



**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**

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Bacharelado em Sistemas de Informação do Instituto Federal de Educação, Ciência e Tecnologia de Goiás - Câmpus Luziânia, como instrumento avaliativo da disciplina de Trabalho de Conclusão de Curso.

Orientador: Wendell Bento Geraldes

Luziânia  
2022

# RESUMO

MORAES, Patrick Cavalcante. **SAD - SISTEMA PARA AVALIAÇÃO DE DESEMPENHO DOCENTE DO INSTITUTO FEDERAL DE GOIÁS**. Luziânia, 2022. 44p. Trabalho de Conclusão de Curso. Instituto Federal de Educação, Ciência e Tecnologia de Goiás

Neste trabalho descreve-se as diversas etapas da construção e implementação de um Sistema de Avaliação Docente (SAD) do Instituto Federal de Goiás. A solução é importante para melhorar o processo de avaliação docente, pois essas avaliações impactam diretamente na progressão de carreira do docente. Há uma inexistência de um sistema computacional estruturado para avaliação de desempenho, sucedendo a um grande trabalho manual dos membros da Comissão Permanente de Pessoal Docente, que possui como função a junção desses registros gerados em sistemas distintos, em uma planilha. Logo no cenário atual o SAD é necessário na automatização do processo diminuindo a margem de erro e melhorando o modelo da avaliação de desempenho docente. O sistema trabalha com as tecnologias mais atuais gerando um sistema de avaliação moderno e robusto que consiga atender as necessidades da instituição.

**Palavras-chave:** Avaliação de desempenho; sistemas de avaliação; docente; otimizar processo.

# ABSTRACT

his work describes the various stages of construction and implementation of a Teacher Evaluation System (SAD) at the Federal Institute of Goiás. The solution is important to improve the teacher evaluation process, as these evaluations have a direct impact on the teacher's career progression. There is a lack of a structured computational system for evaluating performance, which is followed by a lot of manual work by the members of the Permanent Commission of Teaching Personnel, whose function is to join these records generated in different systems, in a spreadsheet. In the current scenario, SAD is necessary to automate the process, reducing the margin of error and improving the teacher performance evaluation model. The system works with the most current technologies, generating a modern and robust evaluation system that can meet the needs of the institution.

**Keywords:** Performance evaluation; assessment systems; teacher; optimize process.

# LISTA DE ILUSTRAÇÕES

Figura 1 – Atividade de Gerenciamento de Requisitos Wiegers (2003).	12
Figura 2 – Modelo de Caso de Uso Ventura (2011).	12
Figura 3 – Modelo de Quadro Kanban Amundsen (2010).	14
Figura 4 – Trello - Gerenciamento de Tarefas Rehkopf (2019).	15
Figura 5 – Dependência Adicionadas.	16
Figura 6 – Criação da Classe.	17
Figura 7 – URL de Acesso.	17
Figura 8 – Exemplo de IU do Swagger Laube (2018).	18
Figura 9 – Exemplo de como adicionar a dependencia do Angular Material disponivel em (LLC, 2021).	19
Figura 10 – Componentes de um Sistema de Banco de Dados Bordallo (2021).	20
Figura 11 – Exemplo de Diagrama de Entidade Relacionamento Cavalcanti (2015).	20
Figura 12 – Modelo de Implementação do Banco de Dados TOTVS (2021).	21
Figura 13 – Processo de Testes Unitários Manoel (2009).	22
Figura 14 – Organização das Classes de Testes Manoel (2009).	23
Figura 15 – Diagrama de Caso de Uso.	25
Figura 16 – Descrição do Casos de Uso.	26
Figura 17 – Quadro Kanban do Sistema de Avaliação Docente,	27
Figura 18 – Diagrama de Entidade Relacionamento.	27
Figura 19 – Execução XAMPP.	28
Figura 20 – Modelo Físico.	28
Figura 21 – Tabela de Histórico de Esquema.	29
Figura 22 – Arquitetura da API RESTful.	29
Figura 23 – Documentação Swagger.	30
Figura 24 – Documentação Swagger - Lançamento Controller.	30
Figura 25 – Método de Autenticação.	31
Figura 26 – Método de Consultar Lançamentos.	31
Figura 27 – Interface - Tela de Login.	32
Figura 28 – Interface - Tela Inicial - Adminitrador.	32
Figura 29 – Interface - Tela Inicial - Discente.	33
Figura 30 – Interface - Tela de avaliações com a seleção da disciplina.	33
Figura 31 – Interface - Tela inicial das avaliações.	34
Figura 32 – Interface - Tela de avaliações com a opção para preencher a nota.	34
Figura 33 – Interface - Tela de avaliações com o alerta de pendência de campo preenchido.	35
Figura 34 – Interface - Tela de listar notas.	35
Figura 35 – Interface - Paginação.	36
Figura 36 – Interface - Ordenação.	36
Figura 37 – Interface - Tela de editar notas.	36
Figura 38 – Interface - Tela de cadastrar empresas.	37
Figura 39 – Interface - Tela de cadastrar usuários	37
Figura 40 – Interface - Relatório das notas lançadas para os docentes.	38
Figura 41 – Estruturação das Classes de Teste.	39
Figura 42 – Execução da Classe de Teste LancamentoControllerTest.	39

Figura 43 – Execução do comando para dez mil requisições. . . . .	39
Figura 44 – Relatório de Desempenho. . . . .	40

# LISTA DE ABREVIATURAS E SIGLAS

AD	<i>Active Directory (Diretório Ativo)</i>
API	<i>Application Programming Interface (Interface de Programação de Aplicação)</i>
CPF	<i>Cadastro de Pessoas Físicas</i>
CNPJ	<i>Cadastro Nacional da Pessoa Jurídica</i>
CPPD	<i>A Comissão Permanente de Pessoal Docente</i>
CPU	<i>Central Processing Unit (Unidade de Central de Processamento)</i>
CSS	<i>Cascading Style Sheets (Folhas de Estilo em Cascata)</i>
DAA	<i>Diretoria de Administração Acadêmica</i>
DTI	<i>Diretoria de Tecnologia da Informação</i>
DTO	<i>Data Transfer Object (Objeto de Transferência de Dados)</i>
FK	<i>Foreign Key (Chave Estrangeira)</i>
HTML	<i>HyperText Markup Language (Linguagem de Marcação de Hipertexto)</i>
HTTP	<i>HyperText Transfer Protocol (Protocolo de Transferência de Hipertexto)</i>
IES	<i>Instituição de Ensino Superior</i>
IDE	<i>Ambiente de Desenvolvimento Integrado</i>
IFG	<i>Instituto Federal de Goiás</i>
JPA	<i>Java Persistence API (API de Persistência Java)</i>
JWT	<i>JSON Web Token</i>
LDB	<i>Lei de Diretrizes e Bases da Educação</i>
PK	<i>Primary Key (Chave Primária)</i>
REST	<i>Representational State Transfer (Transferência Representacional de Estado)</i>
RxJS	<i>Reactive Extensions Library for JavaScript (Biblioteca de Extensões Reativas para JavaScript)</i>
SAD	<i>Sistema de Avaliação Docente</i>
SGBD	<i>Sistema de Gerenciamento de Banco de Dados</i>
SSD	<i>Solid State Drive (Disco de Estado Sólido)</i>
TDD	<i>Test Driven Development (Desenvolvimento Guiado por Testes)</i>
UI	<i>User Interface Design (Interface de Usuário)</i>
URL	<i>Uniform Resource Locator (Localizador Uniforme de Recursos)</i>

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>8</b>
1.1	Problema e Hipótese	9
1.2	Justificativa	9
1.3	Objetivo	10
1.3.1	Objetivo Geral	10
1.3.2	Objetivos Específicos	10
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA</b>	<b>11</b>
2.1	Avaliação Docente do Instituto Federal de Goiás	11
2.2	Gerenciamento de Requisitos de Software	11
2.3	Metodologia Agil	13
2.3.1	Scrum	13
2.3.2	Kanban	14
2.4	API RESTful com Java 8 e Spring Boot	15
2.5	Front-end com Angular 12	18
2.6	Banco de Dados com PHPMyAdmin	19
2.6.1	PhpMyAdmin	21
2.6.2	Recursos Estimativos Para a Alta Performance	21
2.7	Abordagens de Testes de Usabilidade e Desempenho do Sistema	22
2.7.1	Teste de integração com @SpringBootTest	22
<b>3</b>	<b>METODOLOGIA</b>	<b>24</b>
<b>4</b>	<b>RESULTADOS</b>	<b>25</b>
4.1	Modelagem do Processo de Desenvolvimento do SAD	25
4.2	Recomendações Ágil	26
4.3	Estrutura e Arquitetura do Banco de Dados	27
4.4	Desenvolvimento da API RESTful	29
4.5	Interface e Funcionalidades do Sistema de Avaliação Docente (SAD)	32
4.6	Testes	38
<b>5</b>	<b>CONCLUSÃO</b>	<b>41</b>
5.1	Trabalhos futuros	41
	<b>REFERÊNCIAS</b>	<b>42</b>



# 1 INTRODUÇÃO

A partir de 1990, começa a ser implantada no Brasil a avaliação de desempenho, apoiando nos pilares da burocracia, mas revestida de novos ideais, dentre: o foco na desburocratização, descentralização, a autonomia do gestor, a horizontalização e a maior participação da comunidade (BRESSER-PEREIRA, 2006). Desse contexto, percebe-se que a racionalidade existente nas políticas educacionais do país tem a considerar o docente como agente essencial para o avanço da qualidade de ensino e demandar que o trabalho e a capacitação desses profissionais sejam avaliados. Surgindo a necessidade para elaboração de planos de carreira que potencializa a adoção da avaliação de desempenho docente.

A Lei de Diretrizes e Bases da Educação (LDB), no art. 67, estabelece que, os sistemas de ensino promoverão a valorização dos profissionais da educação, assegurando-lhes, inclusive nos termos dos estatutos e dos planos de carreira do magistério (BRASIL, 1996). A existência da avaliação de desempenho indica que: progressão funcional baseada na titulação ou habilitação, e na avaliação do desempenho (BRASIL, 1996), são determinações que possuem a garantia para alcançar a progressão na carreira profissional, sendo submetido a avaliação e possuindo a titulação e habilitação necessárias.

Considerando que a aprovação em avaliação de desempenho individual consta como requisito legal a ser observado e atendido para aquisição de direito à progressão funcional e promoção na carreira docente, conforme disciplinado pela Portaria MEC nº 554, de 20 de junho de 2013 (REITORIA, 2021). A Comissão Permanente de Pessoal Docente (CPPD) do Instituto Federal de Goiás (IFG) (CPPD, Portaria nº 1792/2020), a fim de dar prosseguimento à rotina para promoção e progressão dos docentes do quadro permanente de pessoal do IFG efetuou a criação da Avaliação Docente. Com isso, neste trabalho descreve-se as diversas etapas do desenvolvimento de um Sistema de Avaliação Docente (SAD) para o Instituto Federal de Goiás.

Foi verificado a necessidade da instituição em um sistema de avaliação de desempenho docente, onde observado que o modelo de avaliação aplicado pelos sistemas atuais, já não agrega resultados satisfatórios diante a necessidade atual do IFG. Sendo que o mesmo possui uma grande quantidade de dados que são mantidos e unificados manualmente. A utilização de um novo *software* assumiria uma integridade no controle desses dados gerados. Ao desenvolver um SAD, é construído um sistema que contemple as recomendações da instituição, desde as ferramentas 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 adaptável às mudanças. Essa abordagem propôs melhorias na gestão e acelerar os desenvolvimento do projeto.

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

## 1.1 Problema e Hipótese

O problema escolhido foi: como auxiliar a equipe da CPPD na otimização do processo de avaliação de desempenho docente do Instituto Federal de Goiás?

A inexistência de um sistema estruturado para avaliação de desempenho, somado à 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 e/ou chefe de departamento.

Nesse cenário, este trabalho visa atender a demanda da CPPD, e também mostrar que é possível desenvolver um sistema, 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.

Surge então a necessidade da criação do *software*, onde as avaliações serão armazenadas e servirão como histórico para esses colaboradores, e também serão utilizadas para melhoria e estruturação deste processo, tirando esses registros dos formulários *on-line* que atualmente consistem em um sistema externo, não vinculado a instituição que contempla parte das avaliações e do sistema Q-Acadêmico responsável pela outra parcela das avaliações de desempenho docente. Neste trabalho será apresentado como o Instituto Federal de Goiás pode agregar com uma solução de pesquisa, que tem como proposta construir um Sistema de Avaliação Docente (SAD). Também serão observados seus benefícios como a redução de custos e o aumento da agilidade, da otimização em processos, da acessibilidade e disponibilidade e da segurança junto à CPPD.

## 1.2 Justificativa

De acordo [XAVIER P. PIRES \(2005\)](#) as Instituições de Ensino Superior (IES), 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 responsabilidade 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 G. LIMA \(1996\)](#) os sistemas de avaliação devem ser justos e imparciais, baseados em padrões de desempenho atingíveis, objetivos, claros e 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. Em geral, o sistema visa subsidiar informações a respeito do desempenho dos professores da Instituição, de modo a promover a melhoria do ensino.

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 esse cenário, o IFG apresenta aplicações ineficientes para a automatização dos processos

de avaliações docentes. A CPPD possui uma necessidade de um *software* que seja capaz de efetuar as avaliações dos docentes pertencentes a instituição, sendo 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 realização deste projeto.

#### 1.3.1 Objetivo Geral

O objetivo geral deste projeto é melhorar o processo de avaliação de desempenho docente, contribuindo com a melhoria no processo de avaliação docente e posterior progressão funcional. Essa abordagem de avaliação irá auxiliar a descobrir os pontos fortes e fracos dos docentes, além do fato de que é uma forma importante para que os docentes entendam o processo de avaliação no trabalho.

#### 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;
- Efetuar os testes;
- Validar o protótipo;
- 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 (DOCENTE, 2021).

