API do Sistema de Avaliação de Desempenho Docente.

swagger default (/v2/api-docs) **Explore**

SAD - Sistema de Avaliação de Desempenho Docente API

Documentação da API de acesso aos endpoints do SAD.

cadastro-pf-controller : Cadastro PF Controller Show/Hide | List Operations | Expand Operations

POST	/api/cadastrar-pf	cadastrar
------	-------------------	-----------

cadastro-pj-controller : Cadastro PJ Controller Show/Hide | List Operations | Expand Operations

POST	/api/cadastrar-pj	cadastrar
------	-------------------	-----------

empresa-controller : Empresa Controller Show/Hide | List Operations | Expand Operations

GET	/api/empresas/cnpj/{cnpj}	buscarPorCnpj
-----	---------------------------	---------------

lancamento-controller : Lancamento Controller Show/Hide | List Operations | Expand Operations

POST	/api/lancamentos	adicionar
GET	/api/lancamentos/usuario/{usuarioId}	listarPorUsuarioId
GET	/api/lancamentos/usuario/{usuarioId}/todos	listarTodosPorUsuarioId
GET	/api/lancamentos/usuario/{usuarioId}/ultimo	ultimoPorUsuarioId
DELETE	/api/lancamentos/{id}	remover
GET	/api/lancamentos/{id}	listarPorId
PUT	/api/lancamentos/{id}	atualizar

usuario-controller : Usuario Controller Show/Hide | List Operations | Expand Operations

GET	/api/usuarios/empresa/{id}	atualizar
PUT	/api/usuarios/{id}	atualizar

[BASE URL: / , API VERSION: 1.0]

Figura 7 – Documentação Swagger

2.5 Front-end com Angular 12

O *framework* que está sendo utilizado para o desenvolvimento das interfaces web é o *Angular 12*, plataforma de aplicativo *web*, *front-end* e de código aberto, com a versão de produção mais recente, baseada na linguagem de programação *TypeScript*. Segundo Krill (2021) o angular é uma plataforma *evergreen*, o que significa que ela se mantém atualizada com o ecossistema em evolução da *web*. A remoção de suporte a navegadores legados é permitido concentrar os esforços em fornecer soluções modernas e melhor suporte aos desenvolvedores e usuários.

De acordo com Foundation (2021) projetos desenvolvidos que utilizam como padrão o *Angular 12*, possui como requisito ao menos a versão 10 ou superior do *NodeJS*, portanto foi necessário instalar a versão *NodeJS 10* para execução do projeto. Sendo o *Node.js* um *software* de código aberto, multiplataforma, baseado no interpretador V8 do *Google* e que permite a execução de códigos *JavaScript* fora de um navegador *web*.

A utilização de bibliotecas como facilitador no desenvolvimento de software, auxilia para que o tempo de desenvolvimento seja menor, com a reutilização de códigos existentes. No *front-end* do SAD, serão utilizadas as seguintes bibliotecas de extensões para o *JavaScript*:

- *RxJS*, é uma biblioteca para programação reativa que possibilita o uso da programação para a linguagem de *Java*, sendo uma biblioteca para compor programas assíncronos e baseados em eventos usando sequências *Observables* (RXJS, 2021);
- *Moment.js*, é um pacote *open source* que pode ser utilizado para validar, manipular e fazer o *parse* de datas, definindo assim os valores de data baseados em *string* no *JavaScript* (PANZOLINI, 2018).

Também está sendo utilizado o *Angular Material*, como abordado por LLC (2021) os componentes do *material design* para o *Angular*, possui alta qualidade com componentes internacionalizados e acessíveis para todos, bem testado para garantir desempenho e confiabilidade. Com *APIs* simples, com comportamento consistente entre plataformas. Versátil, para fornece ferramentas que ajudam os desenvolvedores a construir seus próprios componentes personalizados com padrões de interação comuns e customizável dentro dos limites da especificação do *Material Design*.

```
ng add @angular/material
```

Figura 8 – Exemplo de como adicionar a dependencia do Angular Material disponível em (LLC, 2021).

No desenvolvimento das interfaces web do Sistema para Avaliação de Desempenho Docente, a aplicação possui como objetivo de facilitar a experiência do usuário e estimular sua interação com a solução. Abaixo será demonstrado como o desenvolvimento das interfaces ficaram estruturadas.

SAD - Sistema de Avaliação Docente

INSTITUTO
FEDERAL
Goiás
Campus
Luziânia

Email

Senha

Entrar

Desenvolvido por Instituto Federal de Goiás - Campus Luziânia

Figura 9 – Interface - Tela de Login

O SAD possui a tela acima como tela de Login, a interface permite que o usuário acesse a plataforma, inserindo o e-mail e senha adquiridos através de um cadastro de usuário. Mantém também

uma validação se é um e-mail válido e se existe no mínimo 6 caracteres no campo senha, somente a partir dessas validações habilita o botão entrar.

