PAE docs: Plataforma de Gerenciamento do Programa de Assistência Estudantil do IFSP Campus Hortolândia

Daniel Luz Alves [1], Gustavo Bartz Guedes[1]

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Campus Hortolândia CEP: 13183-250 – Hortolândia - SP – Brasil

danielluz.alves@outlook.com, gubartz@ifsp.edu.br

Abstract. *The Student Assistance Program focuses on equal opportunities for all low-income students who are enrolled in all courses offered at Federal Institute of São Paulo, Brazil. It is an extensive and continuous process divided into stages and activities, which includes several analyzes and evaluations, currently done manually. This work presents, PAE docs, a platform that focus on assist the Social-pedagogical Coordination team regarding the tasks related to analysis, validation and cataloging documents associated with student's enrollment in the student aid process. A REST API and a website client, both developed in PHP, composes PAE docs architecture. In addition, an application for Android platform was developed to provide the student's access to the platform.*

Resumo. *O Programa de Assistência Estudantil (PAE) tem como objetivo proporcionar igualdade de oportunidades para os alunos de baixa renda que estejam matriculados e frequentando os cursos oferecidos pelos Institutos Federais de Ensino. O processo de análise para concessão da assistência estudantil é extenso e contínuo sendo dividido em etapas e atividades que incluem diversas avaliações e análises. Grande parte dessas atividades são manuais. Este trabalho apresenta a PAE docs, uma plataforma que auxilia a equipe da Coordenadoria de Assistência Estudantil nas atividades relacionadas a análise, validação e catalogação dos documentos associados a inscrição do aluno no processo de assistência estudantil. A arquitetura da plataforma PAE docs é composta por uma API Rest e um cliente web, ambos desenvolvidos em PHP. A plataforma também possui um cliente Android que foi desenvolvido para prover acesso aos estudantes.*

1. Motivação e Introdução

Com o decreto 7.234/2010 (Brasil, 2010), o Programa Nacional de Assistência Estudantil (PNAES) é elaborado para articulação entre ensino, pesquisa e extensão, com o objetivo de atender os discentes de curso de graduação presencial das Instituições Federais (IFEs) em nove áreas identificadas como fatores de vulnerabilidade socioeconômica: moradia, alimentação, transporte, saúde, inclusão digital, cultura, esporte, creche, apoio pedagógico (acesso, participação e aprendizagem de estudantes com deficiência, com transtornos globais do desenvolvimento e altas habilidades e superdotação). As IFEs possuem autonomia para fixar os critérios socioeconômicos baseado em fatores locais ou regionais.

O Programa de Assistência Estudantil (PAE) do Instituto Federal de São Paulo (IFSP), campus Hortolândia, possui apenas parte das atividades referentes ao processo de inscrição atendida pelo Sistema Unificado de Administração Pública (SUAP). Pelo SUAP o aluno preenche os formulários socioeconômico necessários para iniciar sua inscrição, que será disponibilizado para a Coordenadoria Socio pedagógica (CSP), para análise e solicitação de documentos comprobatórios para validação dos formulários.

O objetivo deste projeto foi desenvolver a *PAE docs*, uma plataforma que tem como foco auxiliar a CSP nas atividades de validação, análise e catalogação dos documentos relacionados ao PAE. Adicionalmente, a plataforma permite que os discentes façam envios dos documentos requisitados por um sistema *web* ou por uma aplicação *Android*.

2. Referencial Teórico

Esta seção apresenta as referências teóricas que serviram de base para a execução deste trabalho e que foram importantes para entendimento da solução e como desenvolvê-la. Na seção 2.1 é apresentada a Política de Assistência Estudantil, base de aplicação do Programa de Assistência Estudantil do Campus Hortolândia; a seção 2.2 e 2.3 abordam a arquitetura orientada a serviços, com o conceito de webservice e REST, respectivamente; a seção 2.4 mostra o *Kanban*, que auxilia na visualização das tarefas e progresso de cada projeto; já a seção 2.5 trata dos padrões de projeto (*Software Design Pattern*); por fim a seção 2.6 apresenta os conceitos de Sistemas de Gerenciamento de Documentos.

2.1. Política de Assistência à Educação no Brasil

Na década de 30, juntamente com o processo de organização das faculdades e universidades no Brasil é possível observar o início das práticas de assistência estudantil. No art. 108 da Lei Orgânica do Ensino Superior, Decreto 19.851/1931 (Brasil,1931), é reconhecida a necessidade da providência e beneficência estudantil. Na constituição de 1934, no §2º do Art. 157, é estabelecido que parte dos fundos para educação da União, Distrito Federal e Estados fossem aplicados ao apoio em auxílios aos estudantes necessitados.

De acordo com Vasconcelos (2010), entre as décadas de 50 a 70, há o surgimento das universidades federais no Brasil, seguido por uma progressiva descentralização do acesso ao ensino superior que por consequência amplia a diversidade socioeconômica dos alunos que fazem parte do corpo estudantil.

Somente em 1988 a assistência estudantil é reconhecida com um direito social de todos. Segundo a constituição de 1988, "[...] Art. 208 O dever do Estado com educação escolar pública será efetivado mediante a garantia de: [...] VIII - atendimento ao educando, em todas as etapas da educação básica, por meio de programas suplementares de material didático-escolar, transporte, alimentação e assistência à saúde. "

Posteriormente, em 2010, surge o Programa Nacional de Assistência Estudantil com objetivo de criar meios de garantir a permanência do aluno relacionando critérios socioeconômicos para traçar o perfil e nível de vulnerabilidade socioeconômica do discente e segue como referência para programas de assistência estudantil em Instituições Federais de todo o Brasil**.**

2.2. Webservice

Segundo Adorno (2010) o *webservice* surgiu como uma evolução dos modelos de computação distribuída na segunda metade dos anos 90.

Trata-se de uma tecnologia de integração de sistemas para o desenvolvimento de softwares ou componentes de software que são capazes de interagir com outras aplicações, independente da linguagem de programação ou sistema operacional. A troca de dados é realizada por meio de requisições.

2.3. Representational State Transfer (REST)

Descrito por Fielding (2000), *REST* é uma abstração dos elementos arquiteturais em um sistema de hipermídia distribuído, que possui uma reunião e fusão de mídias em um ambiente computacional. O seu desenvolvimento está associado as funções dos componentes que fazem parte do sistema, na aplicação de restrições das interações com outros componentes e na interpretação de requisições *HTTP* (Protocolo de Transferência de Hipertexto), como *GET, POST, PATCH, PUT* para processamento e preparo de resposta contendo dados significativos. Assim gerando dados para cada requisição, como exemplificado na Figura 1.

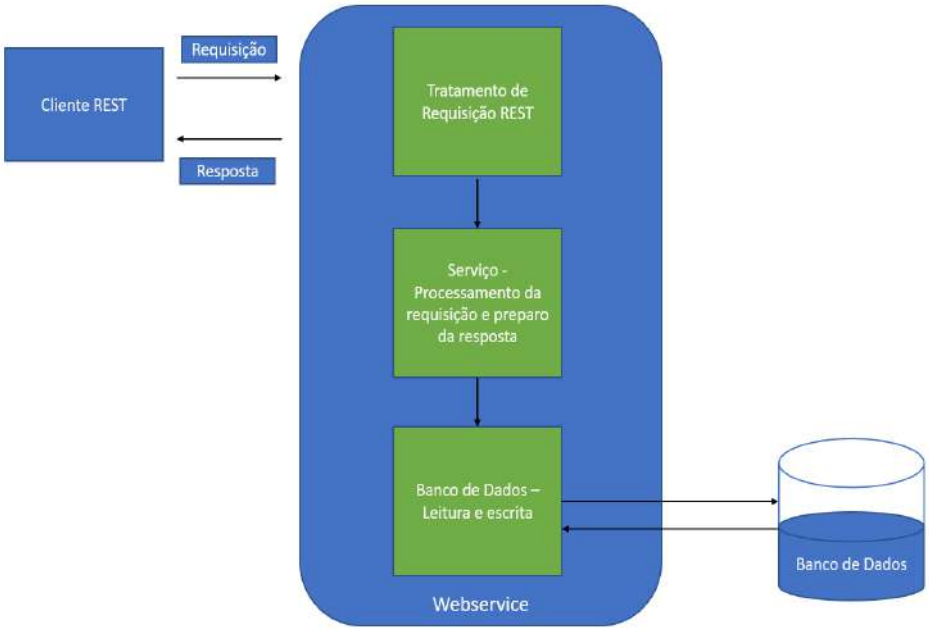

Figura 1. Conceitos arquiteturais REST aplicado no *webservice*. Fonte: Criado pelo autor