Conforme a legislação nacional lei nº 12.772 Brasil (2012), de 28 de dezembro de 2012 que dispõe sobre a estruturação do Plano de Carreiras e Cargos de Magistério Federal e lei nº 12.863 Brasil (2013), de 24 de setembro de 2013 que altera a lei 12.772, de 28 de dezembro de 2012, que dispõe sobre a estruturação do Plano de Carreiras e Cargos de Magistério Federal. O desenvolvimento na Carreira de Magistério Federal, ocorrerá mediante progressão funcional e promoção, na forma disposta nas leis, que estabelece diretrizes gerais para o processo de avaliação de desempenho para fins de progressão e promoção dos servidores pertencentes ao Plano de Carreiras e Cargos de Magistério Federal das Instituições Federais de Ensino vinculadas ao Ministério de Educação de que trata o Capítulo III da Lei 12.772, de 28 de dezembro de 2013.

A organização do IFG contempla 14 campus, e atualmente possui como método a Ficha de Avaliação de Desempenho. A avaliação é feita em dois períodos ao ano, de acordo com a Resolução Consup IFG nº 040/2018 Ferreira (2018) e Regimento Geral do IFG, Art. 190-193,198-201 SILVA (2012), avaliando a atuação dos Docente, atribuindo-lhe nota numa escala de 0(zero) à 10(dez).

Atualmente as avaliações docentes do Instituto Federal de Goiás são feitas em dois sistemas distintos e vinculados a uma planilha eletrônica pelos membros da CPPD, tornando-se um procedimento manual passível a falhas. Quando necessário por diversos motivos, a consulta de uma nota atribuída anteriormente, é solicitado para a CPPD a planilha que está armazenada a nota, este processo requer tempo, e um trabalho manual dos membros da comissão para que consigam disponibilizar essas notas.

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. É necessário a soma das três notas e a divisão pela soma das quantidades, um processo simples, porém que necessita de uma aplicação que consiga unificar as avaliações em um sistema moderno que possa modificar o processo para melhor.

### 2.2 Gerenciamento de Requisitos de Software

Conforme Mendes L. Costa (2022) a gerência de requisitos, abrange as atividades que tem por objetivo designar os atributos para os requisitos do software, definir suas visualizações dos requisitos de forma que as prioridades e o rastreamentos dos requisitos possam ser realizados. O processo corresponde ao conjunto de atividades que auxilia a equipe do projeto a identificar, controlar e rastrear os requisitos, bem como as alterações nos requisitos em muitos momentos do projeto.

A Figura 1 Atividade de Gerenciamento de Requisitos descrita por Wiegers (2003) visa abordar o processo de gerenciamento de requisitos passando por todas as fases que compõem o modelo de gestão. As fases são divididas em quatro, sendo elas: realização do controle de mudanças, controle das versões dos requisitos, realização do acompanhamento de todos os requisitos para visualizar os estados de cada

um e o rastreamento dos requisitos.

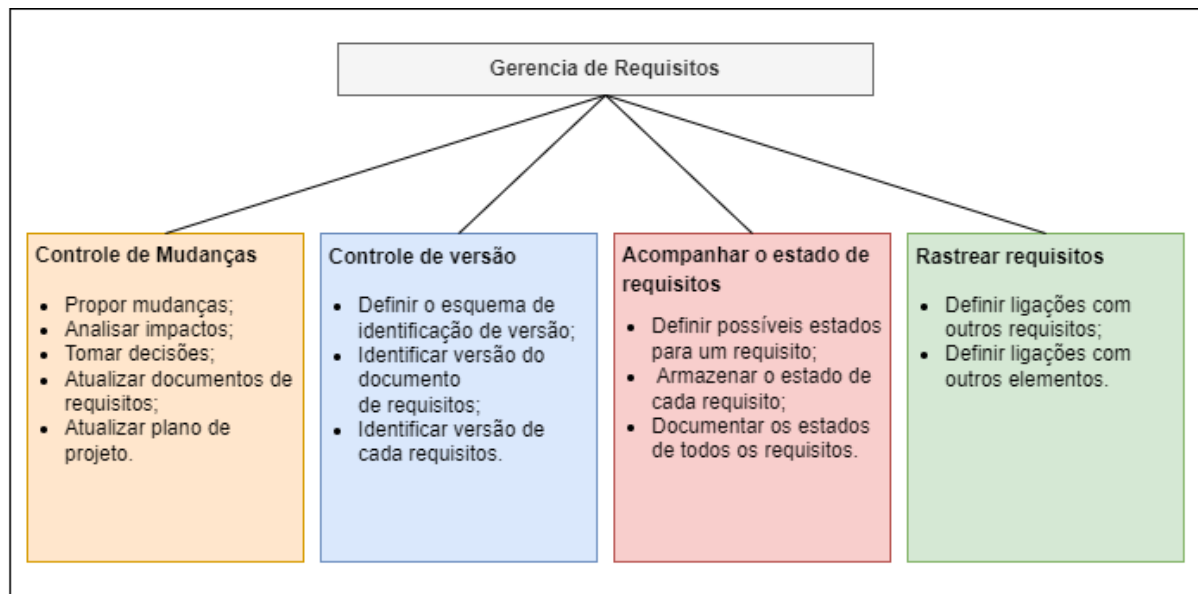


Figura 1 – Atividade de Gerenciamento de Requisitos Wieggers (2003).

O desenvolvimento de sistemas é fundamental que seja projetado para atender às necessidades do cliente. Para ajudar com a coleta das prioridades do usuário deve-se providenciar um levantamento de requisitos de *software*. O levantamento de requisitos de software é um processo que serve para capturar as necessidades do cliente antes de projetar o desenvolvimento, evidenciando que os problemas solucionados pelo sistema serão problemas reais.

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 aprofundado nos detalhes técnicos, informando como o sistema funciona.

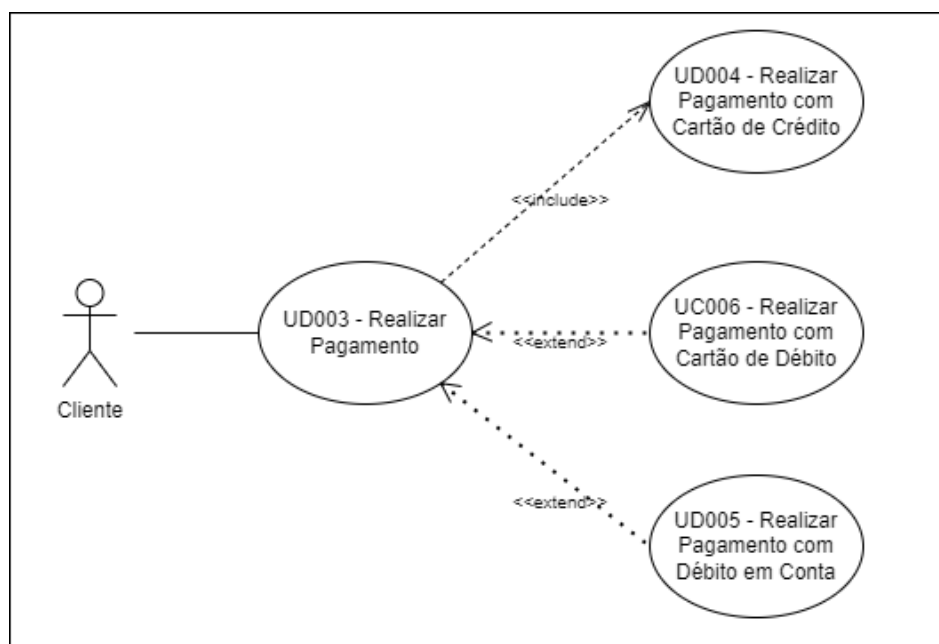


Figura 2 – Modelo de Caso de Uso Ventura (2011).

A abordagem utilizada no caso de uso acima visa descrever as interações entre o sistema com os usuários(atores), ou seja, como as funcionalidades se relacionam entre si e como serão utilizadas pelo usuário, durante o uso do sistema. Os três elementos citados no diagrama é o ator, casos de uso e os relacionamentos, sendo que o ator que fará a execução do caso de uso, já o caso de uso refere-se a interação que será tomada, portanto para que consiga efetuar o relacionamento existe os três principais relacionamentos que são: Inclusão (Include), Extensão (Extends) e Herança (Generalization).

O relacionamentos possui a função de: *Include*, quando o caso de uso A inclui o caso de uso B, significa que sempre que o caso de uso A for executado o caso de uso B também será executado. *Extend*, quando o caso de uso B estende o caso de uso A, significa que quando o caso de uso A for executado o caso de uso B poderá ser executado também. E *generalization* quando o caso de uso B generaliza o caso de uso C isso significa que, além de fazer tudo que está especificado em B, também executará tudo que está especificado no caso de uso C (PLINIO, 2014).

A Figura 2 exemplifica um diagrama de caso de uso com a função de realização de pagamento pelo cliente(ator). O caso de uso UD003 – Realizar Pagamento incluir a realização do pagamento com cartão de crédito pelo caso de uso UD004 – Realizar Pagamento com Cartão de Crédito e pode se estender com o pagamento com cartão de débito ou débito em conta pelos casos de uso UC006 – Realizar Pagamento com Cartão de Débito e UD005 – Realizar Pagamento com Débito em Conta.

## 2.3 Metodologia Agil

Metodologia Ágil é um conjunto de práticas para entender as demandas de um projeto, agir e realizar tudo com eficiência. É uma ponte que tenta eliminar as lacunas no processo de desenvolvimento de software e entregar o produto final com mais rapidez e agilidade, sempre com qualidade (JONATAN, 2020). Contudo a Metodologia ágil é uma forma de conduzir projetos que busca dar maior rapidez aos processos e à conclusão de tarefas.

### 2.3.1 Scrum

Segundo Drumond (2022) o *Scrum* é um método ágil para gestão de projetos. Sendo 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* ou *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.

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*, pode ser divididas as etapas de desenvolvimento do projeto. Outro ponto bastante utilizado da metodologia ágil, sendo 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, são 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 partir da validação será iniciado um novo *backlog* do sistema.

Também pode ser 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.

### 2.3.2 Kanban

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.

O *Kanban* é um sistema, ou seja, uma ferramenta para auxiliar no trabalho da equipe, como uma linguagem de programação, por exemplo. O sistema *kanban* 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 uma junção perfeita (ESPINHA, 2010).

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 é aplicado as recomendações ágil e as práticas e técnicas que são suportadas, usufruindo as melhores abordagens para um cenário de um desenvolvedor.

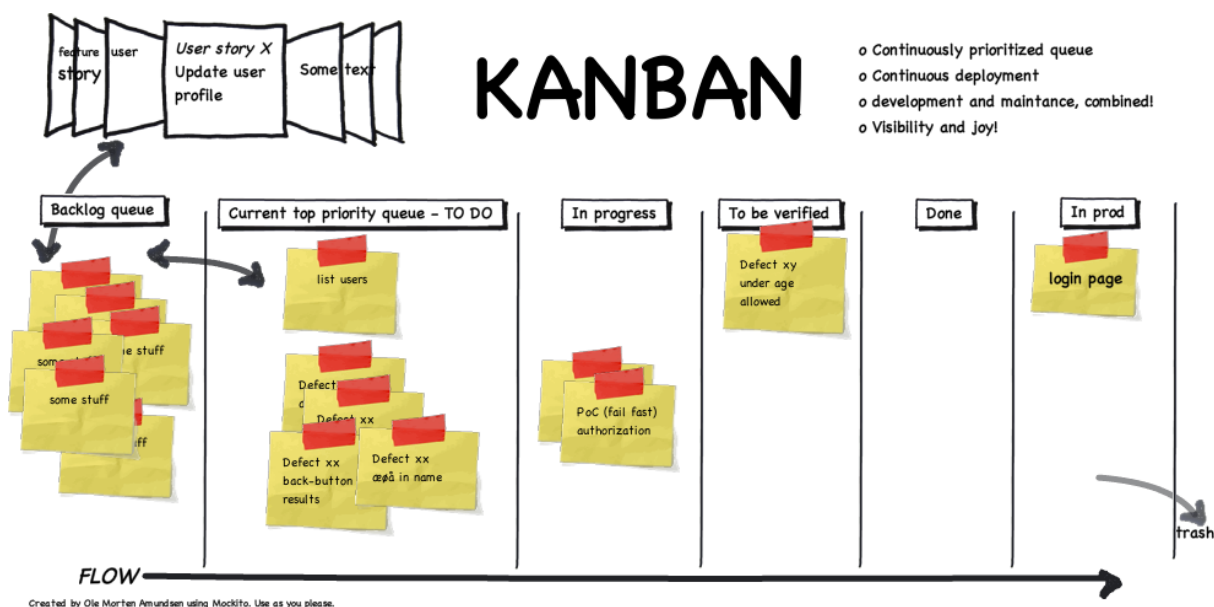


Figura 3 – Modelo de Quadro Kanban [Amundsen \(2010\)](#).

A Figura 3 demonstra a estruturação de um quadro Kanban, evidenciando a estrutura dividida entre colunas com suas atividade. As colunas foram divididas respectivamente em: Fila de Pendências, Fila de Prioridade Máxima Atual - TO DO, Em Andamento, A Ser Verificado, Feito e Em Produção possuindo algumas atividade, como: página de login em produção, listar usuários na fila de prioridade máxima e



falha rápida de autorização que está em andamento. Assim é um modelo que pode ser implementado de um quadro Kanban.

Uma das ferramentas bastante utilizada na criação dos quadros *Kanban* é o Trello. Segundo [Marc \(2014\)](#) o Trello é um “Quadro Kanban Online”, muito mais do que uma simples lista de tarefas ou *task list*. Desta forma, pode-se criar diversos planos, cada um com seu respectivo quadro *Kanban* e definir as etapas ou *buckets* dos quadros. Além disso, pode convidar a equipe e criar, atribuir e acompanhar tarefas, movendo-as entre as colunas ou *buckets*. Também pode utilizá-lo para planejar eventos, fornecer suporte, publicar conteúdos ou organizar um processo e criar quadros *Kanban* usando tarefas de conteúdo avançado com recursos incluindo arquivos, listas de verificação e rótulos.