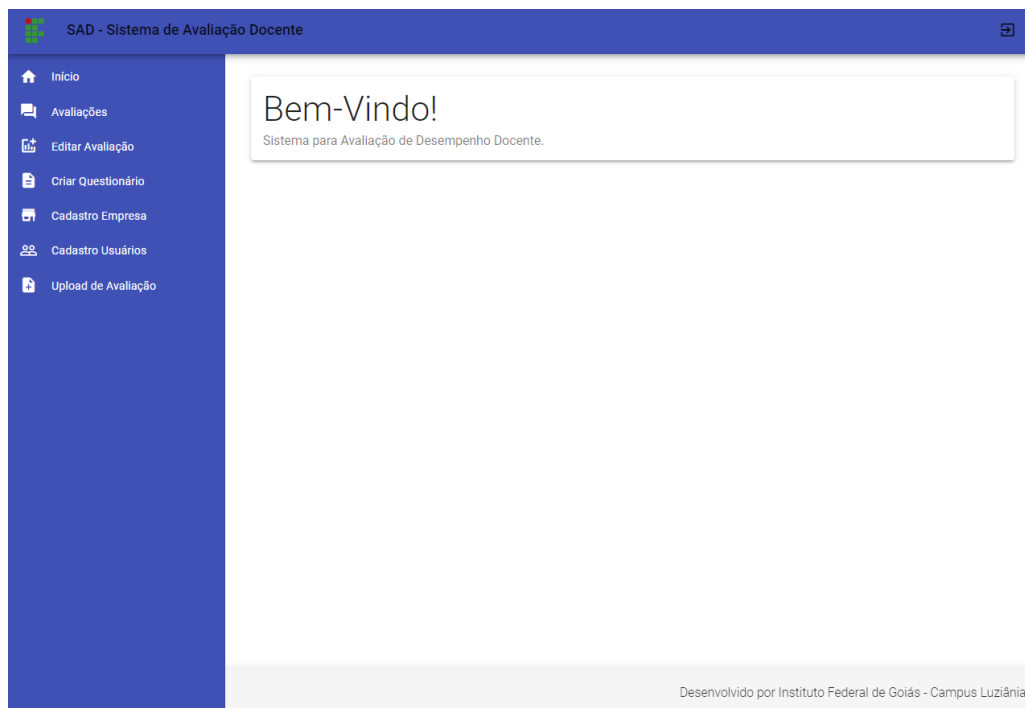


Figura 10 – Interface - Tela Inicial

A tela inicial, possui como função recepcionar o usuário que está acessando o SAD, seja ele docente, discente ou coordenador e desejar uma mensagem que seja bem-vindo.

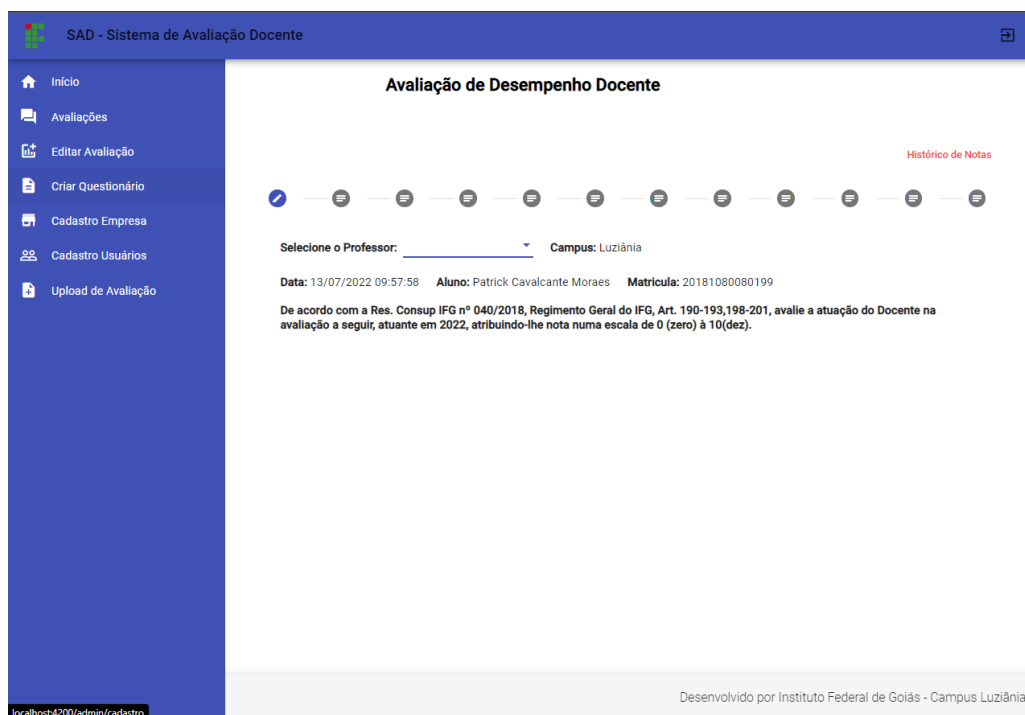


Figura 11 – Interface - Tela inicial das avaliações

Também possui uma tela de avaliação para cada tipo de usuário, sendo liberado de acordo com

o questionário que será respondido. A tela de avaliação possui a principal função do sistema, onde serão feitas as avaliações, na primeira tela carrega os dados do avaliador com o nome e matrícula, juntamente com a data e hora atual. Também é carregado o regimento pertencente as avaliações docente do Instituto Federal de Goiás, após o usuário possuirá a opção de selecionar o docente que será avaliado conforme a imagem abaixo.