2.4. *Kanban*

Foi criado por Taiichi Ohno, o termo *Kanban* pode ser traduzido como "cartão visual". Trata-se de uma metodologia que procura balancear, visualizar e melhorar o tratamento das demandas e assim possibilitar uma melhor visibilidade dos impedimentos ou dificuldades que podem surgir durante a execução das tarefas [OHNO,1988]. Um quadro *Kanban* é divido por colunas que representam as atividades que compõem o workflow do projeto. Cada cartão passa pelas colunas conforme as tarefas relacionadas são realizadas. O *workflow* é variável em complexidade, dependendo assim de como será aplicado no projeto, a Figura 2 exemplifica um *workflow* que começa pela fase "Para fazer" e as atividades relacionadas a um cartão serão consideradas completas quando o cartão estiver localizado na coluna "Feito".

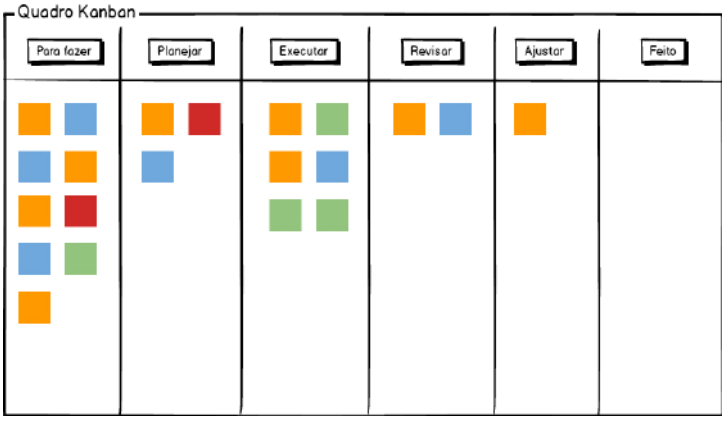

Figura 2. Modelo quadro Kanban. Fonte: Pluga.co, 2018

Com o *Kanban* é possível organizar o desenvolvimento pelo controle das atividades em progresso, estabelecendo que uma nova atividade só deve ser iniciada caso exista disponibilidade suficiente no desenvolvimento, assim permitindo maior possibilidade de sucesso na tentativa de evitar sobrecargas no projeto.

No contexto da Tecnologia da Informação, por meio da análise do fluxo dos cartões criados, é possível identificar quando é necessário que ocorra otimização e priorização das atividades relacionadas ao projeto, seja no nível de gerenciamento, desenvolvimento, validações ou testes.

2.5. *Software Design Pattern*

Trate-se de padrões de projetos, divido em três categorias: Criação, Estrutural e Comportamental, descritos por Gamma (1994). O principal objetivo do uso de *Design Patterns* é acelerar o processo de desenvolvimento por meio de paradigmas de desenvolvimentos e boas práticas que já são testados e aprovados.

2.6. Sistema de gerenciamento de documentos

Um sistema de gerenciamento de arquivos (SGD) fornece armazenamento, versionamento, metadados, segurança, indexação e funcionalidades para recuperação de documentos. Ao aplicar conceitos do workflow é possível melhorar o desempenho do processo pela troca e validação de informações [GREEN,1993].

2.7 Desenvolvimento Incremental

O desenvolvimento incremental possui atividades intercaladas com o objetivo de obter *feedback* mais rápido, possui um custo menor de acomodar mudanças de requisitos comparado a um modelo em cascata [SOMMERVILLE,2011].

3. Metodologia

O desenvolvimento do projeto teve início com o levantamento de requisitos, realizado com a Assistente Social responsável pelo PAE, possibilitando assim identificar os requerimentos e conhecer as etapas e regras de negócio.

De acordo com Engholm (2013) “o levantamento de requisitos deve verificar os objetivos do projeto e todas as particularidades a serem disponibilizadas pelo sistema a ser desenvolvido, com a preocupação de que o sistema atenda às necessidades e expectativas [...]”.

Por meio de entrevistas não estruturadas foi apontado como a maior dificuldade do processo a falta de um sistema ou aplicação que auxiliasse a equipe a gerenciar, validar e catalogar os documentos fornecidos pelos alunos, atualmente trata-se de um processo manual por parte do aluno e da CSP.

No processo atual de inscrição, exemplificado na Figura 3, a primeira etapa ocorre quando o aluno preenche ou atualiza o formulário socioeconômico no sistema SUAP e o associa ao edital aberto no sistema, caso não existam pendências ou discrepâncias no preenchimento a CSP solicita ao aluno os documentos para validação. Esses documentos deverão então ser entregues na CSP até uma data limite.

Caso o aluno não faça a entrega, o processo de inscrição é encerrado, do contrário, a Assistente Social deve validar os documentos e anexá-los na pasta do aluno, armazenada em

um armário. No processo de entrega de documentos, os alunos geram cópias em papel ofício, pegam filas e obrigatoriamente precisam fazer a entrega pessoalmente na sala da CSP.

Atualmente a equipe da CSP é composta apenas por 1 assistente social e o PAE possui uma média de 400 inscritos por edital e cada inscrição possui em média de 5 a 6 documentos a serem entregues e validados.

Com a definição dos requisitos e entendimento do negócio, ocorreu um amadurecimento em relação os requerimentos e o que precisava ser construído para atender as necessidades da Coordenadoria Socio pedagógica, assim o projeto foi divido nas seguintes fases:

1. Entendimento do domínio para modelagem em banco de dados e aplicações.
2. Codificação.

Cada fase foi dividida em subfases menores as quais estavam associadas com um número de tarefas. Ao final de cada etapa foram geradas versões funcionais parciais para validações e prover melhor visão do progresso da solução.