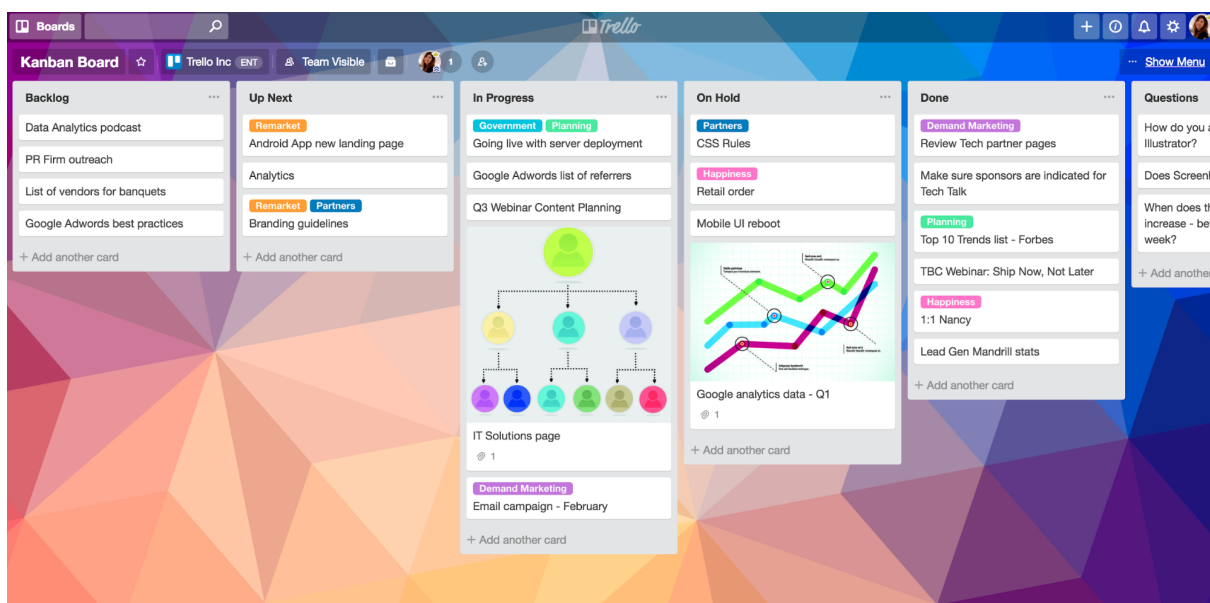


Figura 4 – Trello - Gerenciamento de Tarefas [Rehkopf \(2019\)](#).

## 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](#)).

É utilizado para a criação de diversas *API RESTful* as tecnologias *Java 8* e *Spring Boot*, projetado para consumir informações da *Web*. Segundo [Azevedo \(2019\)](#) essas aplicações têm por objetivo consumir informação por meio de interfaces que implementam uma série de rotinas e padrões que é chamada 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.

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* pode-se implementar 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*, também possui alguns outros mecanismos como *MD5PasswordEncoder* e o *ShaPasswordEncoder*, entretanto atualmente são considerados como obsoletos, por este motivo a melhor escolha é a 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 é criado 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 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, deve-se adicionar a dependência do *Spring Boot Web*, para isso é adicionado ao arquivo pom.xml a tag `<dependency>`, com o endereço do framework e o artefato id conforme evidenciado na Figura 5.

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

Figura 5 – Dependência Adicionadas.

Após salvar o arquivo para instalar as dependências, que incluem o *Tomcat*, *Jackson*, entre outras. Depois, é 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 6 – Criação da Classe.

No código acima, `@RestController` é 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 URLs 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, pode ser executada a aplicação e acessado com a seguinte URL para testar o controller.

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

Figura 7 – URL de Acesso.

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.

Com o Swagger UI, a partir da especificação da API, pode-se criar documentações elegantes e acessíveis ao usuário, permitindo assim uma compreensão maior da API, pois além de poder ver os endpoints e modelos das entidades com seus atributos e respectivos tipos, o módulo de UI possibilita que os usuários da API interajam intuitivamente com a API usando uma sandbox. A sandbox é uma plataforma de testes onde as aplicações podem ser alteradas sem interferir no meio de produção. Nela, os desenvolvedores podem executar todas as operações de mudanças experimentais que vão garantir o bom funcionamento da solução, evitando danos que possam prejudicar o sistema (DIAS, 2021).

O *Swagger* é a solução para documentar e criar ambientes para testes de uma *API RESTful*, podendo facilmente integrado com o *Spring Boot*, e de modo automático extrairá todas as informações da API do código-fonte. O modelo abaixo mostrar como as informações são documentadas no Swagger, mostrados os métodos implantados e os parâmetros necessários para as execuções.

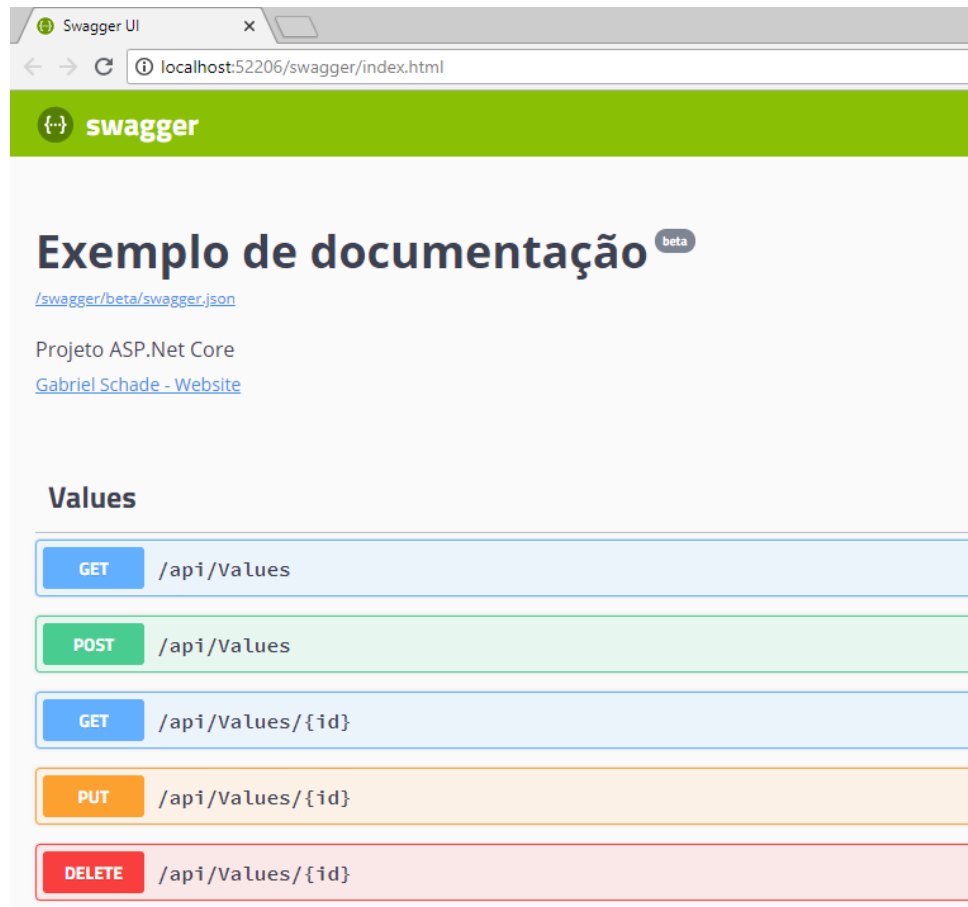


Figura 8 – Exemplo de IU do Swagger Laube (2018).

Também pode fazer a utilização de um dos recursos disponíveis do *Spring Boot* que desenvolve 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, configurando 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.

Para a efetivação das requisições sem a necessidade de criar uma interface que execute a chamada, uma ferramenta bastante utilizada é o Postman. Conforme abordado por Rodrigues (2014) o Postman é uma aplicação que permite realizar requisições HTTP a partir de uma interface simples e intuitiva, facilitando o teste e depuração de serviços REST.

## 2.5 Front-end com Angular 12

O *framework* que está sendo mais utilizado atualmente 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 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.

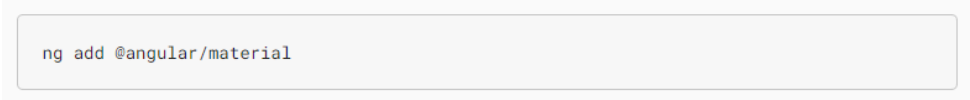
Quando é construído uma interfaces com o Angular é criado componentes para cada uma das

partes dessa interface, de forma que possa aproveitar em outras partes da aplicação sem repetição de código. Quando a prática de componentização e reaproveitamento chega ao extremo, pode-se utilizar bibliotecas de componentes prontas. Angular possui diversos recursos, e as bibliotecas fornecem elementos visuais e comportamentos padrão, que pode-se importar para a aplicação e utilizá-los, sem a necessidade de preocupar com templates *HTML* e estilos *CSS*. No universo *Angular*, uma das bibliotecas mais utilizadas é o *Angular Material* (NOLETO, 2022).

Caso seja realmente necessário o Angular se comunicar com as bibliotecas JavaScript, um exemplo que pode ser utilizado é a integração com o *jQuery* ao projeto Angular. O *jQuery* está entre as mais utilizadas, se não a mais utilizada bibliotecas *JavaScript*. 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* com Angular pode ser utilizadas bibliotecas de extensões para o *JavaScript*, como:

- *jQuery*, é uma biblioteca livre que contém funções da linguagem de programação *JavaScript* que interage com páginas em *HTML*, desenvolvida para simplificar os *scripts* executados e/ou interpretados no navegador de internet do usuário (BALDUINO, 2012);
- *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).

Já *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, fornece ferramentas que ajudam os desenvolvedores a construir seus próprios componentes personalizados com padrões de interação comuns e customizáveis dentro dos limites da especificação do *Material Design*.



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

Figura 9 – Exemplo de como adicionar a dependencia do Angular Material disponível em (LLC, 2021).

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 é 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*.

## 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.

Os bancos de dados desempenham um papel importante, senão fundamental, quando se trata de hospedagens. Como blogs, sites, lojas, aplicativos e diferentes tipos de sistemas online virtuais contam com esse recurso para armazenar todo tipo de informação. Qualquer aplicativo ou sistema baseado na web precisa de algum tipo de banco de dados.

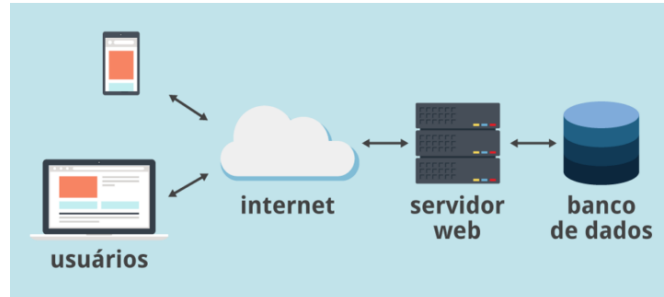


Figura 10 – Componentes de um Sistema de Banco de Dados [Bordallo \(2021\)](#).

A arquitetura mostrada na Figura 10 corresponde ao funcionamento de um sistema e como seus componente se interligam a um banco de dado. Os usuários acessam as suas interfaces do sistemas, que comunica com a internet. A internet nessa estrutura acessa o servidor web que armazena as informações em um banco de dados. Essa é uma arquitetura básica da comunicação de um sistema com um banco de dados.

Conforme evidenciado por [Santos \(2020\)](#) pode-se desenvolver o diagrama de entidade relacional para facilitar o projeto lógico 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 essas entidades.

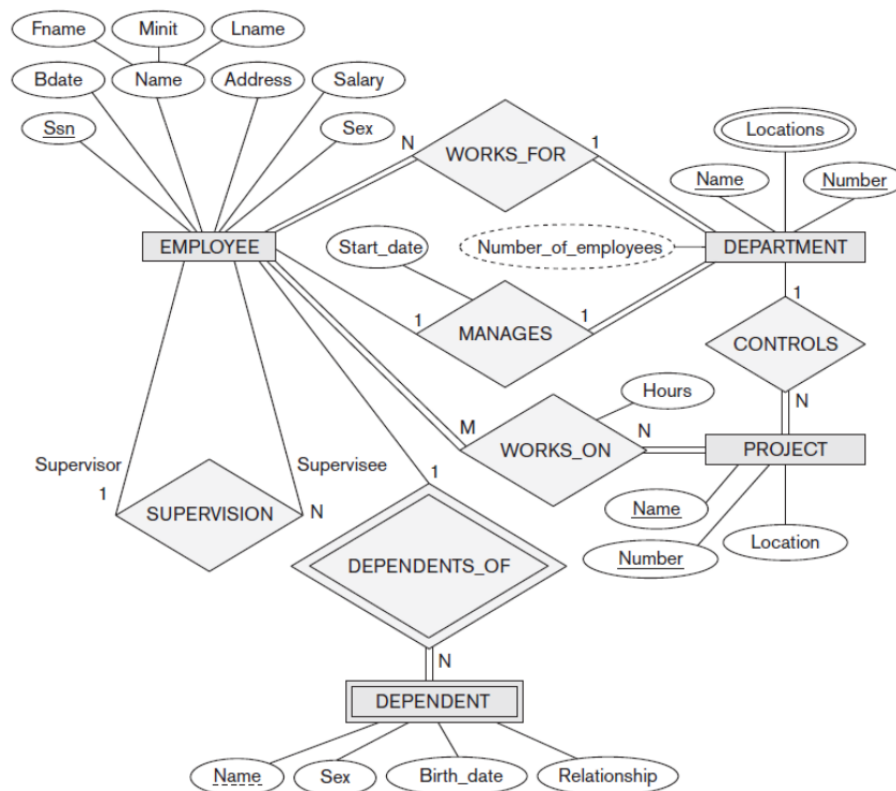


Figura 11 – Exemplo de Diagrama de Entidade Relacionamento [Cavalcanti \(2015\)](#).

Percebe-se a necessidade de armazenar e manipular dados e neste contexto é imprescindível que estes sejam armazenados de forma organizada e permitam um posterior acesso eficiente. Assim, torna-se necessário implementar o banco de dados. O Diagrama Entidade Relacionamento utiliza elementos gráficos para descrever o modelo de dados de um sistema com alto nível de abstração.