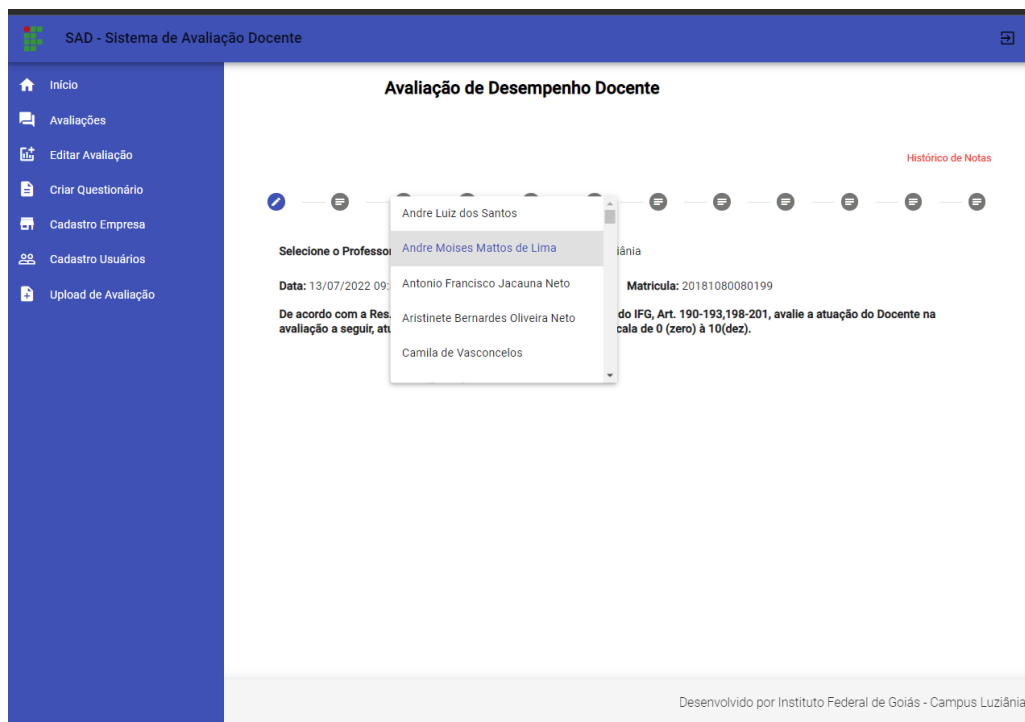


Figura 12 – Interface - Tela de avaliações com a seleção do docente

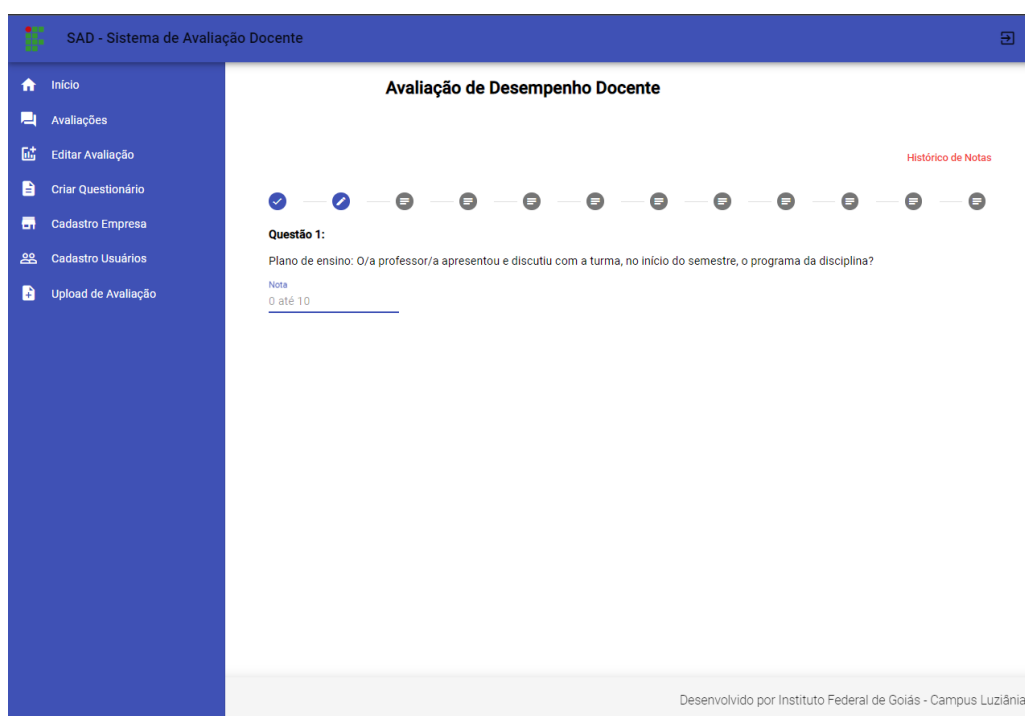
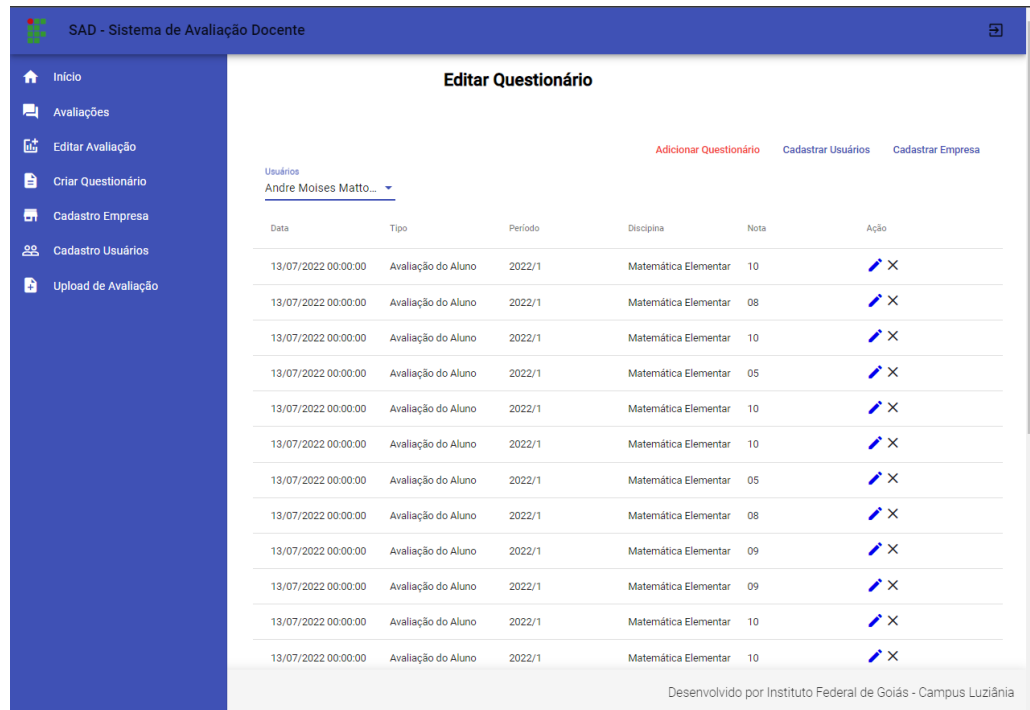