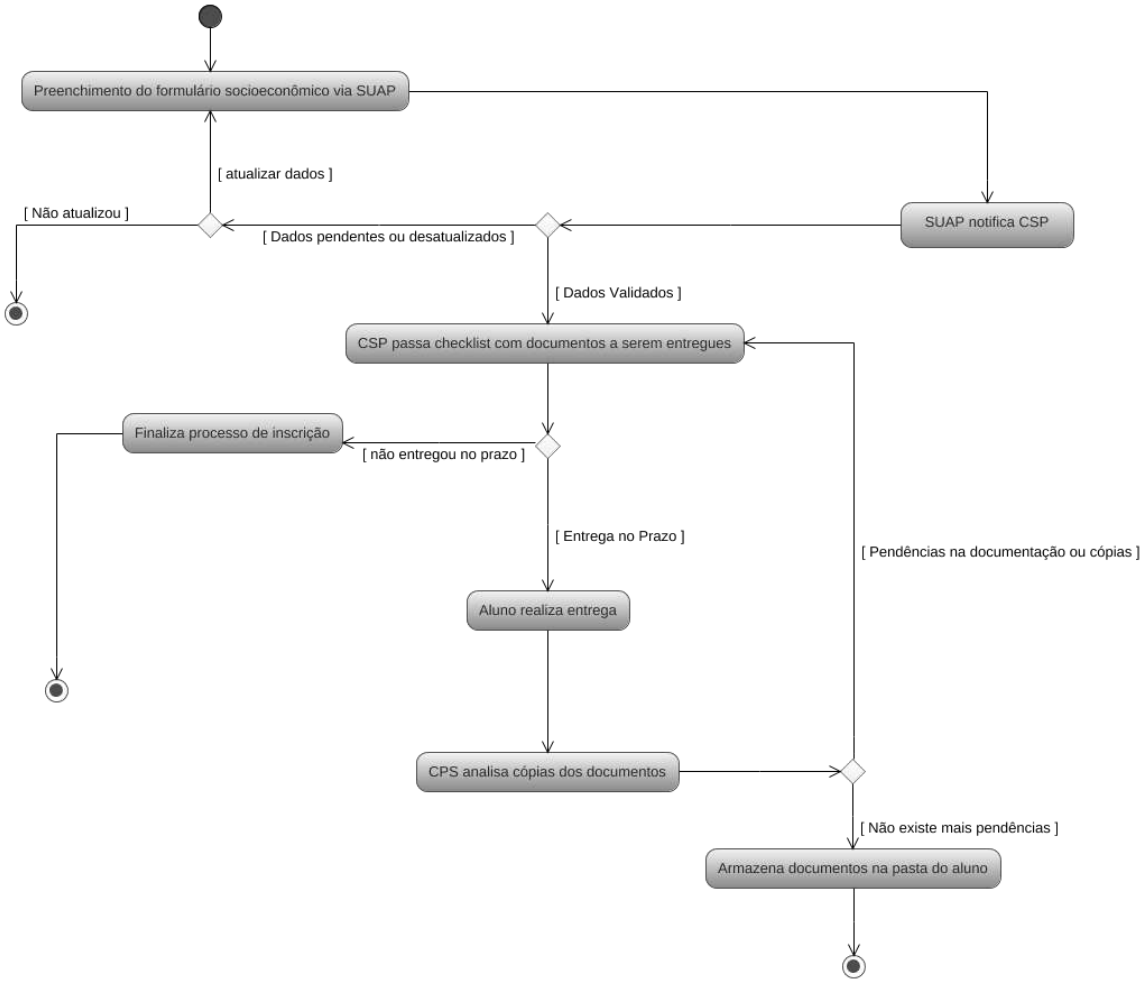

Figura 3. Diagrama de atividade PAE. Fonte: Criado pelo autor.

4. Desenvolvimento

No projeto o *Kanban* foi utilizado e integrado ao *Github* para auxiliar na visualização e definição das atividades. A Figura 4, mostra parte do quadro do desenvolvimento do *webservice*. Esse quadro foi dividido em 3 colunas que representam o workflow do projeto em questão: "*To Do*" (Para fazer, não entrou em desenvolvimento), "*In progress*" (Executando, em desenvolvimento) e "*Done*" (Feito, desenvolvimento finalizado).

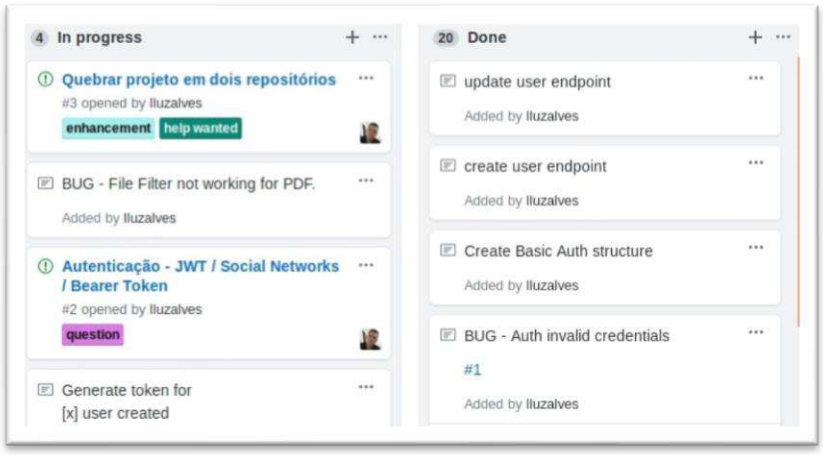

Figura 4. Quadro Kanban para desenvolvimento do projeto. Fonte: Criado pelo autor

No desenvolvimento da aplicação e do cliente web, foi utilizado a linguagem de programação PHP com uso do *Slim Framework*. O *Slim* é um microframework que facilita o desenvolvimento de aplicações web e aplicações *REST*. Trata-se de um *dispatcher*, responsável pelo redirecionamento para a execução de requisições. Ao receber um pedido de um cliente invoca uma rota associada e retorna uma resposta por meio do Protocolo de Transferência de Hipertexto (*HTTP*).

Para a persistência dos dados foi utilizado o Sistema de Gerenciamento de Banco de dados (SGBD) *Mysql*, que é de código aberto e possui comprovado desempenho, confiabilidade e facilidade de uso [Oracle 2019].

O aplicativo *Android* foi desenvolvido em *Kotlin*, utilizando o *Android Studio*, ferramenta oficial de desenvolvimento do *Google*. A Tabela 1 resume as tecnologias utilizadas.

Tabela 1. Tecnologias utilizadas.

Nome	Tipo	Descrição	Versão
PHP Storm	Ferramenta de Desenvolvimento	Desenvolvimento de projetos PHP	5.3-7.2
Android Studio	Ferramenta de Desenvolvimento	Desenvolvimento de projetos para plataforma Android	3.2.2
Apache	Servidor Web	Servidor de aplicação	2.4.27
MySQL	Banco de dados SQL	Banco de dados relacional	5.6.37
Slim Framework 3	Framework	Framework de desenvolvimento para PHP	3.0
PHP 7	Linguagem de Programação	Desenvolvimento de software	7.0
Kotlin	Linguagem de Programação	Desenvolvimento de software	1.3.2

Para a implantação da aplicação *REST* foi usando o servidor *Apache Tomcat* exposto pela ferramenta *ngrok*, usada para exposição via *Internet* de servidores locais e *firewall*, que gera um endereço *https* público.

A Figura 5 apresenta o *workflow* do processo de aprovação de documentos no sistema. Ao enviar o documento com sucesso, é definido o status do item para “Em análise” e o administrador é notificado que ocorreu uma nova inserção.

Ao fazer a análise do item adicionado, o usuário administrador pode aceitar o documento e caso o faça, o status do item é mudado para “validado” e ele não poderá mais ser editado ou apagado.

Se existir uma pendência, erro ou informações divergentes, o administrador pode solicitar para o usuário realize alterações.

Enquanto o documento estiver com o status “Em análise” o usuário que adicionou o documento poderá alterar ou remover o item inserido e caso ocorra alguma edição no documento, o administrador é notificado e o processo volta para a etapa “Analisar documento”.

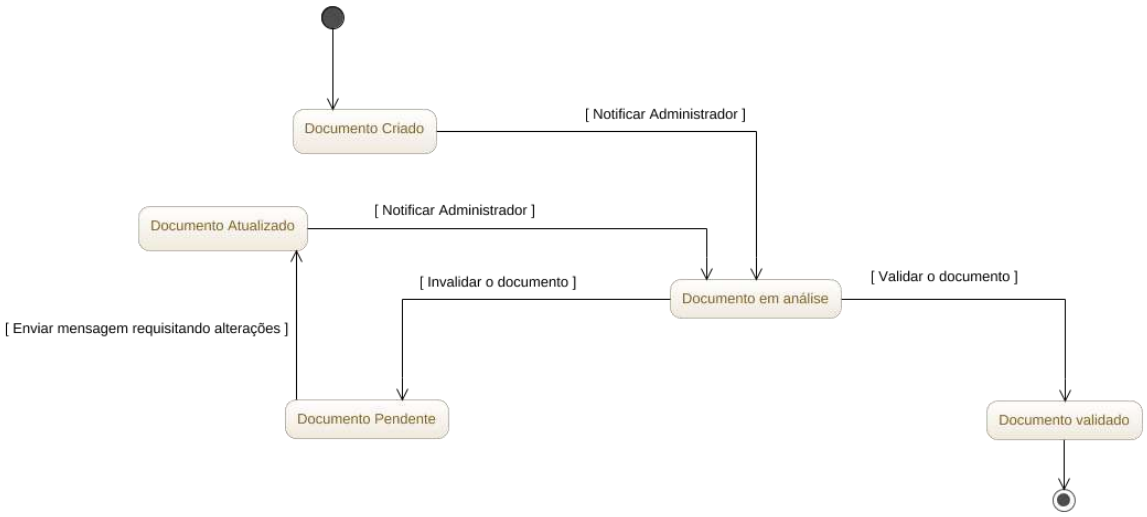

Figura 5. Diagrama de estado para documentos adicionados. Fonte: Criado pelo autor

4.1 Arquitetura plataforma

Na arquitetura definida, a aplicação do servidor e do cliente evoluem de maneira independente. O cliente tem acesso apenas aos recursos disponibilizados pela aplicação servidor, por meio da identificação pelo *Uniform Resource Identifier* (URI), que são as interfaces utilizadas pelos clientes para acessar os recursos desejados no *webservice*. Ao realizar uma requisição para um recurso disponível, o cliente indica o tipo de operação requisitada que está realizando, exemplificadas na Tabela 2.