Para [Macêdo \(2011\)](#) o conceito básico de chave de um banco é que é uma ou mais colunas que distinguem 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.

### 2.6.1 PhpMyAdmin

O PhpMyAdmin é um administrador de bancos de dados em MySQL, proporcionando um trabalho de gestão e edição muito mais prático com aplicações. A ferramenta, de código aberto e uso livre, é voltada para desenvolvedores que trabalham desenvolvendo sites e ferramentas, e que precisam de uma interface mais simples. Seu principal papel é, justamente, tornar o trabalho mais simples ([SOUZA, 2020](#)).

A utilização do PhpMyAdmin é uma ferramenta destinada a lidar com a administração do Sistema de Gerenciamento de Banco de Dados (SGBD) MySQL pela Web, sendo o MySQL simplesmente um sistema que pode armazenar e gerenciar esses dados. Uma das principais características do PhpMyAdmin é a facilidade de se trabalhar com o SGBD MySQL, desenvolvida exclusivamente para o trabalho com o MySQL e também possui um foco na modelagem física.

Umas das principais ferramentas utilizadas no gerenciamento do PhpMyAdmin em diferentes sistemas operacionais é o XAMPP. O XAMPP é um pacote com os principais servidores de código aberto do mercado, incluindo FTP, banco de dados MySQL e Apache com suporte às linguagens PHP e Perl segundo [Higa \(2012\)](#).

### 2.6.2 Recursos Estimativos Para a Alta Performance

A abordagem da Figura 12 que pode-se utilizar no momento da implantação, visando o desempenho do banco de dados, é cria-se 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.

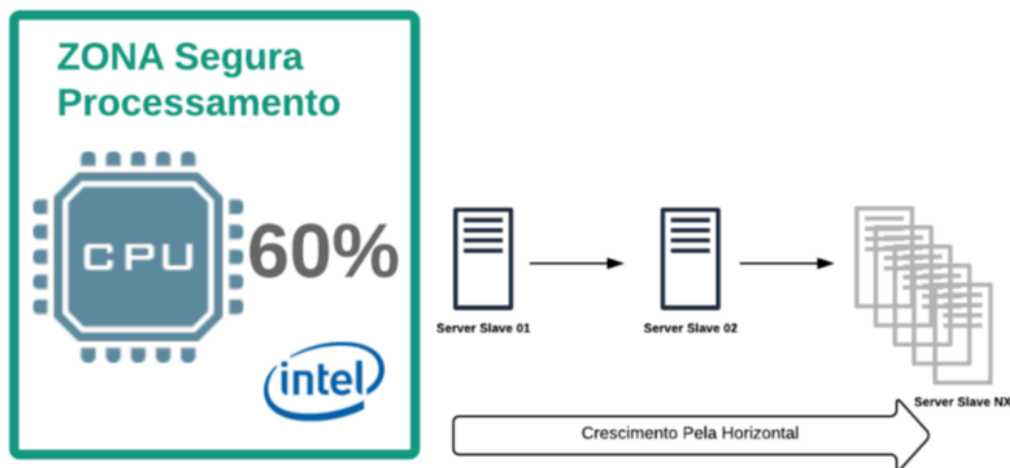


Figura 12 – Modelo de Implementação do Banco de Dados [TOTVS \(2021\)](#).

Uma das recomendação que também pode ser seguida no momento da implantação do sistema, 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

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. Sendo efetuado o modelo 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.

### 2.7.1 Teste de integração com @SpringBootTest

O conceito de TDD (Desenvolvimento Guiado por Testes) define que antes de criar um código novo (classe), deve-se escrever um teste (classe de test case). Essa prática traz vários benefícios às equipes de desenvolvimento e inclusive estes testes serão usados como métrica em todo o tempo de vida do projeto. Na imagem abaixo um modelo de como funciona o processo de testes unitários dentro do projeto ([MANOEL, 2009](#)).

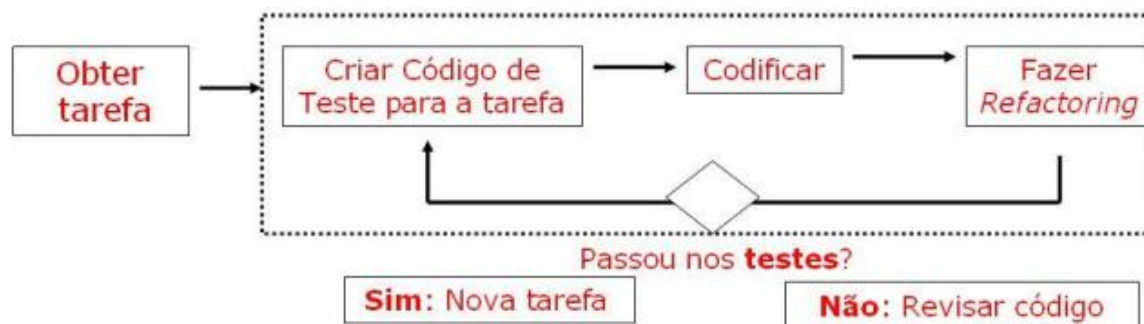


Figura 13 – Processo de Testes Unitários [Manoel \(2009\)](#).

No TDD para que possa obter um nova tarefa de desenvolvimento é necessário criar o código de teste para classe, codificar e efetuar a refatoração, caso a classe da tarefa passe no teste pode-se seguir para a criação de uma nova classe. Porém caso essa classe não passe no teste é necessário revisar o código e efetuar novamente o teste até que essa classe seja aprovada conforme mostrado na Figura 13.

Uma abordagem que se utiliza no momento do desenvolvimento é o teste de integração, o *Spring Boot* utilizado na criação da *API RESTful* possui o mecanismo JUnit que após ser implementado possui testes unitários em Java. O JUnit é um framework que facilita o desenvolvimento e execução de testes unitários em código Java. Na imagem abaixo, apresenta um diagrama que mostra a forma como as classes de testes ficam organizadas em um projeto codificado em Java.



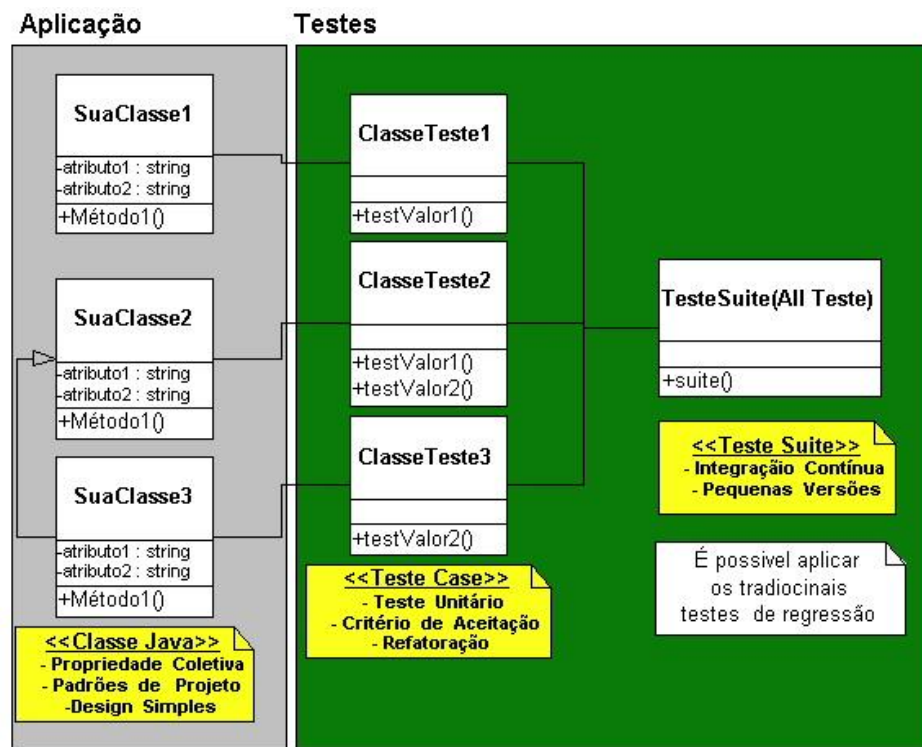


Figura 14 – Organização das Classes de Testes [Manoel \(2009\)](#).

Como o nome sugere, os testes de integração se concentram na integração de diferentes camadas do sistema. Idealmente, deve-se manter os testes de integração separados dos testes de unidade e não devem ser executados junto com os testes de unidade, porém pode-se fazer isso usando um perfil diferente para executar apenas os testes de integração. Algumas razões para realização nesse formato, podem ser que os testes de integração são demorados e pode precisar de um banco de dados real para serem executados. No entanto, também suporta a utilização do armazenamento de persistência H2 na memória.

O H2 é um banco de dados em memória que permite todas as operações, permitindo assim testar as aplicações mesmo sem um banco de dados já definido, com um console acessível pelo browser dentro do contexto da aplicação.



### 3 METODOLOGIA

Nesta seção é apresentado como este projeto é desenvolvido para atingir o objetivo esperado. Com a proposta da criação do sistema de avaliação docente, são demonstrando as etapas que levaram para o desenvolvimento do projeto.

Na primeira etapa, conforme reuniões com os atuais membros pertencentes à comissão da CPPD, são levantados os pontos necessários para o projeto do sistema para avaliação de desempenho docente. Sendo realizadas reuniões para se possa ser feito as interações e troca de informações que foram importantes para as decisões do projeto em questão. A abordagem visa 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 segunda etapa são 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. Nesta etapa são levantados os requisitos do sistema, para capturar os objetivos e as necessidades que o sistema irá atender. Juntamente com criação dos diagramas de caso de uso e diagrama de entidade relacionamento para concepção da visão geral do sistema.

Na terceira etapa é criado o protótipo do sistema, aplicando os requisitos levantados na etapa anterior. Nesta etapa é desenvolvido a API RESTful utilizando *Java 8* com o *Spring Boot* e integração com o front-end desenvolvido com o *Angular 12*, no processo de desenvolvimento é implementado o *Swagger* para documentar toda a estrutura da API, e a comunicação do *PhpMyAdmin* como a ferramenta de administração do banco de dados em *MySQL*. Também é aplicado no decorrer do desenvolvimento abordagens segundo as metodologias ágeis como *Scrum* e também o *Kanban*.

Na quarta e última etapa é efetuado a entrega do protótipo do sistema para a validação da CPPD, verificando possíveis novas atividades e melhorias para o sistema que possam ser desenvolvidos em trabalhos futuros.

## 4 RESULTADOS

Neste capítulo são apresentados os resultados obtidos com o desenvolvimento do Sistema de Avaliação Docente. Demonstrando as tecnologias fundamentadas, em mecanismo para o desenvolvimento do sistema e a aplicação estruturada como SAD.

### 4.1 Modelagem do Processo de Desenvolvimento do SAD

Baseado no que está sendo demonstrado no diagrama de caso de uso abaixo, o sistema possui funções que atenderam o cadastro de usuário, diferenciando seus papéis. As ações apresentadas pelos casos de uso demonstram como está representada 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.

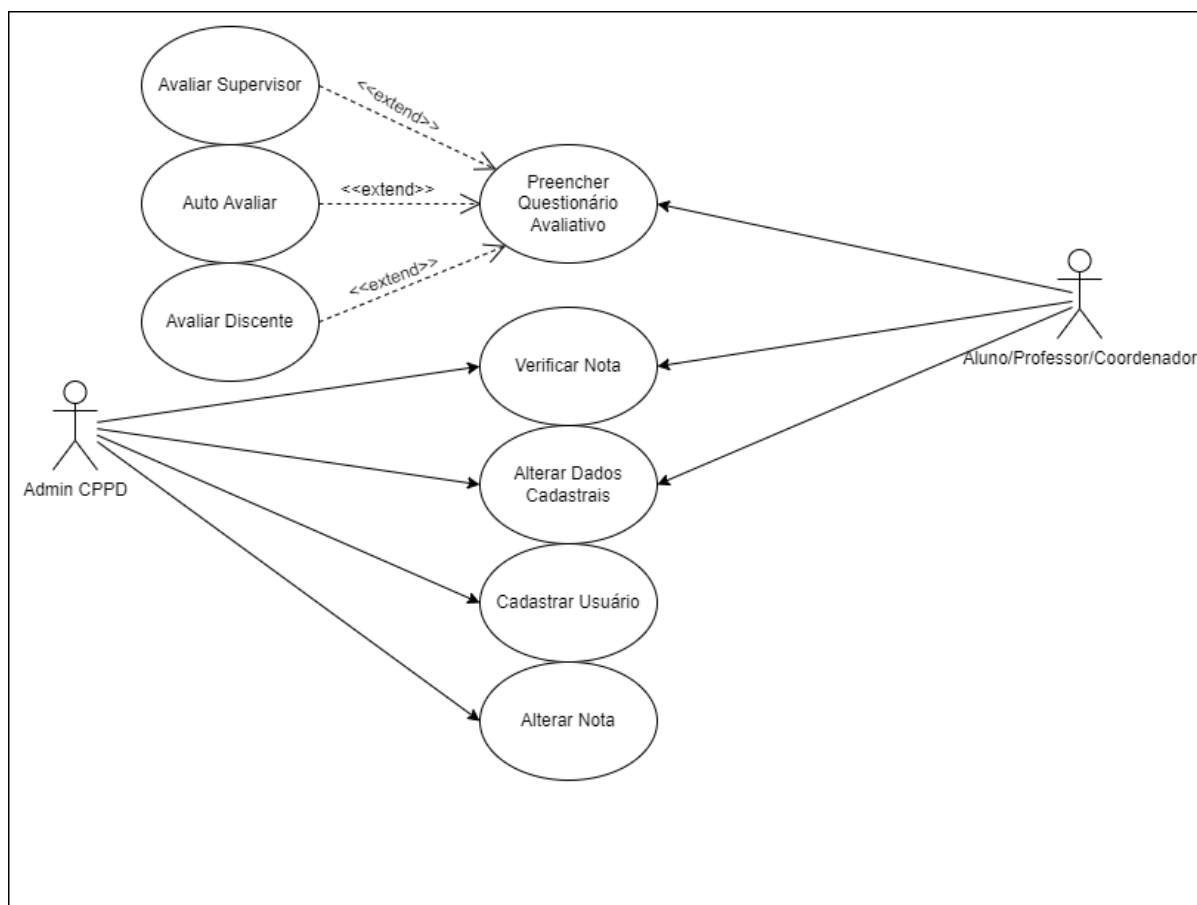


Figura 15 – Diagrama de Caso de Uso.