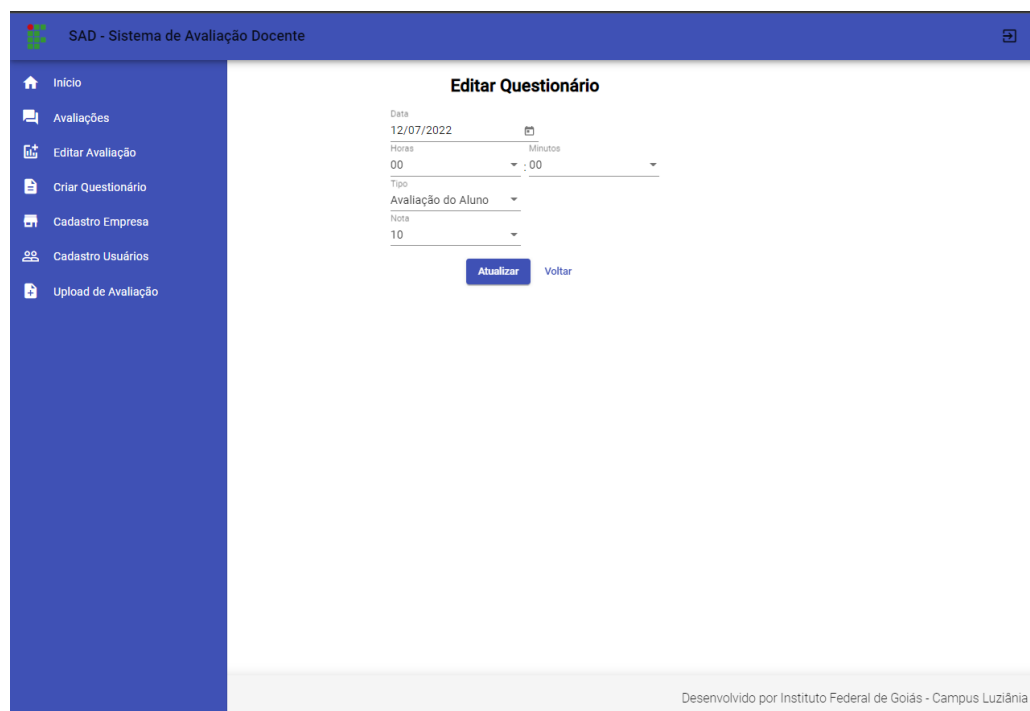
Figura 13 – Interface - Tela de avaliações com a opção para preencher a nota

Após todo o preenchimento do questionário de forma linear, atribuindo uma nota de 0 à 10 em cada resposta, conforme demonstrado na imagem acima. Será liberado na última aba, uma confirmação se o usuário deseja enviar o questionário, juntamente com um botão de enviar que possui a função de fazer um requisição na API RESTful.



Data	Tipo	Período	Disciplina	Nota	Ação
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	08	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	05	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	05	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	08	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	09	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	09	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	
13/07/2022 00:00:00	Avaliação do Aluno	2022/1	Matemática Elementar	10	

Figura 14 – Interface - Tela de listar notas



Editar Questionário

Data: 12/07/2022

Horas: 00 Minutos: 00

Tipo: Avaliação do Aluno

Nota: 10

Atualizar **Voltar**

Figura 15 – Interface - Tela de editar notas

Juntamente com a tela de listar os questionários, que são listados a partir de um filtro que necessita atribuir o docente as quais deseja verificar as notas. Possui algumas ações como a de edição

e exclusão para o usuário que possui permissão de administrador. Essa interface também permite a ordenação e paginação dos lançamentos de acordo com a necessidade do usuário.

Caso seja chamada a função de edição, retornara a tela que o administrador conseguirá fazer as alterações necessárias nos campos data, horas, minutos, tipo da avaliação e nota.