Tabela 2. Métodos *HTTP*

Requisição *HTTP*	Descrição
GET	Requisição de alguns dados
POST	Requisição enviar alguns dados
DELETE	Requisição para remover algum dado

O *webservice* então faz uso dos componentes arquiteturais para resolver a solicitação, os componentes são:

1. **Banco de dados**: responsável pela persistência dos dados;

2. **Servidor**: tem como responsabilidade receber as requisições dos clientes, locais ou da Internet, e enviar as respostas, gerenciando as conexões e provendo os componentes da aplicação;
3. **API**: Desenvolvida em PHP com *Slim Framework*, segue o modelo cliente-servidor, separando as responsabilidades em dois ambientes, o do cliente trata de questões como interfaces, experiência do usuário e o do servidor trata da comunicação com o bando de dados, gerenciamento de arquivos e outros;
4. **Clientes**: Consome os recursos disponíveis no *webservice*.

4.1 Esquema do Banco de Dados

O mapeamento das entidades; usuário, documento, notificações de documentos, notificações de usuário e edital e seus respectivos atributos e associações, foi realizado após o levantamento de requisitos e entendimento do negócio e serviu para a criação do banco de dados, cujo esquema é apresentado na Figura 7..

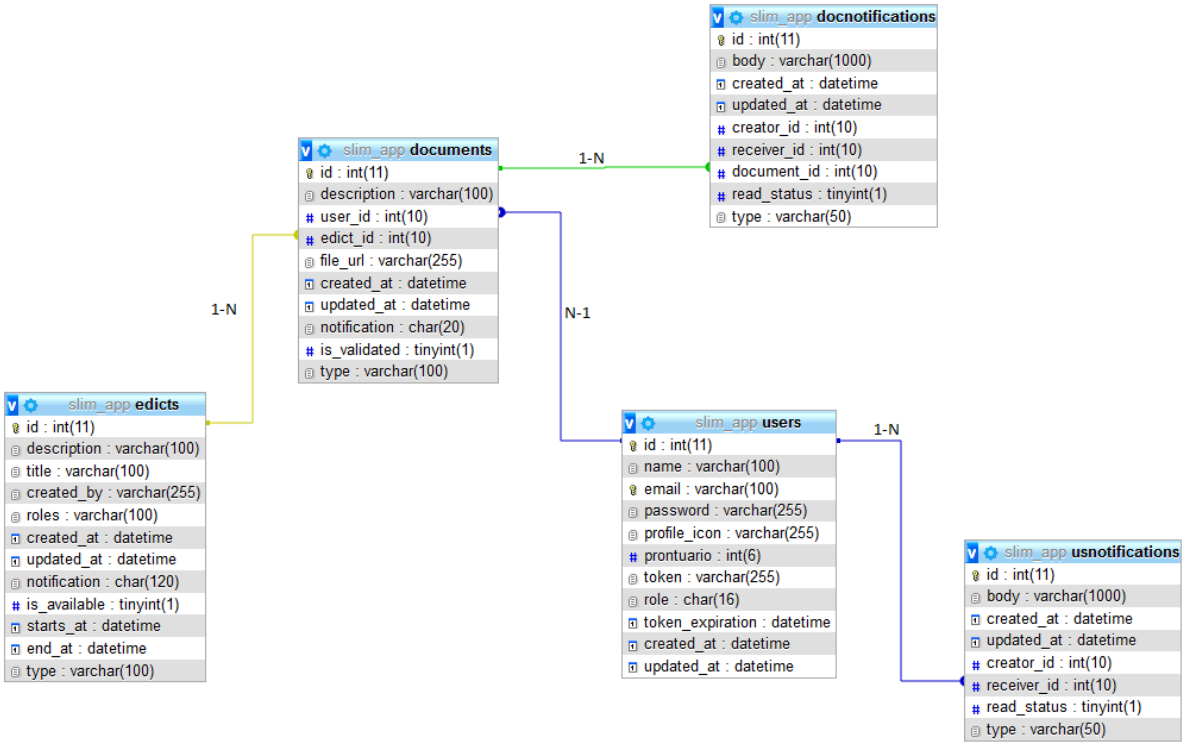

Figura 7. Esquema do banco de dados, *webservice*. Fonte: Criado pelo autor

4.2 Arquiteturas cliente *Android* e Web

O cliente *Android* foi desenvolvido em *Kotlin*, baseado na documentação do site *Android Developers* [*Google*, 2019].

Foram utilizados os componentes arquiteturais recomendados pelo *Google*, como:

1. *Room*, para persistência dos dados no dispositivo;
2. *Navigation,* para a navegação na aplicativo;
3. Componentes de Interface de Usuário (UI) como *Fragment*, que são unidades básicas que demandam menos recursos do dispositivo;
4. Componentes base como *Android KTX* para escrever códigos mais concisos e idiomáticos. Faz uso das convenções, APIs, funções e propriedades de extensão da linguagem *Kotlin*. A Figura 8 exemplifica como o KTX deixa o código mais

conciso, utilizando funcionalidades do *Kotlin*, como: funções estendidas que possibilitam adicionar novas funcionalidades em uma classe existente sem a necessidade de ter acesso a classe original ou código fonte; sobrecarga de operadores e desconstrução de declarações, funcionalidade que permite desconstruir um objeto em múltiplas variáveis.

Figura 8. Exemplo do uso do KTX, transição de fragments. Fonte: *Google*

A arquitetura do aplicativo usa o padrão arquitetural MVP, sendo *Model* (representa os dados e objetos) - *View* (camada de apresentação) - *Presenter* (camada mediadora, comporta-se como um observador da *View* e do *Model*). Esse padrão de software arquitetural permite a separação entre as camadas de apresentação e lógica.

A principal vantagem desse padrão, exemplificado na Figura 9, é ter o *Presenter* como um mediador entre a *View* e o *Model*, o que facilita na interceptação das ações do usuário para atualizar ou modificar o modelo ou a representação na *View*. A aplicação possui camadas para dados, domínio, interfaces de usuários, que são separadas por pacotes no módulo *app*, como exemplificado na Figura 10.

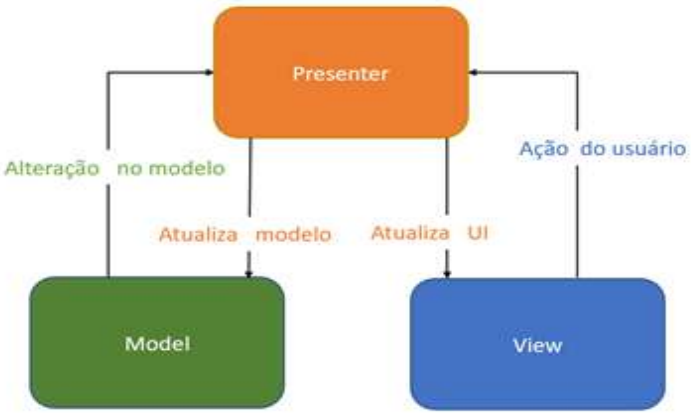

Figura 9. Padrão MVP, aplicativo *Android*. Fonte: Criado pelo autor.