Este diagrama possui dois atores que são Admin CPPD e Aluno/Professor/Coordenador e demonstra a perspectiva que os usuários possuíram no SAD, os oito casos de usos evidencia as funções, possuindo também os *extend* que é utilizada para incluir um comportamento opcional de um caso de uso extensor para um caso de uso estendido.

O propósito do documento abaixo é apresentar a relação e descrição sucinta dos casos de uso do sistema. Fornecendo uma descrição clara e consistente do que o sistema deverá fazer, consistindo na

explicitação de todas as diferentes funcionalidade do sistema, que permitirá inferir e identificar mais claramente outras necessidades.

Identificação	Atores	Descrição	Rastreabilidade
Preencher Questionário Avaliativo	Usuário	O ator receberá um questionário pertinente à avaliação do Docente. O mesmo será preenchido e enviado via sistema.	
Verificar Nota	Admin CPPD	O ator verificará as notas atribuídas pelos Usuários aos Docentes.	Possui uma relação com o casos de uso "Preencher Questionário Avaliativo"
Alterar Dados Cadastrais	Admin CPPD e Usuário	O ator receberá a permissão de alterar os dados cadastrados no sistema.	Possui uma relação com o caso de uso "Cadastrar Usuário"
Cadastrar Usuário	Admin CPPD	O Admin CPPD possui a função de cadastrar o usuário para que ele possa ter atribuições pertinentes.	Possui uma relação com o casos de uso "Alterar Dados Cadastrais"
Alterar Nota	Admin CPPD	O Admin CPPD possui a função de alterar a nota lançada pelo usuário no sistema.	Possui uma relação com o casos de uso "Preencher Questionário Avaliativo"

Figura 16 – Descrição do Casos de Uso.

## 4.2 Recomendações Ágil

Seguindo as boas práticas de desenvolvimento é utilizado a ferramenta Trello para o desenvolvimento de um quadro Kanban para uma gestão ágil do desenvolvimento do Sistema de Avaliação Docente. O quadro Kanban é implementado com o intuito de fornecer e manter informações sobre o desenvolvimento do SAD.

Nele é incluído as colunas Andamento, Concluído, *Backlog* que se refere o que precisa ser elaborado e o Painel de Controle que auxilia no controle das tarefas. Os recursos da ferramenta são bastante utilizados, podendo definir prazos, critério de prioridade até alertas via e-mail referente a prazos de

entregas pendentes.

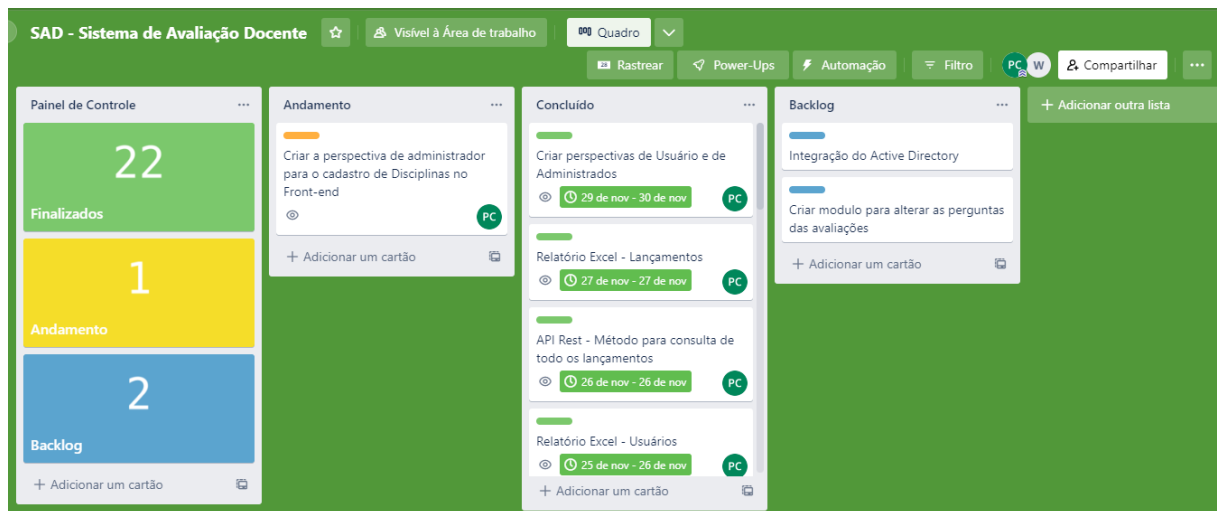


Figura 17 – Quadro Kanban do Sistema de Avaliação Docente,

Com isso é estruturado um cenário que definia os *Sprint Backlog*, o *Time Box* e o *Product Backlog* do Scrum, assim é aplicado as recomendações ágil e as práticas e técnicas que são suportadas, usufruindo as melhores abordagens para um cenário de desenvolvimento.

### 4.3 Estrutura e Arquitetura do Banco de Dados

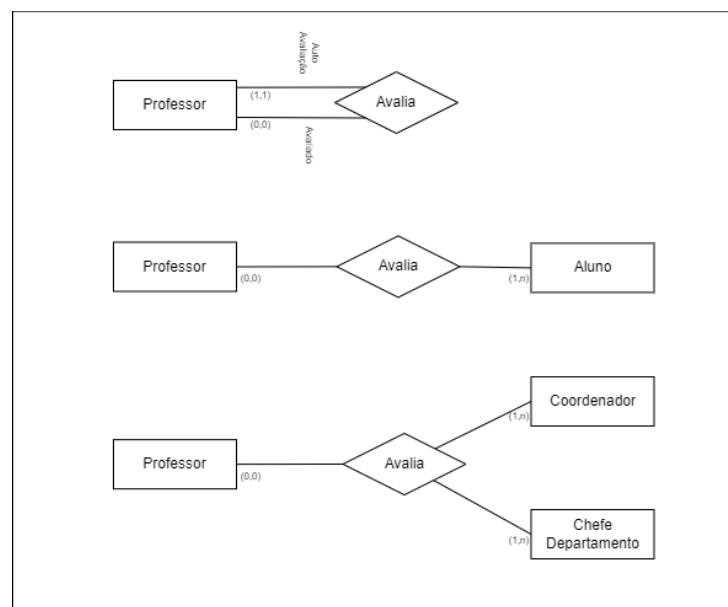


Figura 18 – Diagrama de Entidade Relacionamento.

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

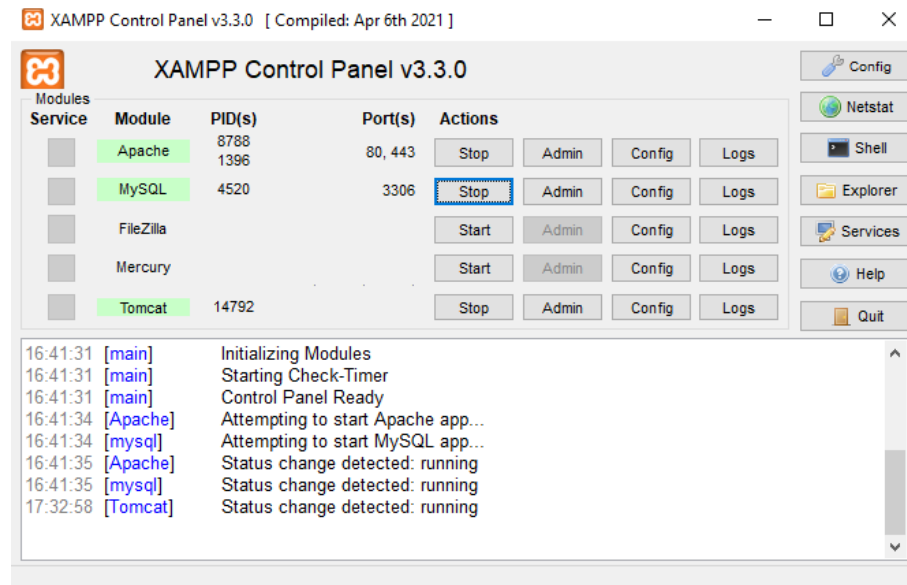


Figura 19 – Execução XAMPP.

A utilização do sistema XAMPP como ferramenta de gerenciamento do *Apache*, *MySQL* e *Tomcat*, auxiliou a utilização do *PhpMyAdmin* no servidor *web Windows*. O XAMPP inicia localmente o *PhpMyAdmin* que por sua vez está armazenado o banco de dados *sad* com suas respectivas tabelas. A imagem acima mostra a sua inicialização com algumas informações importantes em sua interface, como: Quais serviços estão em execução, portas utilizadas, além de um controle de logs e um painel de configuração que auxiliar em sua manutenção.

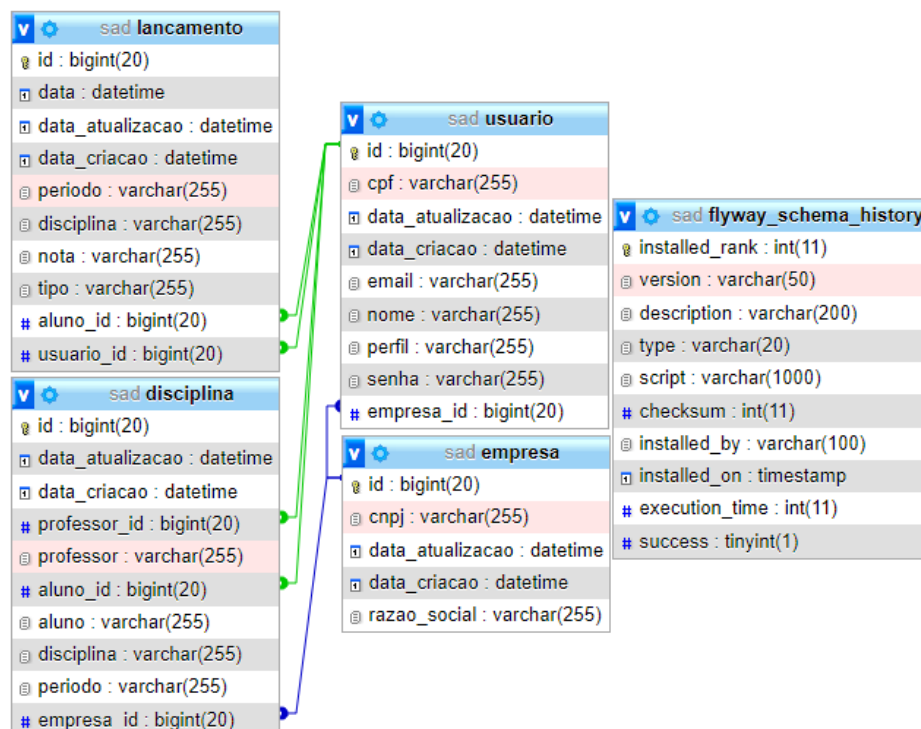


Figura 20 – Modelo Físico.

Na imagem acima demonstra como está modelado o banco de dados *sad* e suas respectivas tabelas,

contendo: usuario, empresa, lacamento, disciplina e flyway\_schema\_history. Sendo que os relacionamentos feitos entre as tabelas, são realizados entre as chaves primárias e a chaves estrangeiras, construindo um banco de dados mais seguro evitando possíveis registros duplicados, pois os registros serão únicos dentro da tabela determinada pelas chaves.

Já a tabela flyway\_schema\_history é responsável pelo controle e 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 populada a partir que a *API RESTful* 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	-411787613	root	2022-11-26 22:51:38	919	1
2	2	admin padrao	SQL	mysql/V2__admin_padrao.sql	-565854505	root	2022-11-26 22:51:38	18	1
3	3	usuarios	SQL	mysql/V3__usuarios.sql	2105727979	root	2022-11-26 22:51:38	56	1
4	4	lancamentos alunos	SQL	mysql/V4__lancamentos_alunos.sql	-1656889231	root	2022-11-26 22:51:40	1406	1
5	5	lancamentos autoavaliacao	SQL	mysql/V5__lancamentos_autoavaliacao.sql	1586414590	root	2022-11-26 22:51:40	28	1

Figura 21 – Tabela de Histórico de Esquema.

Os registros da tabela flyway\_schema\_history guardam o histórico de arquivos imputados no banco de dados do sad. Armazenando o nome do arquivo, tipo, versão, data e tempo da execução e se foi executado com sucesso, conforme evidenciado na imagem acima.

#### 4.4 Desenvolvimento da API RESTful

Para que se possa 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.

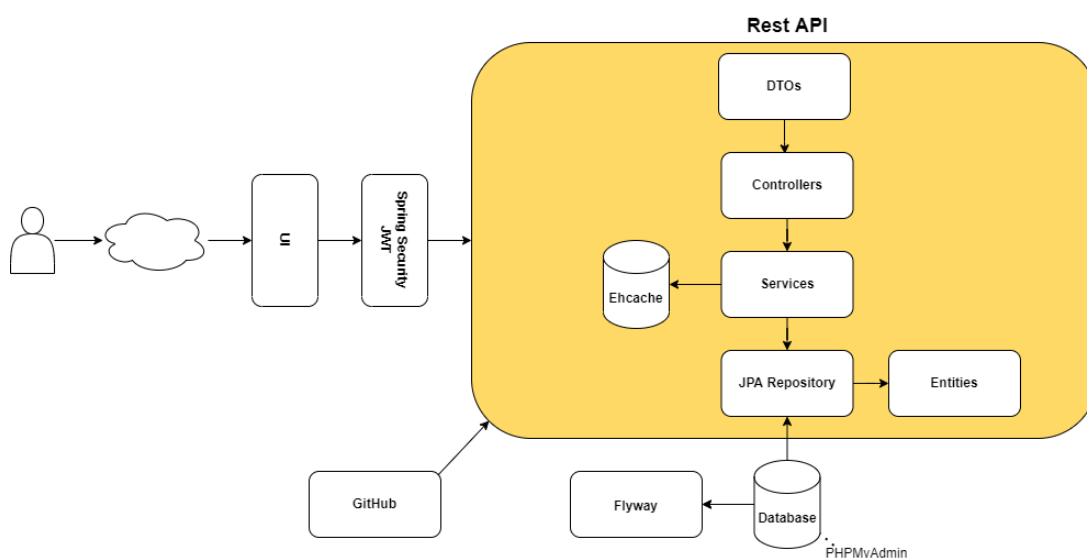


Figura 22 – Arquitetura da API RESTful.