A interface apresenta um cabeçalho azul com o logo e o texto "SAD - Sistema de Avaliação Docente". À esquerda, há um menu vertical azul com ícones e links para: Início, Avaliações, Editar Avaliação, Criar Questionário, Cadastro Empresa (destacado), Cadastro Usuários e Upload de Avaliação. O formulário principal, intitulado "Cadastro de Empresa", contém campos de entrada para: CNPJ, Razão Social, CPF, Nome, Email e Senha (acompanhada de um ícone de olho para alternar visibilidade). Abaixo dos campos, há dois botões: "Cadastrar" (desativado) e "Voltar" (ativo). No rodapé, o texto "Desenvolvido por Instituto Federal de Goiás - Campus Luziânia" é exibido. No canto inferior esquerdo, uma barra de status indica "localhost:4200/cadastro-pf".

Figura 16 – Interface - Tela de cadastrar empresas

A interface apresenta o mesmo cabeçalho e menu lateral da Figura 16. O formulário principal, intitulado "Cadastro de Usuário", contém campos de entrada para: CNPJ, CPF, Funcao (menu suspenso), Nome, Email e Senha (acompanhada de um ícone de olho para alternar visibilidade). Abaixo dos campos, há dois botões: "Cadastrar" (desativado) e "Voltar" (ativo). No rodapé, o texto "Desenvolvido por Instituto Federal de Goiás - Campus Luziânia" é exibido. No canto inferior esquerdo, uma barra de status indica "localhost:4200/cadastro-pf".

Figura 17 – Interface - Tela de cadastrar usuários

Os cadastros que somente o usuário com privilégio de administrador pode acessar, estão nas telas

cadastrar empresa e usuário. O cadastrar empresa possui uma classe em JavaScript de validação do CNPJ e CPF, aonde são aceitos somente CNPJ e CPF validos. Os demais campos são o de Razão Social, Nome, E-mail e Senha, com validações de 6 dígitos no campo Senha e validação de e-mail no campo E-mail.

Após a efetuação do cadastro da empresa este CNPJ cadastrado deverá ser vinculado aos usuário de acordo com a empresa que ele atua. Este cenário existente, fará a validação do usuário pertencente a cada instituto e a segregação dos usuários. No cadastro do usuário além dos campos necessário comparados com o do cadastro da empresa que são CNPJ da empresa, CPF, E-mail e Senha validos. Existe o campo Função que possui como atribuição a diferenciação dos níveis de acesso do usuário, essas funções são denominadas no SAD como Aluno, Professor e Coordenador/Chefe Dep.

2.6 Banco de Dados com PHPMyAdmin

Segundo [Ricardo \(2006\)](#), um banco de dados é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico, ou seja, sempre que for possível agrupar informações que se relacionam e tratam de um mesmo assunto, pode-se dizer que existe um banco de dados.

A fim de estruturar o banco de dados foi constituído a estrutura de tabelas conforme demonstrado diagrama de entidade relacional na figura 18. O modelo conceitual demonstra os relacionamentos de avaliações feitas para um docente do Instituto Federal de Goiás e apresenta um diagrama de entidade relacionamento, em que são identificados os relacionamentos entre as entidade e sua cardinalidades. Também são apresentados para cada entidade suas chaves primarias e secundarias bem como seus dados.

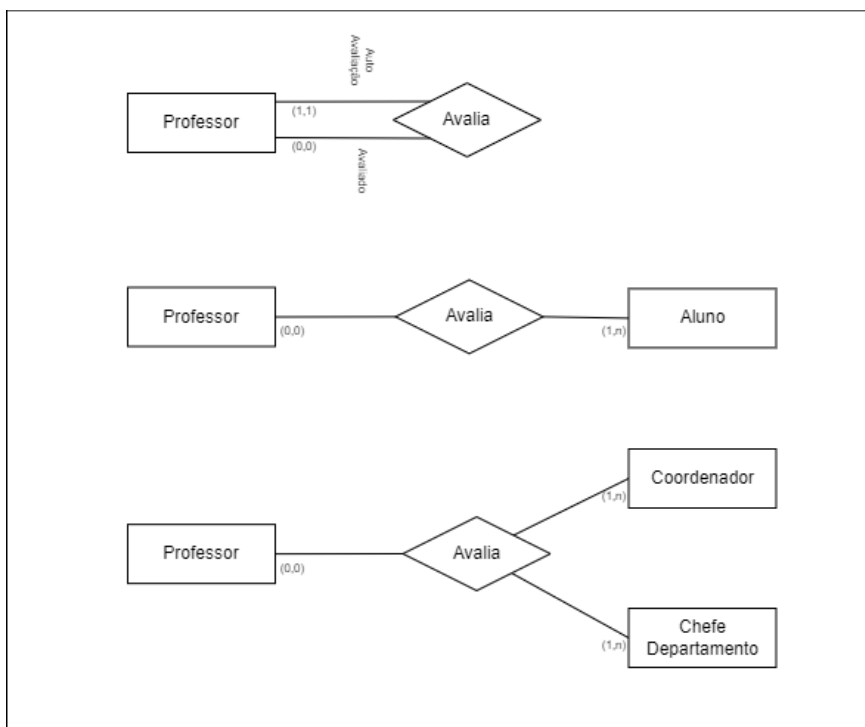


Figura 18 – Diagrama de Entidade Relacionamento

Conforme evidenciado por [Santos \(2020\)](#) foi desenvolvido o diagrama de entidade relacional para facilitar o projeto logico do banco de dados, permitindo a representação da estruturação lógica, sendo um dos modelos de dados com maior capacidade semântica e representa um problema como um conjunto de entidades e relacionamentos entre estas entidades.