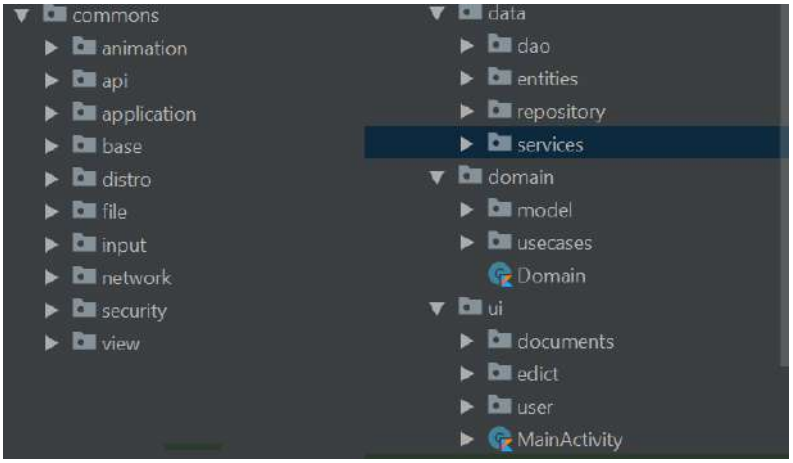

Figura 10. Projeto *Android*, camadas por pacote. Fonte: Criado pelo autor.

Em relação ao cliente web aplicou-se o conceito de *Middleware*, que serve para manipular e validar uma requisição antes dela chegar à camada do controlador via rota, esta camada é associada a outras camadas da aplicação que aplicam o padrão MVC (Modelo, *View*, Controlador) e estão exemplificadas na Figura 11.

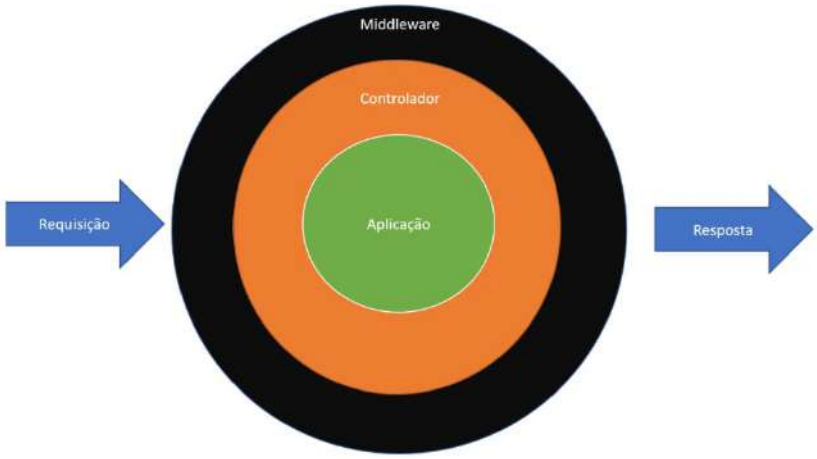

Figura 11. Arquitetura web usando o *Slim* framework. Fonte: Criado pelo autor

4.3 WebService

Esta seção apresenta o desenvolvimento do *webservice*, tendo início com a autenticação, que ocorre pela verificação das credenciais informadas durante a tentativa de conexão e consiste em enviá-las de um cliente remoto para o servidor usando um protocolo de autenticação.

Foi utilizado o *Basic Authentication* do *HTTP*, usando um cabeçalho *HTTP* especial por meio do uso de credenciais no formato ¨usuário:senha¨, estas são codificadas e enviadas por meio da *Secure Socket Layer* (SSL), um protocolo criptografado que garante maior segurança no envio e recebimento dos dados.

Após o processo de autenticação ocorrer com sucesso, é gerado uma chave (*token*), que é informado na resposta. A Figura 12 mostra o retorno de uma autenticação realizada com sucesso. O *token* é enviado no cabeçalho *HTTP* sempre que o aplicativo cliente faz uma requisição, na propriedade *Authorization*.

```
{
    "message": "Authenticated",
    "code": 200,
    "role": "aluno",
    "token": "1654aee2697afbfec7870c15a1d8a6d7d25f9942fe928d65cddb2fe87d5918b075e66b3cd56ed2e8c50cf2
        aa4040f40735dde2335"
}
```


Figura 12. Retorno após autenticação com sucesso. Fonte: Criado pelo autor.

As rotas definidas na *API* foram baseadas nas entidades definidas no esquema do banco de dados. A Tabela 3 mostra alguma das rotas criadas, cujos nomes foram baseados nas entidades criadas na modelagem do banco de dados. É por meio dessas rotas que os recursos do servidor são acessados.

Tabela 3. Rotas do PAE docs

Tipo de Requisição *HTTP*	URI	Utilização
GET	/user/all	Recupera dados de todos os usuários
POST	/documents	Cria um documento
DELETE	/documents/id	Remove documento que possui a id informada

A Figura 13, exemplifica o fluxo da interação entre o servidor e os aplicativos clientes. A troca de dados é realizada no formato *JavaScript Object Notation* (JSON).

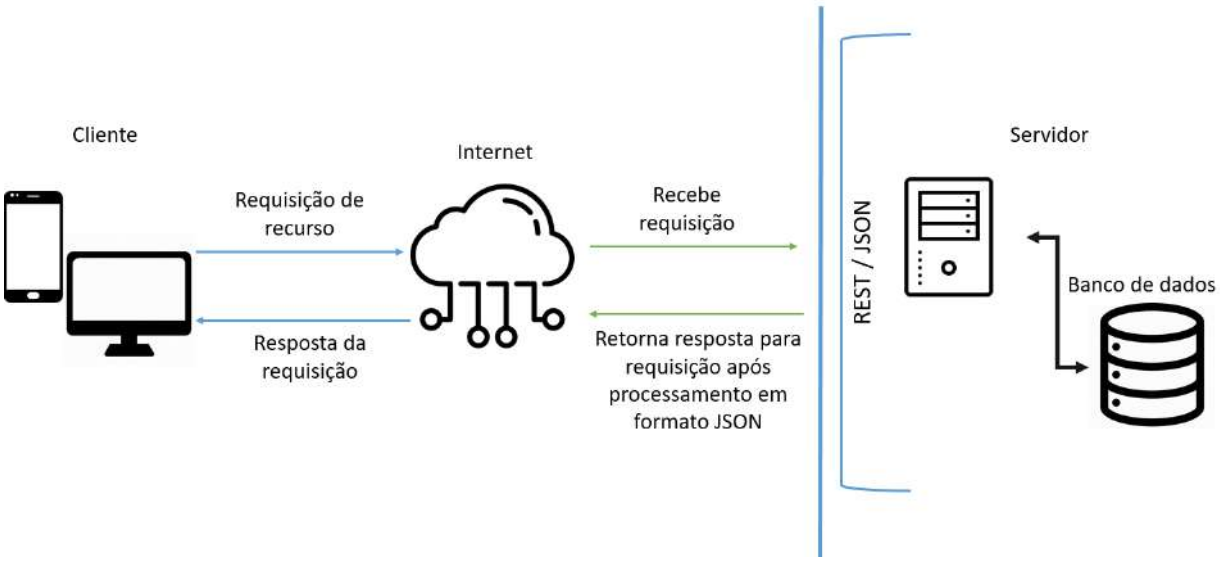

Figura 13. Visão geral da interação cliente e servidor. Fonte: Criado pelo autor

4.4 Desenvolvimento cliente web

A última fase de desenvolvimento do projeto foi a criação das aplicações clientes. Nessa etapa foram desenvolvidos dois clientes, uma para *Web* e outro para o sistema *Android*.

A Tabela 4 descreve as funcionalidades implementadas e a disponibilidade delas para cada perfil de usuário nos clientes mencionados posteriormente, o cliente web possui todas as funcionalidades com exceção da captura de imagem via dispositivo móvel.

Tabela 4. Funcionalidades cliente Web e Android

Funcionalidades	Cliente Web	Cliente *Android*	Perfil
Cadastrar-se	Sim	Sim	Somente aluno
Recuperar senha	Sim	Sim	Aluno e administrador
Adicionar editais	Sim	Não	Somente administrador
Adicionar documentos	Sim	Sim	Aluno e administrador
Capturar imagem	Não	Sim	Somente aluno
Editar documentos	Sim	Sim	Aluno e administrador
Visualizar inscritos	Sim	Não	Somente administrador
Visualizar editais	Sim	Sim	Aluno e administrador
Visualizar documentos	Sim	Sim	Aluno e administrador
Visualizar anexos	Sim	Não	Aluno e administrador
Enviar anexos	Sim	Sim	Aluno e administrador
Enviar mensagem	Sim	Não	Somente administrador
Validar documentos	Sim	Não	Somente administrador
Selecionar edital	Sim	Sim	Aluno e administrador
Remover documento	Sim	Sim	Aluno e administrador

Caso seja necessário criar um usuário com perfil aluno no cliente web, é preciso acessar a página de registro e efetuar o cadastro informando prontuário, e-mail, nome e senha, como mostra a Figura 14.