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*. Para as chamadas ao *Services*, os *Controllers* serão responsáveis e farão a interface com *Services* e também 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, fará 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.

Uma boa documentação de uma *API REST* é crucial para desenvolvedores, testadores, mantenedores entenderem claramente o comportamento fornecido por ela. Sendo capaz de apoiar o time nas etapas de desenvolvimento, compreensão e manutenção de código.

A ferramenta que foi utilizada é o Swagger UI, uma interface funcional também foi disponibilizada para a API, abaixo a interface do Swagger UI da *API RESTful* do Sistema de Avaliação de Desempenho Docente, desenvolvida para documentar a solução, destrinchando cada método utilizado.



Figura 23 – Documentação Swagger.

lancamento-controller : Lancamento Controller			Show/Hide	List Operations	Expand Operations
POST	/api/lancamentos	adicionar			
GET	/api/lancamentos/aluno/{alunoId}	listarPorAlunoId			
GET	/api/lancamentos/todos	listarTodos			
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			

Figura 24 – Documentação Swagger - Lançamento Controller.

A listagem dos métodos de *Lancamento Controller* acima, mostra como está parametrizado cada

método para o lançamento das notas do SAD, evidenciando todas a listagens para as consultas das notas, desde de todas as notas lançadas ou as notas de um usuário ou até mesmo a última nota de um usuário, passando também pelos métodos de inclusão, alteração e exclusão. Uma forma de manter a *API RESTful* do SAD bem estruturada e compreensível para os futuros responsáveis pelo *software*.

Para efetuar os testes da API no momento do desenvolvimento foi utilizada a ferramenta Postman, a ferramenta possui como função a chamada do método para efetivação da requisição desejada. Cada método possui seu tipo e parâmetros de acordo com a requisição, será evidenciado alguns tipos que foram criados para a *API RESTful* do SAD.

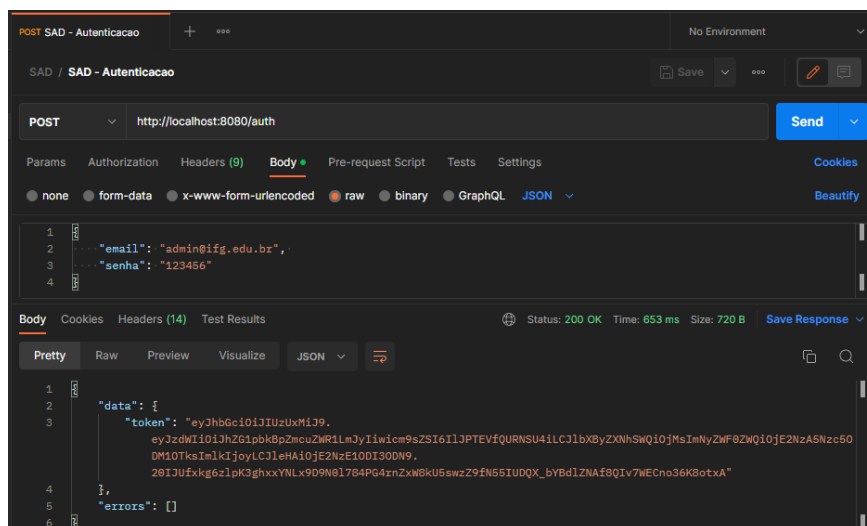


Figura 25 – Método de Autenticação.

Primeiro para que consiga acessar os métodos é necessário a autenticação passando como parâmetro o usuário e senha disponível no banco de dados. Após a chamada do método é retornado o *token* de autenticação que poderá ser utilizado nas demais requisições. Acima a imagem evidencia como é feita a chamada do endereço, com os parâmetros no JSON para a consulta.

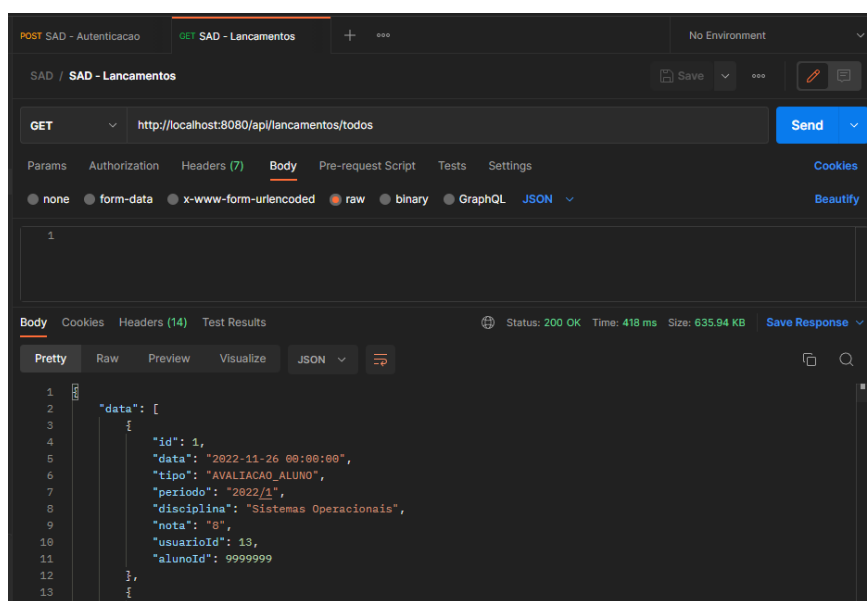


Figura 26 – Método de Consultar Lançamentos.



Após para que consiga efetuar a requisição nos demais métodos é necessário informar o *token* no cabeçalho informando a chave e o valor conforme parametrizado na API, logo efetuando a chamada do endereço com os demais parâmetros necessários conseguirá efetuar a requisição. O exemplo a seguir evidencia a chamada do método de consulta das notas lançadas do SAD armazenadas no banco de dados, para as demais requisições basta seguir conforme a documentação parametrizada no Swagger.

#### 4.5 Interface e Funcionalidades do Sistema de Avaliação Docente (SAD)

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

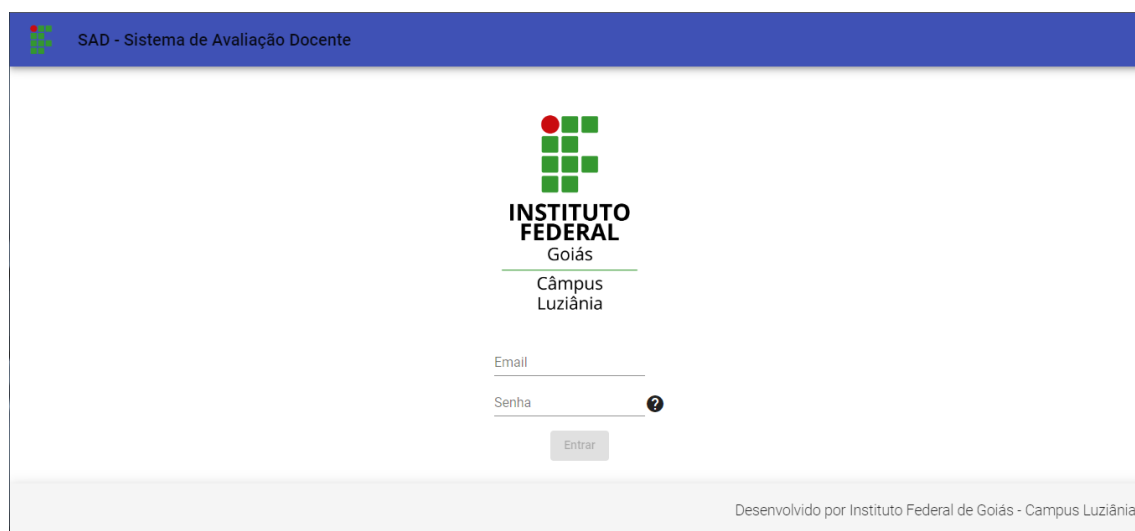


Figura 27 – Interface - Tela de Login.

O SAD possui a tela da Figura 27 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.

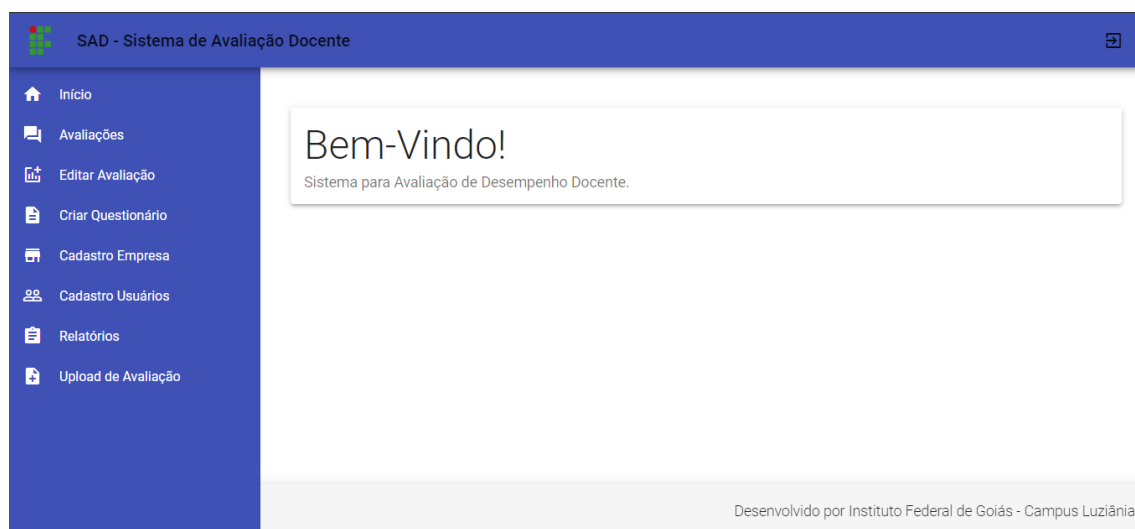


Figura 28 – Interface - Tela Inicial - Adminitrador.

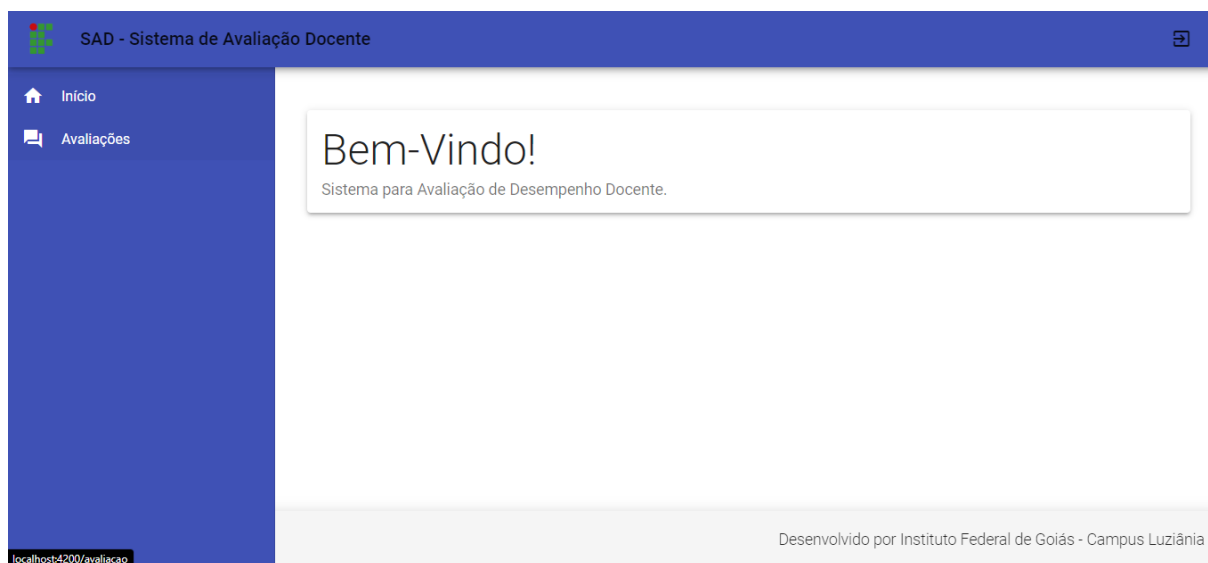


Figura 29 – Interface - Tela Inicial - Discente.

A tela inicial, possui como função recepcionar o usuário que está acessando o SAD, seja ele docente, discente, coordenador ou administrador e desejar uma mensagem que seja bem-vindo. Nessa interface já consegue-se observar que os menus estarão disponíveis de acordo com o nível de acesso de cada usuário.

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 são feitas as avaliações, na primeira tela carrega os dados do avaliador com o nome e campus, juntamente com a data e hora atual e as disciplinas que estão pendentes para avaliar.



Figura 30 – Interface - Tela de avaliações com a seleção da disciplina.

Juntamente é carregado o regimento pertencente às avaliações docentes do Instituto Federal de Goiás, após o usuário possuirá a opção de selecionar a disciplina que carregará o docente que será avaliado, conforme as imagens abaixo.

Essa interface busca ser relativamente fácil de aprender e utilizar, o usuário tem várias abas para

a interação com o sistema, é possível alternar de uma aba para outra. O projeto centrado no usuário é uma abordagem de projeto de interface onde a análise das atividades do usuário é primordial para o sucesso do projeto como um todo. A interação dos usuários e a apresentação de informação foram integradas em uma formulação coerente de interface.