O banco de dados utilizado no projeto, é o MySQL com o PhpMyAdmin, uma ferramenta de software livre, destinada a lidar com a administração do MySQL pela Web. Desenvolvida exclusivamente para se trabalhar com o SGBD MySQL, uma das principais característica é a facilidade de se trabalhar com o SGBD MySQL, onde possui um foco na modelagem física.

Na imagem abaixo demonstra com está modelado o banco de dados sad e suas respectivas tabelas, sendo elas usuario, empresa, lancamento, flyway_schema_history e hibernate_sequence. Segundo [Macêdo \(2011\)](#) o conceito básico de chave de um banco é que é uma ou mais colunas que distiguem uma linha das demais dentro de uma tabela, sendo esta chamada de chave primária (*PK – Primary Key*) ou para relacionar com outra tabela, chamada de chave estrangeira (*FK – Foreign Key*). Essas chaves é que determinam a unicidade de cada registro dentro de uma tabela.

Logo os relacionamentos entre as tabelas usuario e lancamento juntamente com o relacionamento das tabelas usuario e empresa, são feitos entre as chaves primárias e a chaves estrangeiras, construindo um banco de dados mais seguro evitando possíveis registros duplicados, pois os registros serão unicos dentro da tabela determinada pelas chaves.

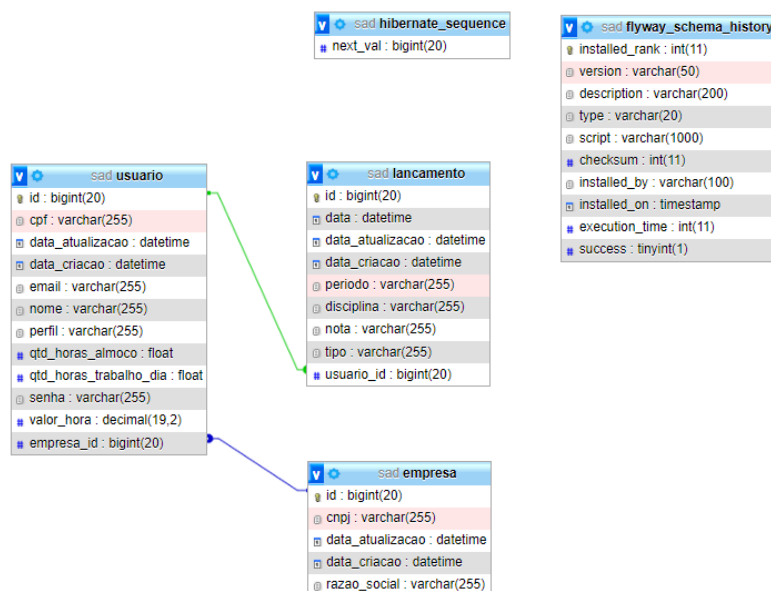


Figura 19 – Modelo Físico

Já as tabelas flyway_schema_history e hibernate_sequence são tabelas de controle que possui funções essenciais com o backend do sistema SAD. A flyway_schema_history é uma tabela de histórico de esquema do banco de dados, ela é popular a partir que a APIRESTful quando o flyway começa a escanear o sistema e localiza os arquivos SQL no diretório destinado, efetuando as migrações para SGBD, que gerará um histórico do esquema com as versões migradas, de acordo com a implementação:

installed_rank	version	description	type	script	checksum	installed_by	installed_on	execution_time	success
1	1	init	SQL	mysql/V1_init.sql	-1554986286	root	2022-07-13 09:18:58	600	1
2	2	admin padrao	SQL	mysql/V2_admin_padrao.sql	1747777283	root	2022-07-13 09:18:58	10	1
3	3	usuarios	SQL	mysql/V3_usuarios.sql	-538863607	root	2022-07-13 09:18:58	84	1
4	4	lancamentos alunos	SQL	mysql/V4_lancamentos_alunos.sql	-536828696	root	2022-07-13 09:19:01	1649	1
5	5	lancamentos autoavaliacao	SQL	mysql/V5_lancamentos_autoavaliacao.sql	-976114091	root	2022-07-13 09:19:01	34	1

Figura 20 – Tabela de Histórico de Esquema

A abordagem pensada está visando desempenho do banco de dados, será criado uma zona segura de processamento seguindo um padrão de escalar verticalmente, refere-se à quantidade que pode ser consumida sem atingir o limite de saturação. Neste cenário, a zona segura é considerada ao respeitar a quantidade de recursos estipulada, ou utilizando até 60% de CPU.

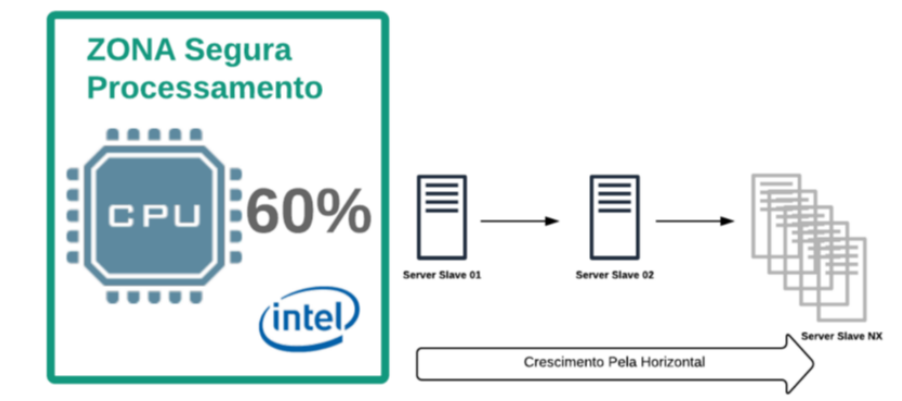


Figura 21 – Modelo de Implementação do Banco de Dados TOTVS (2021)

Uma das recomendação a ser seguida no momento da implantação do SAD, no *hardware* físico indicado para uma melhor performance, o uso de volumes *High Performance* (como *SSD/Flash*), considerando a volumetria/conexões do ambiente.