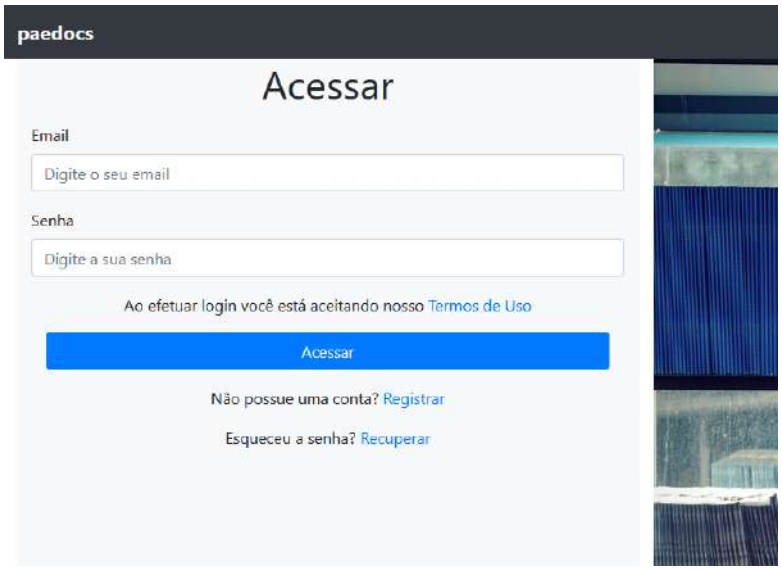

Figura 14. Tela de autenticação, cliente web.

4.4.1 Visão Geral (Dashboard)

A funcionalidade *dashboard* apresenta uma visão geral para o usuário baseado no perfil. Para o perfil aluno a visualização é dos documentos adicionados, como mostra a Figura 15, junto com uma barra de ações e notificações.

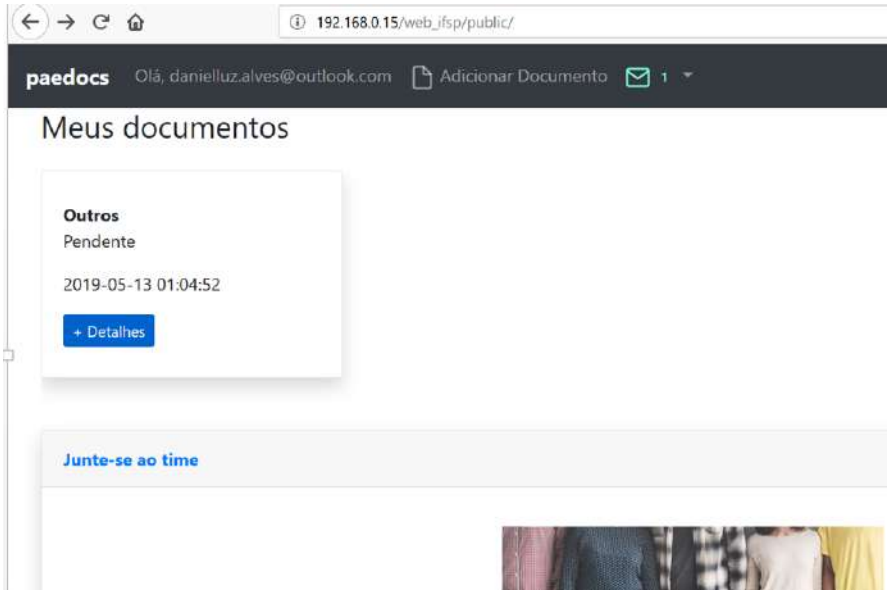

Figura 15. Tela inicial, após autenticação para perfil aluno, cliente Web.

Para o perfil administrador é apresentada uma visualização dos editais criados, como mostra a Figura 16, juntamente com uma barra de ações e notificações.

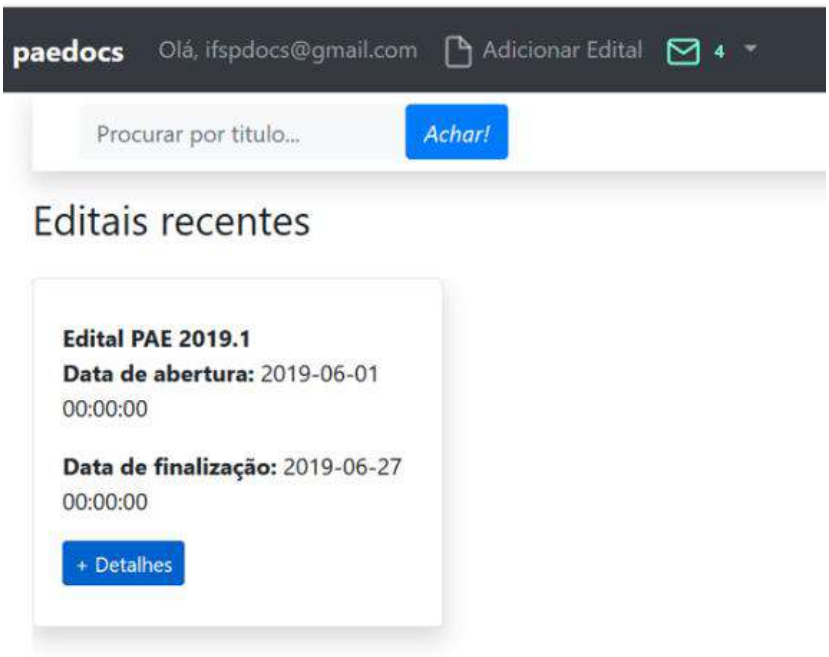

Figura 16. Tela inicial, após autenticação para perfil administrador, cliente web.

4.4.2 Gerenciar documentos

Ao selecionar "+ Detalhes" de um item na *dashboard* do cliente web, o usuário com perfil aluno é redirecionado para uma tela com a visão detalhada do documento escolhido, exemplificado na Figura 17. A funcionalidade de gerenciar documentos abrange alguns dos casos de uso do perfil aluno como: editar, criar e remover documentos.

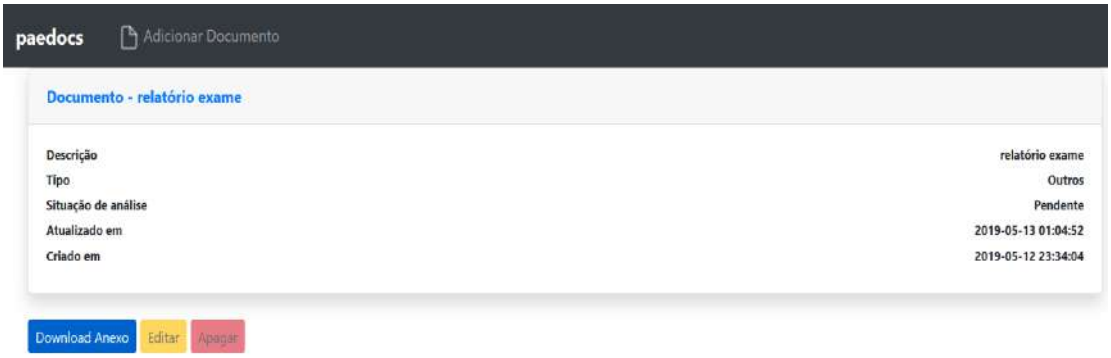

Figura 17. Gerenciamento de documentos, perfil aluno.

4.4.3 Adicionar documentos

Para adicionar novos documentos, o aluno tem essa funcionalidade disponível na barra de navegação. Ao preencher o formulário, mostrado na Figura 18, as informações necessárias são: descrição, tipo de documento e seleção do edital. O usuário deverá anexar um arquivo do tipo imagem que ficará associado ao documento como anexo.