The screenshot shows the 'Avaliação de Desempenho Docente' (Teacher Performance Evaluation) form. The interface has a blue header with the title 'SAD - Sistema de Avaliação Docente' and a sidebar with 'Início' (Home) and 'Avaliações' (Evaluations). The main content area displays a progress bar with 12 steps, the first of which is active. Below the progress bar, the evaluator's name 'Patrick Cavalcante Moraes' is shown. A dropdown menu for 'disciplina' (discipline) is set to 'Álgebra Linear'. The campus is 'Luziânia' and the date is '14/12/2022 09:48:43'. The professor's name is 'Alan Santos Gois'. A note states: 'De acordo com a Res. Consup IFG nº 040/2018, Regimento Geral do IFG, Art. 190-193,198-201, avalie a atuação do Docente na avaliação a seguir, atuante em 2022, atribuindo-lhe nota numa escala de 0 (zero) à 10(dez).' The footer indicates it was developed by 'Instituto Federal de Goiás - Campus Luziânia'.

Figura 31 – Interface - Tela inicial das avaliações.

Após todo o preenchimento do questionário de forma linear, atribuindo uma nota de 0 à 10 em cada resposta, conforme demonstrado na imagem abaixo. 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*.

This screenshot shows the evaluation form with a question. The progress bar now shows the second step as active. The question is 'Questão 2: Assiduidade e pontualidade: Esteve presente regularmente às atividades e cumpre horários de início e fim em sala de aula?'. Below the question is a rating scale from 0 to 10, with radio buttons for each number. The number 9 is selected. The footer remains the same, indicating it was developed by 'Instituto Federal de Goiás - Campus Luziânia'.

Figura 32 – Interface - Tela de avaliações com a opção para preencher a nota.

Caso não seja preenchido algum campo mostrará um alerta informando o campo necessário para o preenchimento, somente após preenchimento de todos os campos conseguirá enviar o formulário. Essa validação é importante, evitando que ocorra registro com informações pendentes no momento do envio da avaliação.

Efetuada o envio da avaliação de desempenho do docente selecionado, o sistema efetuará uma validação e disponibilizará na lista de docentes do usuário, somente os docentes que estão pendente a ser avaliados, uma validação que também é importante, pois evita a duplicidade de registros armazenados no banco de dados.



Figura 33 – Interface - Tela de avaliações com o alerta de pendência de campo preenchido.

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. A tela oferece algumas opções de ação como a de edição e exclusão da nota lançada 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.

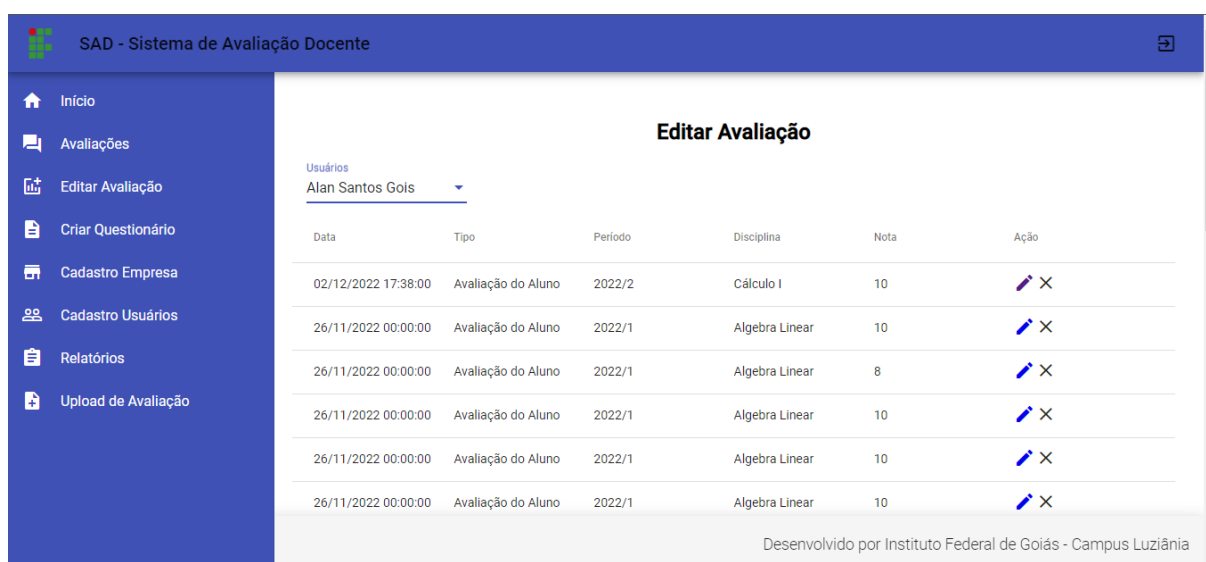


Figura 34 – Interface - Tela de listar notas.

A paginação permite que as informações sejam quebradas entre páginas, com o intuito de evitar uma enorme quantidade de dados e ter uma página enorme, cheia de registros. Já a ordenação, consegue organizar os resultados de acordo com uma ou mais colunas da tabela, podendo definir a ordem dos resultados como crescente ou decrescente.

Qtd. por página: 25    1 - 25 / 81    < >

Figura 35 – Interface - Paginação.












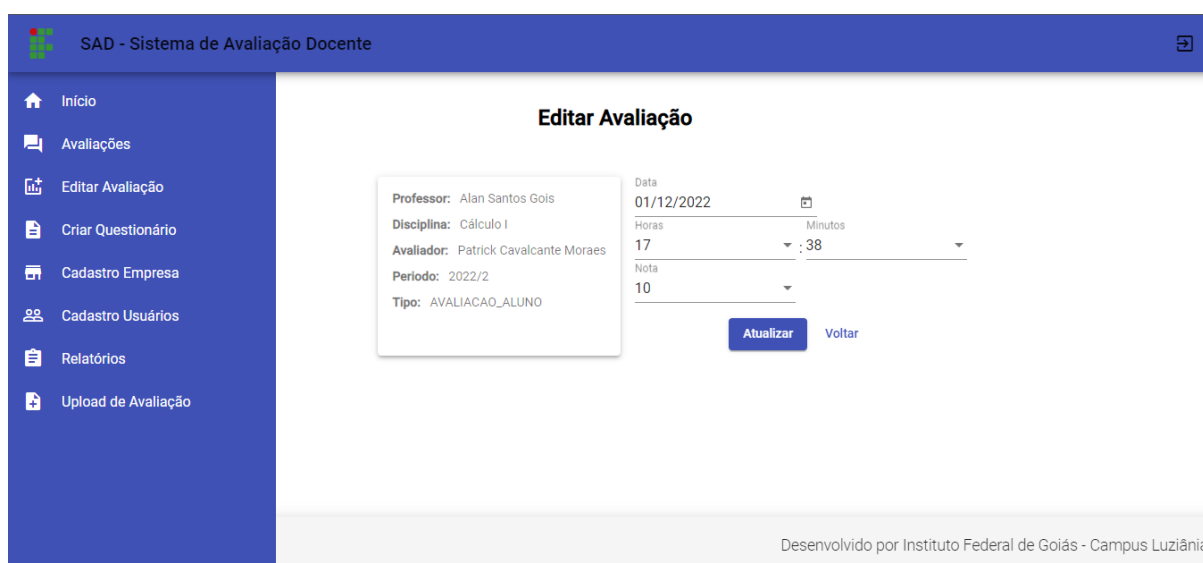
Data ↓	Tipo	Período	Disciplina	Nota	Ação
02/12/2022 17:38:00	Avaliação do Aluno	2022/2	Cálculo I	10	 
26/11/2022 00:00:00	Avaliação do Aluno	2022/1	Algebra Linear	10	 
26/11/2022 00:00:00	Avaliação do Aluno	2022/1	Algebra Linear	8	 
26/11/2022 00:00:00	Avaliação do Aluno	2022/1	Algebra Linear	10	 
26/11/2022 00:00:00	Avaliação do Aluno	2022/1	Algebra Linear	10	 
26/11/2022 00:00:00	Avaliação do Aluno	2022/1	Algebra Linear	10	 

Figura 36 – Interface - Ordenação.

A ordenação acima demonstra o campo da data sendo ordenado de forma crescente. Caso seja chamada a função de edição, retornará a tela que o administrador conseguirá fazer as alterações necessárias nos campos data, horas, minutos e nota juntamente com as demais informações para consulta.



**SAD - Sistema de Avaliação Docente**

**Editar Avaliação**

Professor: Alan Santos Gois  
 Disciplina: Cálculo I  
 Avaliador: Patrick Cavalcante Moraes  
 Período: 2022/2  
 Tipo: AVALIACAO\_ALUNO

Data: 01/12/2022  
 Horas: 17  
 Minutos: 38  
 Nota: 10

**Atualizar** **Voltar**

Desenvolvido por Instituto Federal de Goiás - Campus Luziânia

Figura 37 – Interface - Tela de editar notas.

Os cadastros que somente o usuário com privilégio de administrador pode acessar, estão nas interfaces cadastrar empresa e usuários. O cadastrar empresa possui uma classe em JavaScript de validação do CNPJ e CPF, onde são aceitos somente CNPJ e CPF válidos. 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 válidos. 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.

The screenshot displays the 'Cadastro de Empresa' (Company Registration) form within the SAD - Sistema de Avaliação Docente application. The interface features a blue sidebar on the left with navigation links: Início, Avaliações, Editar Avaliação, Criar Questionário, Cadastro Empresa, Cadastro Usuários, and Upload de Avaliação. The main content area is titled 'Cadastro de Empresa' and contains the following fields: CNPJ, Razão Social, CPF, Nome, Email, and Senha (password). The password field includes a question mark icon for password requirements. At the bottom of the form are two buttons: 'Cadastrar' (Register) and 'Voltar' (Back). The footer of the page indicates the system was developed by Instituto Federal de Goiás - Campus Luziânia. The browser address bar shows 'localhost:4200/cadastro-pf'.

Figura 38 – Interface - Tela de cadastrar empresas.

The screenshot displays the 'Cadastro de Usuário' (User Registration) form within the SAD - Sistema de Avaliação Docente application. The interface features a blue sidebar on the left with navigation links: Início, Avaliações, Editar Avaliação, Criar Questionário, Cadastro Empresa, Cadastro Usuários, Relatórios, and Upload de Avaliação. The main content area is titled 'Cadastro de Usuário' and contains the following fields: CNPJ, CPF, Funcao (Function) with a dropdown arrow, Nome, Email, and Senha (password). The password field includes a question mark icon for password requirements. At the bottom of the form are two buttons: 'Cadastrar' (Register) and 'Voltar' (Back). The footer of the page indicates the system was developed by Instituto Federal de Goiás - Campus Luziânia.

Figura 39 – Interface - Tela de cadastrar usuários

A fim de apresentar os resultados de acordo com a necessidade e uma forma fácil para extração das informações, foi criado o módulo que efetua a geração dos relatórios do sistemas. Disponibilizando

os relatórios do cadastro de usuário, das disciplinas e das notas lançadas para os docentes, possuindo também a função para extração do relatório em CSV.



The screenshot displays the 'SAD - Sistema de Avaliação Docente' web application. On the left is a blue sidebar with navigation icons and labels: 'Início', 'Avaliações', 'Editar Avaliação', 'Criar Questionário', 'Cadastro Empresa', 'Cadastro Usuários', 'Relatórios', and 'Upload de Avaliação'. The main content area is titled 'Relatórios'. It features a dropdown menu labeled 'Selecione o relatório: \*' with 'Lançamentos' selected, and a blue 'Exportar CSV' button. Below this is a table with the following data:

Id	Professor	Aluno	Nota	Disciplina	Período	Tipo	Data
1	Cristiane Roberta da Silva	Carga de Dados	8	Sistemas Operacionais	2022/1	AVALIACAO_ALUNO	2022-11-26 00:00:00
2	Cristiane Roberta da Silva	Carga de Dados	7	Sistemas Operacionais	2022/1	AVALIACAO_ALUNO	2022-11-26 00:00:00
3	Cristiane Roberta da Silva	Carga de Dados	10	Sistemas Operacionais	2022/1	AVALIACAO_ALUNO	2022-11-26 00:00:00

At the bottom right of the interface, it says 'Desenvolvido por Instituto Federal de Goiás - Campus Luziânia'.

Figura 40 – Interface - Relatório das notas lançadas para os docentes.

Sendo assim, os relatórios gerenciais tomaram um papel muito importante e são a principal forma de analisar as informações da Avaliação de Desempenho Docente, possibilitando uma tomada de decisão mais ágil, reduzindo riscos e expondo oportunidades.

#### 4.6 Testes

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

Foi efetuado também na primeira etapa uma reunião para efetivação dos testes de aceitação juntamente com os integrantes da CPPD, com um protótipo já elaborado foram efetuadas abordagens de aceitação, apresentando o protótipo e verificando as necessidades e possíveis melhorias. As necessidades levantadas trouxe uma melhor visão da regra de negócio da instituição, e uma perspectiva dos usuários se o projeto conseguiria atender as necessidade da comissão.

Após a concretização dos pontos de função levantados anteriormente, foi apresentado o protótipo final, demonstrando todas as funcionalidades desenvolvidas para o sistema SAD. Essa abordagem foi estabelecida para que consiga efetivar o que foi proposto e disponibilizar o sistema para a implantação no Instituto Federal de Goiás.

Com a carga parcial dos dados feita pela *API RESTful*, os membros da CPPD conseguiram visualizar o que foi proposto e desenvolvido, visando trazer um cenário para o ponto de vista que o usuário utilizará, corrigindo eventuais problemas que poderiam ocorrer após á implementado na instituição e com início da utilização do SAD.

Durante o processo de desenvolvimento foi utilizado o conceito de TDD, que após a criação de cada classe foi implementado um classe teste. Utilizando os recursos do *@SpringBootTest* com o mecanismo do JUnit para os testes unitários. Para os testes de interação foi implementado armazenamento de persistência H2 na memória que faz a consulta somente dos registros armazenados na memória criados para o teste da classe.