2.7 Abordagens de Testes de Usabilidade e Desempenho do Sistema

Baseado nos requisitos levantados juntamente com a Comissão Permanente de Pessoal Docente, foi desenvolvido o Sistema para Avaliação de Desempenho Docente, e para efetuação dos testes de desempenho e usabilidade do sistema, serão desenvolvidas algumas abordagens para concretização do projeto.

Segundo Patel (2021) teste de usabilidade é um método de verificação de funcionalidades da interface de uma plataforma digital. É empregado em websites, aplicações e outras ferramentas, levando usuários reais à execução de determinadas tarefas. Após sua realização, é realizada uma análise de usabilidade e das principais dificuldades.

Conforme abordado por UFPE (2006) o teste de desempenho é uma classe de testes implementada e executada para caracterizar e avaliar as características relacionadas ao desempenho do destino do teste, como perfis de sincronização, fluxo de execução, tempos de resposta e confiabilidade e limites operacionais.

Portanto, ao realizar o teste enxergará a interface do ponto de vista de que a utilizará. Será efetuado o modelos de teste que visa a descoberta de problemas, esse modelo de teste quando aplicado, seu objetivo é identificar e corrigir eventuais problemas existentes na aplicação, observando quais são os obstáculos para a fluida utilização.

Existirá numa primeira etapa a efetuação dos testes de aceitação juntamente com os integrantes da CPPD, com um protótipo já elaborado serão efetuadas abordagens de aceitação juntamente com alguns docentes e alunos, elaborando um processo de avaliação de desempenho. Essa avaliação será preenchida, e a partir deste preenchimento poderá avaliar se os dados gerados conseguiram atender a demanda esperada e se o desempenho e usabilidade do software necessitará de ajustes.

3 HIPÓTESE

A hipótese deste trabalho é apresentar como o Instituto Federal de Goiás pode agregar com uma solução de pesquisa, que tem como proposta construir um Sistema de Avaliação Docente, pois há uma inexistência de um sistema estruturado para avaliação de desempenhos, conforme descrito no capítulo 1.1(Problema). Também será observado seus benefícios como a redução de custos e o aumento da agilidade, da otimização em processos, da acessibilidade e disponibilidade e da segurança junto a CPPD.

4 METODOLOGIA

Nesta seção é apresentado como este projeto será desenvolvido para atingir o objetivo esperado. Sendo dividido em quatro etapas:

- Na primeira etapa, será realizado uma pesquisa bibliográfica sobre o tema em livros, artigos, dissertações, teses e e-books. Para a busca deste material, serão utilizadas ferramentas online como Google Scholar, Jurn, CAPES, Bibliotecas digitais de universidades e a biblioteca da própria instituição.
- Na segunda etapa será levantado os requisitos definindo os requisitos funcionais e não funcionais. E a criação do diagrama de caso de uso, para concepção da visão geral do sistema.
- Na terceira etapa será criado o protótipo do sistema, aplicando os requisitos levantado na etapa anterior. Nesta etapa será desenvolvido a API RESTful e integração com o front-end, aplicando no decorrer do desenvolvimento abordagens segundo as metodologias ágeis.
- Na quarta e última etapa será efetuado uma comparação com os processos destacados anteriormente e a execução dos testes de aceitação do software juntamente com a CPPD. Criando possíveis novos pontos de função e melhorias para o sistema.

5 RECURSOS NECESSÁRIOS

Os recursos necessários para a criação do Sistema para Avaliação de Desempenho serão:

- Computador com configuração mínima de, 3.0 GHz de processamento, 4Gb de RAM, 2Gb de espaço livre no HD;
- Papel Sulfite e cartuchos para impressão do trabalho;
- Internet para realizar as pesquisas;
- Software IDE de Desenvolvimento IntelliJ e Visual Studio Code.
- XAMPP responsável pelo Apache e MySQL.
- A plataforma OverLeaf para o desenvolvimento do trabalho escrito.

6 CRONOGRAMA

Esta seção apresenta o cronograma deste projeto, como será dividida as atividades dentro do tempo estimulado para a realização deste trabalho.

- **Atividade 1:** Fazer o levantamento de requisitos;
- **Atividade 2:** Modelar os requisitos na forma de diagramas;
- **Atividade 3:** Elaborar o projeto de software;
- **Atividade 4:** Implementar o software;
- **Atividade 5:** Apresentar o protótipo a CPPD;
- **Atividade 6:** Validar o protótipo;
- **Atividade 7:** Efetuar os testes;
- **Atividade 8:** Escrita do projeto.