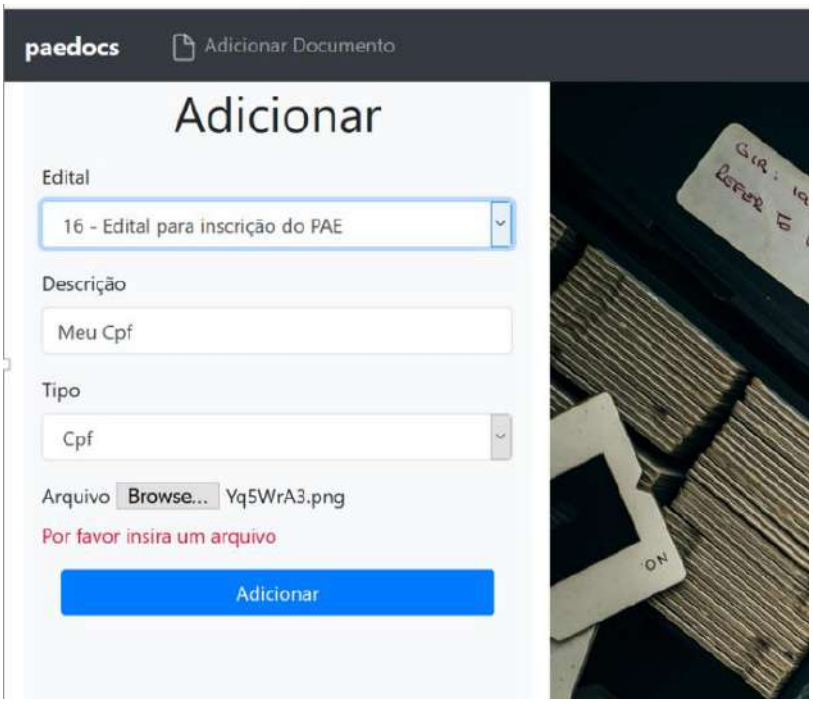

Figura 18. Adicionar documentos.

Essa mesma funcionalidade está disponível para o perfil administrador. Fica disponível na visualização do perfil de um inscrito em um edital, exemplificado na Figura 19.

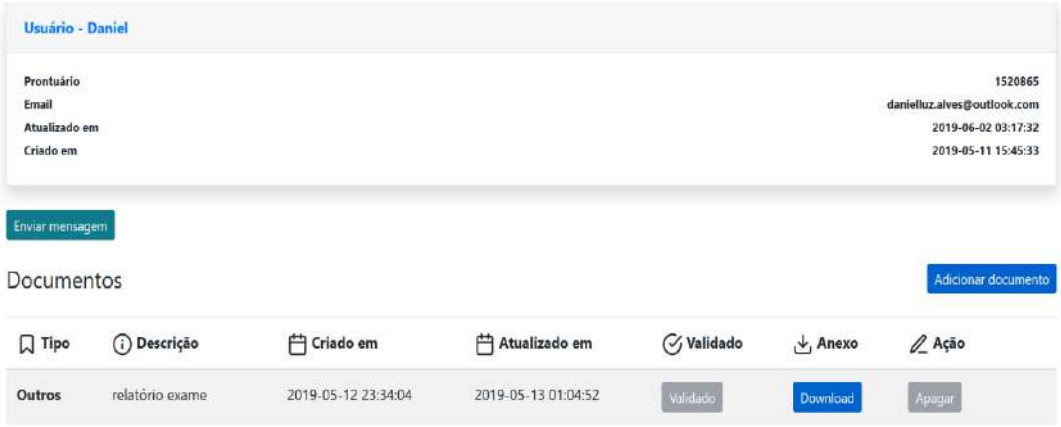

Figura 19. Acesso a funcionalidade adicionar documentos.

4.4.4 Gerenciar inscritos cliente web

Trata-se de uma funcionalidade exclusiva do perfil administrador. Ao selecionar "+ Detalhes" de um edital, na *dashboard*, é possível selecionar a opção "Visualizar inscrições" de um edital, Figura 20.

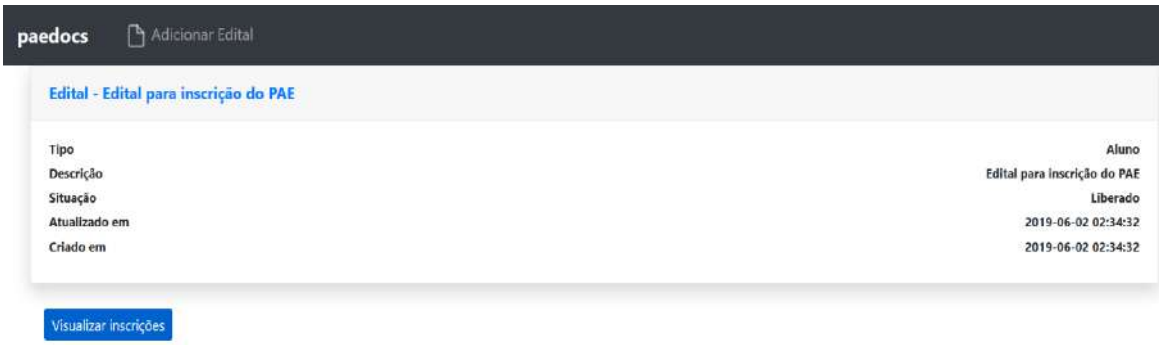

Figura 20. Detalhes do edital selecionado, perfil administrador.

O perfil administrador pode visualizar "+Detalhes" de um inscrito no edital e, caso o faça, a aplicação web vai redirecionar para uma página com detalhes do inscrito e os documentos associados a ele, como ilustra a Figura 21.

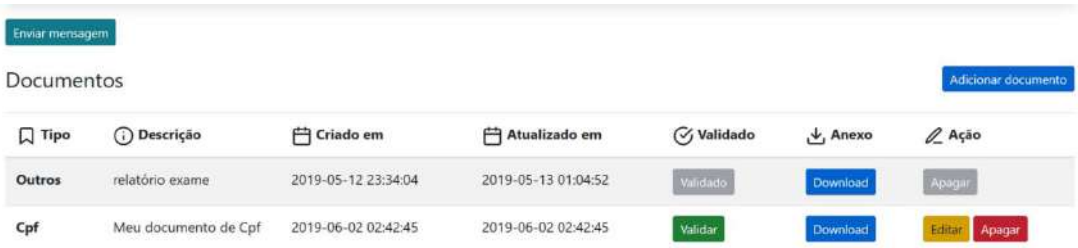

Figura 21. Acesso a funcionalidade gerenciar inscrito.

A funcionalidade de gerenciar inscrito abrange ações como: editar documentos do inscrito selecionado, adicionar documentos para o inscrito selecionado, deletar documentos, enviar mensagem para o inscrito selecionado, validar documentos pendentes, visualizar anexo associado ao documento.

4.5 Desenvolvimento cliente Android

A aplicação para *Android,* mostrado na Figura 22, teve o desenvolvimento de funcionalidades similar a aplicação web, mas é direcionado para o perfil aluno. Possui as funcionalidades de autenticação, selecionar edital, registrar conta, adicionar documentos, editar documentos, deletar documentos e visualizar documentos inseridos.

O diferencial desse cliente é a capacidade de adicionar como anexo as fotos tiradas pela câmera do dispositivo, removendo assim a necessidade do aluno gerar cópias físicas dos documentos.