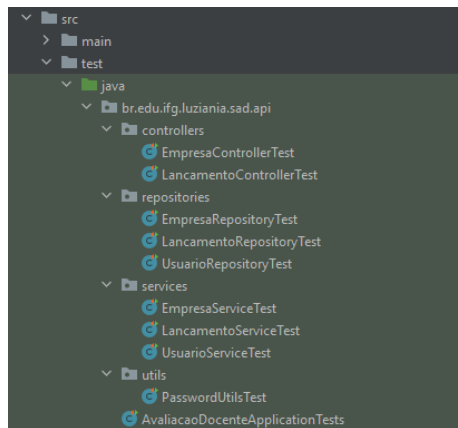


Figura 41 – Estruturação das Classes de Teste.

Na Figura 41 é demonstrado a estrutura das classes teste para *API RESTful* do SAD, as classes foram implementadas e após a execução retorna no terminal o resultado da obtido. Quando os testes forem executados em modo gráfico, os métodos testados podem apresentar os seguintes resultados: verde para sucesso, roxo para exceção e vermelho para falha.

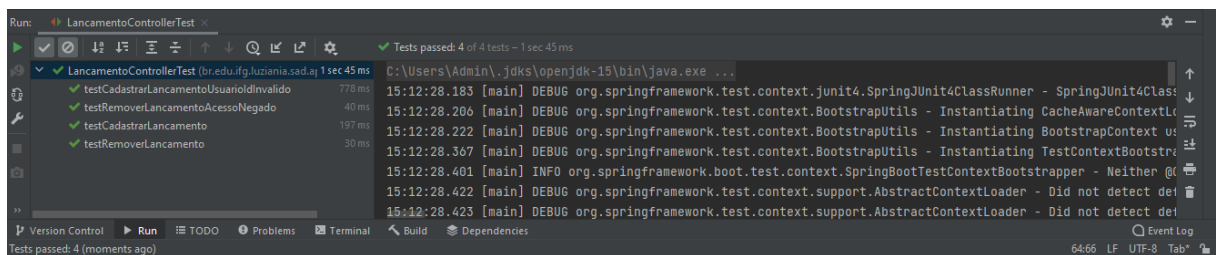


Figura 42 – Execução da Classe de Teste LancamentoControllerTest.

Para os testes de desempenho da *API Restful*, foi simulado múltiplos acessos em paralelo, e obteve algumas métricas sobre a execução. Com Apache AB que é um executável que vem junto com o servidor Apache HTTP, e é ideal para testes de performance, e disponibiliza via linha de comando realizar requisições HTTP, permitindo configurar a quantidade de requisições, quantidade de requisições em paralelo, espaço de tempo para realizar as requisições, entre outros. Sendo que no final, ele exibe um relatório com o plano de execução, demonstrando dados e métricas do que foi executado.

O Apache AB está localizado dentro do diretório bin da instalação do Apache, e é acessado via linha de comando, com a utilização do terminal do XAMPP. Navegando até o diretório bin dentro da instalação do Apache HTTP Server, foi executado o comando da Figura 43, o comando executará dez mil requisições a URL informada, sendo que com uma concorrência de mil ao mesmo tempo, ao término da execução será exibido o relatório.

```
Admin@HP-PC c:\xampp\apache\bin
# ab -n 10000 -c 100 http://localhost:8080/api/disciplinas
```

Figura 43 – Execução do comando para dez mil requisições.

No relatório de desempenho da Figura 44 é a evidencia a execução das requisições, que são divididas entre mil até completar as dez mil requisições. São mostrado alguns dados, como: nome do host do servidor, porta do servidor, caminho do documento, comprimento do documento, nível de simultaneidade,



tempo gasto para testes, pedidos completos, solicitações com falha, total transferido, HTML transferido, solicitações por segundo, tempo por solicitação e taxa de transferência entre outros.

```

This is ApacheBench, Version 2.3 <$Revision: 1879490 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking localhost (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests

Server Software:
Server Hostname:      localhost
Server Port:          8080

Document Path:        /api/disciplinas
Document Length:       204 bytes

Concurrency Level:     100
Time taken for tests:   5.208 seconds
Complete requests:     10000
Failed requests:        0
Non-2xx responses:     10000
Total transferred:     4910000 bytes
HTML transferred:      2040000 bytes
Requests per second:   1920.29 [#/sec] (mean)
Time per request:      52.075 [ms] (mean)
Time per request:      0.521 [ms] (mean, across all concurrent requests)
Transfer rate:          920.77 [Kbytes/sec] received

Connection Times (ms)
      min    mean[+/-sd] median    max
Connect:    0      0  0.7      0      23
Processing:  1    51 114.9     33    1909
Waiting:    1    48 112.3     30    1908
Total:      1    52 114.9     33    1909

Percentage of the requests served within a certain time (ms)
 50%    33
 66%    39
 75%    44
 80%    47
 90%    60
 95%    95
 98%   411
 99%   733
100%  1909 (longest request)

```

Figura 44 – Relatório de Desempenho.

A partir do relatório de desempenho da *API Restful*, pode-se obter que o tempo percorrido de 5.208 segundos para a execução do teste de dez mil requisições que foi adequado para a aplicação, garantindo assim sua qualidade. Assim pode-se prever que a aplicação conseguirá executar dez mil requisições em um tempo hábil sem nenhum tipo de stress ou falha e que os demais dados do relatório estão coerentes para um sistema.

## 5 CONCLUSÃO

A avaliação de desempenho docente demonstra que o Instituto Federal de Goiás está atento com a garantia de desempenho e progressão de seus docentes. Esse trabalho apresentou o Sistema de Avaliação de Desempenho Docente (SAD) para atender com mais qualidade esse processo de avaliação, evidenciando também que o modelo aplicado atualmente já não agregaria resultados satisfatórios diante o cenário atual.

A solução proposta pode ser implantada para resolver a necessidade de um sistema que contemple as três avaliações do docente, e que consiga integrar essas informações em dados concretos e de fácil manuseio para a Comissão Permanente de Pessoal Docente unificando todo o processo em uma única aplicação.

A evolução da tecnologia permitiu que os sistemas avançassem cada vez mais no que se refere à solução de problemas e se tornaram mais acessíveis às instituições. O objetivo deste projeto é justamente aproveitar os benefícios da tecnologia para a criação do SAD que poderá fornecer apoio às decisões tornando-as mais eficientes e eficazes, onde o administrador conseguiria coletar e controlar todas as informações e conseguirá utilizá-las da maneira mais adequada.

Assim, o que se define como um processo bem sucedido é aquele que cria mudanças claras e duráveis de comportamento e/ou desenvolvimento de habilidades, de modo a promover um incremento na efetividade da organização. Pode-se então concluir que o SAD para avaliação de desempenho docente é essencial para a saúde da instituição e para a uma melhor qualidade de seu serviço.

### 5.1 Trabalhos futuros

A contemplação do Sistema de Avaliação Docente dado ao desenvolvimento do SAD conforme abordado neste trabalho, daria-se em sua utilização pelo Instituto Federal de Goiás, logo para que se consiga implementar a ferramenta deve-se seguir algumas aprovações como a da CPPD que já foi avaliada e efetivada positivamente, em seguida pode ser feita a avaliação por parte do Diretoria de Tecnologia da Informação(DTI) do IFG para sua implementação. O SAD possui algumas funcionalidades que ainda necessitam ser desenvolvidas, como a integração do *Active Directory*(AD) para o controle de acessos, uma das melhorias que pode ser feita é a criação da integração do SAD com o AD.

O SAD também é uma ferramenta que foi proposta para ser facilmente adaptada para outros sistemas de avaliações de desempenho. Como trabalhos futuros, podem ser desenvolvidos a partir desse sistema outros sistemas de apoio à decisão que pode ser implantado em diversas instituições públicas ou privadas.

## REFERÊNCIAS

- AMUNDSEN, O. *KANBAN e SCRUM, combinados!* 2010.  
<https://olemortenamundsen.wordpress.com/2010/03/19/kanban-and-scrum-combined/>. Acessado em 14 Set, 2022.
- 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.
- BALDUINO, P. *Dominando JavaScript com jQuery*. [S.l.]: Cada do Código, 2012.
- BORDALLO, B. O que é banco de dados e qual o seu papel na hospedagem de sites. p. 01, 02 2021.
- BRASIL, D. R. *Plano de Carreiras e Cargos de Magistério Federal*. 2012.  
[http://www.planalto.gov.br/ccivil\\_03/a/to2011-2014/2012/lei/l12772.htm](http://www.planalto.gov.br/ccivil_03/a/to2011-2014/2012/lei/l12772.htm). Acessado em 05 Dez, 2022.
- BRASIL, F. H. C. *Estabelece as diretrizes e bases da educação nacional*. 1996.  
[https://www.planalto.gov.br/ccivil\\_03/leis/l9394.htm](https://www.planalto.gov.br/ccivil_03/leis/l9394.htm). Acessado em 17 Set, 2022.
- BRASIL, M. T. *Restuturação do Plano de Carreiras e Cargos de Magistério Federal*. 2013.  
[https://www.planalto.gov.br/ccivil\\_03/a/to2011-2014/2013/lei/l12863.htm](https://www.planalto.gov.br/ccivil_03/a/to2011-2014/2013/lei/l12863.htm). Acessado em 05 Dez, 2022.
- BRESSER-PEREIRA, L. C. *Gestão do setor público: estratégia e estrutura para um novo Estado*. [S.l.]: Fundação Getúlio Vargas: 21-38, 2006.
- CAVALCANTI, T. Mapeamento do modelo entidade relacionamento (er) para o modelo relacional. 01 2015.
- CENTENARO, J. *Desenvolvimento de um Software Web para Gerenciamento de Requisitos de Software*. 2014. Monografia (Especialização em Desenvolvimento de Sistemas para Internet e Dispositivos Móveis), UTFRR (Universidade Tecnológica Federal do Paraná), Francisco Beltrão, Brazil.
- 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.
- DIAS, W. *Documentando sua API Rest com Swagger*. 2021.  
<http://www2.decom.ufop.br/terralab/documentando-sua-api-rest-com-swagger/>. Acessado em 27 Set, 2022.
- DIGITE. *O que é Kanban?* 2022. <https://www.digite.com/pt-br/kanban/o-que-e-kanban/>. Acessado em 21 Jan, 2022.
- DOCENTE, C. P. de P. *Comissão Permanente de Pessoal Docente*. 2021.  
<http://www.ifg.edu.br/comissoes/cppd>. Acessado em 02 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.
- FERREIRA, A. *Regimento Geral*. 2018. <http://www.ifg.edu.br/component/content/article/70-ifg/a-instituicao/conselhos/conselho-superior/209-consup?showall=start=4>. Acessado em 05 Dez, 2022.
- FOUNDATION, O. *Node JS*. 2021. <https://nodejs.org/en/>. Acessado em 08 Jan, 2022.
- HIGA, P. *O que é XAMPP e para que serve*. 2012. <https://www.techtudo.com.br/noticias/2012/02/o-que-e-xampp-e-para-que-serve.ghtml>. Acessado em 13 Dez, 2022.

- JONATAN, R. *Metodologia Ágil: descubra o que é, os principais tipos + 6 funções que ela cumpre nos projetos de qualquer área*. 2020. <https://resultadosdigitais.com.br/marketing/metodologia-agil/>. Acessado em 17 Dez, 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.
- LAUBE, K. *Swagger na prática*. 2018. <https://klauslaube.com.br/2018/03/15/swagger-na-pratica.html>. Acessado em 15 Set, 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.
- MANOEL. *JUnit Tutorial*. 2009. <https://www.devmedia.com.br/junit-tutorial/1432>. Acessado em 09 Dez, 2022.
- MARC, M. *Trello for Project Management*. [S.l.]: amazonKindle, 2014.
- MENDES L. COSTA, R. L. R. *O Gerenciamento de Requisitos e a sua Importância Em Projetos de Desenvolvimento de Software*. 2022. <http://www.gestaouniversitaria.com.br/>. Acessado em 13 Set, 2022.
- NOLETO, C. *Angular Material: como instalar e usar? Primeiros passos!* 2022. <https://blog.betrybe.com/framework-de-programacao/angular-material/>. Acessado em 21 Dez, 2022.
- OLIVEIRA 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.
- PLINIO, V. *Caso de Uso - Include, Extend e Generalização*. 2014. <https://www.ateomomento.com.br/caso-de-uso-include-extend-e-generalizacao/>. Acessado em 20 Dez, 2022.
- REHKOPF, M. *O que é um painel Kanban?* 2019. <https://www.atlassian.com/br/agile/kanban/boards>. Acessado em 07 Dez, 2022.
- REITORIA, I. F. de G. *RESOLUÇÃO 45/2021 - REI-CONSUP/REITORIA/IFG*. 2021. [http://ifg.edu.br/attachments/article/209/RESOLUÇÃO%2045%202021%20-%20REI-CONSUP\\_REITORIA\\_IFG.pdf](http://ifg.edu.br/attachments/article/209/RESOLUÇÃO%2045%202021%20-%20REI-CONSUP_REITORIA_IFG.pdf). Acessado em 17 Set, 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.
- RODRIGUES, J. *Testando serviços Web API com Postman*. 2014. <http://www.linhadecodigo.com.br/artigo/3712/testando-servicos-web-api-com-postman.aspx>. Acessado em 13 Dez, 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.
- SILVA, J. *Regimento Geral*. 2012. <https://www.ifg.edu.br/attachments/article/123/regimento122015.pdf>. Acessado em 05 Dez, 2022.

- SOUZA, I. *phpMyAdmin: saiba o que é e aprenda como instalar e criar um banco de dados nele*. 2020. <https://rockcontent.com/br/blog/phpmyadmin/>. Acessado em 26 Set, 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>. 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>. Acessado em 02 Jan, 2022.
- VENTURA, P. *Entendendo definitivamente o que é um Caso de Uso*. 2011. <https://www.ateomomento.com.br/o-que-e-caso-de-uso/>. Acessado em 13 Set, 2022.
- WIEGERS, K. *Software requirements: Practical techniques for gathering and managing requirements throughout the product development cycle*. 2nd edition. 01 2003.
- XAVIER P. PIRES, J. F. F. C. *Projeto de Avaliação Institucional Avaliação Interna (Auto-Avaliação)*. [S.l.]: Bahia, 2005.