Abaixo temos a tabela do cronograma a ser seguido:

Cronograma									
Mês	Semana	Atividade							
		1	2	3	4	5	6	7	8
Abril	S1	X							X
	S2	X							X
	S3		X						X
	S4		X						X
Maio	S1		X						X
	S2		X						X
	S3		X						X
	S4			X					X
Junho	S1			X					X
	S2			X					X
	S3			X					X
	S4			X					X
Julho	S1			X					X
	S2			X					X
	S3			X					X
	S4			X					X
Agosto	S1				X				X
	S2				X				X
	S3				X				X
	S4				X				X
Setembro	S1					X			X
	S2					X			X
	S3						X		X
	S4						X		X
Outubro	S1						X		X
	S2						X		X
	S3						X		X
	S4						X		X
Novembro	S1							X	X
	S2							X	X
	S3							X	X
	S4							X	X
Dezembro	S1								X
	S2								X
	S3								X
	S4								X

REFERÊNCIAS

- AZEVEDO, M. *Construindo uma API RESTful com Java e Spring Framework*. 2019. <https://mari-azevedo.medium.com/construindo-uma-api-restful-com-java-e-spring-framework-46b74371d107>. Acessado em 12 Jul, 2022.
- CENTENARO, J. Desenvolvimento de um software web para gerenciamento de requisitos de software. 2014.
- CRONAPP, R. *Afinal, quais são as diferenças entre Product Backlog e Sprint Backlog?* 2021. <https://blog.cronapp.io/diferencas-entre-product-backlog-e-sprint-backlog>. Acessado em 06 Jan, 2022.
- DIGITE. *O que é Kanban?* 2022. <https://www.digite.com/pt-br/kanban/o-que-e-kanban/>. Acessado em 21 Jan, 2022.
- DRUMOND, C. *O que é o scrum?* 2022. <https://www.atlassian.com/br/agile/scrum>. Acessado em 21 Jan, 2022.
- ESPINHA, R. *Kanban - O que é e TUDO sobre como gerenciar fluxos de trabalho*. 2010. <https://artia.com/kanban/>. Acessado em 18 Jan, 2022.
- FOUNDATION, O. *Node JS*. 2021. <https://nodejs.org/en/>. Acessado em 08 Jan, 2022.
- GOIAS, I. F. de. *Comissão Permanente de Pessoal Docente*. 2021. <http://www.ifg.edu.br/comissoes/cppd>. Acessado em 02 Jan, 2022.
- KRILL, P. *O que há de novo no Angular 12*. 2021. <https://www.infoworld.com/article/3607428/whats-new-in-angular-12.html>. Acessado em 08 Jan, 2022.
- LLC, G. *Angular Material - Material Design components for Angular*. 2021. <https://material.angular.io/>. Acessado em 08 Jan, 2022.
- MACÊDO, D. *Entendendo as Chaves dos Bancos de Dados*. 2011. <https://www.diegomacedo.com.br/entendendo-as-chaves-dos-bancos-de-dados/>. Acessado em 13 Jul, 2022.
- OLIVEIRA-CASTRO G. LIMA, G. V. M. Implantação de um sistema de avaliação de desempenho: métodos e estratégias. *Revista de Administração*, v. 31, n. 3, p. 38-52, 1996.
- PANZOLINI, B. *Datas no JavaScript com Moment.js*. 2018. <https://tableless.com.br/trabalhando-com-moment/>. Acessado em 08 Jan, 2022.
- PATEL, N. *Teste De Usabilidade: O Que É e Como Fazer Passo a Passo*. 2021. <https://neilpatel.com/br/blog/teste-de-usabilidade/>. Acessado em 13 Jan, 2022.
- RICARDO. *Conceitos Fundamentais de Banco de Dados*. 2006. <https://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649>. Acessado em 18 Jan, 2022.
- RXJS. *RxJS - Reactive Extensions Library for JavaScript*. 2021. <https://rxjs.dev/>. Acessado em 08 Jan, 2022.
- SANTOS, P. *O Que É Modelo Entidade-Relacionamento (MER)?* 2020. <https://cadernodeprova.com.br/modelo-entidade-relacionamento-mer/>. Acessado em 09 Jan, 2022.
- SOUZA, M. *API RESTful com Spring Boot e Java 8*. [S.l.], 2021.
- TOTVS. *GCP - Google Cloud Platform*. 2021. <https://tdn.totvs.com/display/public/PROT/GCP+-+Google+Cloud+Platform>. Acessado em 21 Jan, 2022.

- UCHOA, J. P. *Evolução da metodologia do desenvolvimento de sistemas*. 2022. [Http://www.linhadecodigo.com.br/artigo/2108/evolucao-da-metodologia-do-desenvolvimento-de-sistemas.aspx](http://www.linhadecodigo.com.br/artigo/2108/evolucao-da-metodologia-do-desenvolvimento-de-sistemas.aspx). Acessado em 21 Jan, 2022.
- UFPE. *Conceito - Teste de Desempenho*. 2006. [Https://www.cin.ufpe.br/gta/rup-vc/core.base_rup/guidances/concepts/performance_testing37A31809.html](https://www.cin.ufpe.br/gta/rup-vc/core.base_rup/guidances/concepts/performance_testing37A31809.html). Acessado em 18 Jan, 2022.
- VALE, S. *O que é Hash e como funciona?* 2020. [Https://www.voitto.com.br/blog/artigo/o-que-e-hash-e-como-funciona](https://www.voitto.com.br/blog/artigo/o-que-e-hash-e-como-funciona). Acessado em 02 Jan, 2022.
- XAVIER P. PIRES, J. F. F. C. *Projeto de Avaliação Institucional Avaliação Interna (Auto-Avaliação)*. [S.l.]: Bahia, 2005.