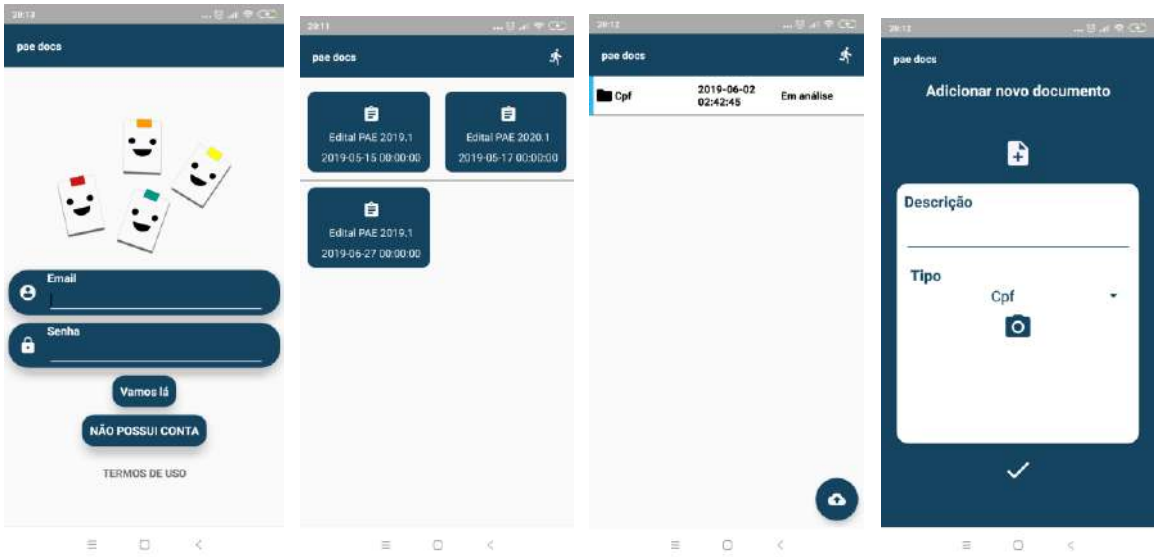

Figura 22. Aplicativo *Android* para o perfil aluno.

5. Conclusão

Este projeto teve como proposta o desenvolvimento da plataforma *PAE docs*, orientada a serviços com acesso via web e aplicativo *Android*. Ambos clientes, web e *Android*, permitem anexar documentos relacionados ao processo de inscrição do Programa de Assistência Estudantil do IFSP *campus* Hortolândia. O cliente web atende também o perfil de administrador, para realização das atividades de gerenciamento. A solução apresentada funciona como um repositório de arquivos que podem ser manipulados e validados pela instituição e pelo aluno.

Com a informatização da etapa de validação de documentos do PAE por meio da plataforma apresentada é possível reduzir a quantidade de filas e espera por parte dos alunos para entregar a documentação assim como o tempo gasto pela equipe sócio pedagógica para validar as cópias dos documentos.

A plataforma fica disponível 24 horas, acessível pela Internet e garante uma redução do uso de papéis, pastas, materiais diversos (grampos, envelopes e marcadores) e do uso do espaço físico da sala da CSP para armazenar cópias dos documentos, que ficam armazenados digitalmente.

Os códigos de desenvolvimento estão disponíveis no *GitHub,* para futuras melhorias e implementações de novas funcionalidades, descritos na tabela 5.

Tabela 5. Repositórios de Códigos

Item da Plataforma	Link
Web Service	https://github.com/lluzalves/slim_rest_api
Aplicação Web	https://github.com/lluzalves/web_ifsp
Aplicativo *Android*	https://github.com/lluzalves/docApp

A utilização de algumas ferramentas como *Git*, *Kanban* e o uso do desenvolvimento incremental com frameworks auxiliaram na organização e execução do projeto, facilitando o desenvolvimento e dando maior visibilidade do que estava completo e do que estava faltando implementar.

O cliente *Android* facilita a captura dos documentos para os usuários do perfil aluno, pois os dispositivos possuem câmeras de captura.

A integração entre as aplicações cliente e servidor foi um desafio e proporcionou novos conhecimentos e técnicas de desenvolvimento juntamente com o uso do PHP associado ao *Slim Framework*.

A expectativa é que o projeto possa evoluir em funcionalidades e atender as necessidades atuais e novas que venham surgir na Instituição em relação a captura e manipulação de dados para processos internos.

Referências

Brasil, Decreto Nº 7.234, De 19 De Julho De 2010. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2010/decreto/d7234.htm/, Acesso em: 12 de fevereiro 2019.

Brasil, Decreto Nº 19.851, De 11 De Abril De 1931. Disponível em: https://www2.camara.leg.br/legin/fed/decret/1930-1939/decreto-19851-11-abril-1931-505837-publicacaooriginal-1-pe.html , Acesso em: 12 de fevereiro 2019.

GAMMA, E. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1a ed.

SOMMERVILLE, I. (2011). *Engenharia de Software*. Pearson, 9a ed.

FIELDING, R., Architectural Styles and the Design of Network-based Software Architectures, 2000.

GREEN, William, Introduction to Electronic Document Management Systems, 1993, System Design Considerations, Acesso em: 12 de fevereiro 2019.

Brasil, Decreto Nº 8.659, De 5 De Abril De 1911. Disponível em: https://www2.camara.leg.br/legin/fed/decret/1910-1919/decreto-8659-5-abril-1911-517247-publicacaooriginal-1-pe.html, Acesso em: 12 de fevereiro 2019.

Brasil, Constituição da República Federativa do Brasil de 1988. Disponível em: http://www.planalto.gov.br/ccivil_03/constituicao/constituicao.htm, Acesso em: 12 de fevereiro de 2019.

Brasil, Lei Nº 11.892, De 29 De Dezembro de 2008, institui a Rede Federal de Ensino. Disponível em : http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2008/lei/l11892.htm, Acesso em: 12 de fevereiro de 2019.

Slim Framework, Disponível em: http://www.slimframework.com/docs/, Acesso em: 12 de fevereiro 2019.

GitHub, Disponível em: http://www.github.com, Acesso em: 12 de fevereiro 2019.

OHNO, T. Toyota Production System – beyond large-scale production. Productivity Press, p.29, 1988, Acesso em: 13 de fevereiro 2019.

Ngrok, Disponível em: http://ngrok.com, Acesso em: 12 de fevereiro 2019.

VASCONCELOS, N. Programa Nacional de Assistência Estudantil: uma análise da assistência estudantil ao longo da história da educação superior no Brasil. Revista da Católica, Uberlândia, v. 2, n. 3, p. 399-411, 2010. Disponível em: <http://www.catolicaonline.com.br/revistadacatolica/artigosv2n3/29-Pos-Graduacao.pdf> Acesso em: 18 de fevereiro 2019

Fernandes, N. G. de O. A política de assistência estudantil e o Programa Nacional de Assistência Estudantil: o caso da Universidade Feral de Itájuba, 2012

GOMES, A. D. Web Services SOAP em Java 2ª Edição, Novatec, 2010.

BASS, L.; CLEMENTS, Paul; KAZMAN, Rick. Software Architecture in Practice. Third Edition, Pearson Education 2013.

ENGHOLM, H. Análise e Design Orientados a Objetos, Primeira Edição, Novatec, 2013

Brasil, Plano Nacional de Assistência Estudantil http://www.andifes.org.br/wp-content/files_flutter/Biblioteca_071_Plano_Nacional_de_Assistencia_Estudantil_da_Andifes_completo.pdf, Acesso em: 02 de fevereiro 2018.

Arquitetura Android, https://developer.android.com/guide, Acesso em: 15 de janeiro 2019

Kanban. Documentação para Desenvolvedores. Disponível em: https://en.wikipedia.org/wiki/Kanban/ , Acesso em: 25 de dezembro 2018.

Document Management System. Disponível em: https://en.wikipedia.org/wiki/Kanban_(development)#Software_development/, Acesso em: 25 de dezembro 2018.

Mysql, Alto Desempenho, Confiável e Fácil de Usar. Disponível em: https://www.oracle.com/br/MySQL/ , Acesso em: 12 de fevereiro 2019.

Stallings, W. (2014). *Criptografia E Segurança De Redes*. Pearson Brasil, 6a ed.

Documento Digitalizado Restrito

TCC do Discente

Assunto: TCC do Discente
Assinado por: Gustavo Guedes
Tipo do Documento: Relatorio
Situação: Finalizado
Nível de Acesso: Restrito
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Gustavo Bartz Guedes**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 04/09/2019 22:02:28.

Este documento foi armazenado no SUAP em 04/09/2019. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 229972
Código de Autenticação: dd2ebd51f4