

**UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO**

**OpenPDI - SOFTWARE PARA APOIO DIDÁTICO
EM PROCESSAMENTO DIGITAL DE IMAGENS**

Trabalho entregue como requisito para composição de nota na disciplina de Engenharia de Software, ministrada pelas Professoras Dra. Aline Maria Malachini Miotto Amaral e Dra. Thelma Elita Colanzi Lopes, junto ao Programa de Pós-Graduação em Ciência da Computação *Stricto Sensu*.

Cleber de Souza Relli - pg54798

Diego Nicácio Santos da Silva - pg403421

Jéfer Benedett Dörr - pg54802

Marcelo Ribeiro Donatão - pg403427

MARINGÁ
PARANÁ – BRASIL

2021

SUMÁRIO

PROPOSTA	3
OBJETIVO GERAL	3
OBJETIVOS ESPECÍFICOS.	3
MÉTODO PROPOSTO	4
EXPECTATIVA	4
EQUIPE	4
DESENVOLVIMENTO	4Arquitetura detalhada do sistema
	6
TECNOLOGIAS UTILIZADAS	8
Linguagem de Programação	8
Framework	9
Sobre o Django	9
Banco de Dados	10
Padrão de Projeto	10
Estimativa do tamanho do código (LOC)	11
Funcionalidades implementadas	12
Fechamento	15
Avaliação do processo de desenvolvimento	16
Dificuldades encontradas	17
Observações importantes:	17
COMO OBTER E RODAR O OPENPDI	17
Trabalhos Futuros	26

PROPOSTA

O processamento digital de imagens (PDI) é uma das áreas de conhecimento da Visão Computacional (VC) que tem tido destaque em diversas vertentes da tecnologia. Observa-se de maneira mais comum sua utilização em setores que ganham maior visibilidade, como medicina, transportes e pesquisas voltadas a análises mais delicadas geralmente a, imagens e vídeos.

Sua utilização tem possibilitado avanços na área de informática aplicada ou mesmo a ciência da computação, isso se dá em relação ao aumento nas pesquisas globais, ferramental computacional cada vez mais evoluído e ainda a utilização do processamento digital de imagens como componente curricular nas instituições de ensino.

A manipulação de imagens proporciona desafios em vários setores, esses algoritmos estão presentes em imagens de satélites, sensores, câmeras, softwares especializados, segurança, monitoramento, entre outros que poderiam ser citados.

OBJETIVO GERAL

Desenvolver uma ferramenta WEB para auxílio no processo de ensino aprendizagem em disciplinas que envolvam o processamento digital de imagens.

OBJETIVOS ESPECÍFICOS

- Propor um melhor entendimento sobre os conceitos de processamento digital de imagens;
- Permitir a experimentação dos alunos de graduação com técnicas de processamento digital de imagens;
- Avaliar o protótipo com base nas boas práticas de desenvolvimento de software;
- Discutir as experiências dos alunos bem como a performance da ferramenta computacional desenvolvida.

MÉTODO PROPOSTO

A metodologia proposta visa promover um melhoramento no aproveitamento do processo de ensino aprendizagem, relacionado aos conteúdos didáticos e os códigos (fonte) que os alunos e professores irão manipular. O professor será o agente centralizador, com a possibilidade de ilustrar o código e o respectivo resultado em

imagens. E, também para o aluno que terá acesso ao código (fonte) e aos exemplos para compreender o que estará acontecendo e conciliar com o conteúdo trabalhado em aula.

Serão utilizadas as bibliotecas do *python* como *numpy*, *opencv*, *pillow*, *scikit-image*, *tkinter* e *matplotlib*.

Este material dará base de conhecimento para futuras disciplinas de visão computacional e reconhecimento de padrões, processamento digital de imagens e aprendizagem de máquina.

EXPECTATIVA

Espera-se com esse protótipo de ferramenta computacional poder propiciar aos acadêmicos uma experiência válida, cheia de descobertas e se possível despertar o interesse pelo processamento digital de imagens.

Ainda, realizar a entrega de um software que auxilia de maneira didática os acadêmicos de graduação dos cursos de computação da Universidade Estadual de Maringá, Universidade Paranaense - UNIPAR (Unidades com cursos e EAD) e Universidade Federal do Paraná - UFPR (Palotina).

EQUIPE

- Cleber de Souza Relli - pg54798 – Doutorando
- Diego Nicácio Santos da Silva - pg403421 - Mestrando
- Jéfer Benedett Dörr - pg54802 - Doutorando
- Marcelo Ribeiro Donatão - pg403427 - Mestrando

DESENVOLVIMENTO

O sistema OpenPDI ficará disponível sob a licença *General Public License v3*¹ (GPL) e poderá ser obtido diretamente do seu repositório no Github². O código fonte está aberto e disponível para poder ser alterado ou ter novas funções adicionadas. Embora seja importante dizer que as alterações devem ser registradas de modo que sempre se saiba quem realizou as alterações.

¹ <https://www.gnu.org/licenses/gpl-3.0.pt-br.html>

² <https://github.com/nicacio16/ProjetoES>

O sistema é escrito em linguagem Python e utiliza o Framework de desenvolvimento ágil, o Django³ que permite criar aplicações Web. Desta forma, o OpenPDI é uma solução multiplataforma e poderá ser executado em diferentes ambientes, sendo necessário apenas um navegador para executá-lo.

Apesar de executar em um navegador web, o OpenPDI trabalha no sistema cliente-servidor localmente e não é necessário nenhum cadastro para ser utilizado. Como projeto futuro poderá ser proposto o uso online centralizado hospedado em algum servidor.

O mecanismo de funcionamento do OpenPDI já permitiria trabalhar desta forma, no entanto, seria necessário ter um servidor disponível para instalar a aplicação, implementar controle de usuário, algumas métricas de segurança e garantia de disponibilidade. Mas devido ao contexto do desenvolvimento caber na disciplina de Engenharia de Software, considerando o tempo hábil para serem desenvolvidos e testados estes requisitos, optou-se por deixar essa funcionalidade como projetos futuros para viabilizar a entrega de uma versão funcional, conforme o prometido.

O uso previsto do OpenPDI é como uma ferramenta de apoio ao ensino dos conceitos de Processamento Digital de Imagens (PDI), logo os dois possíveis usos seriam pelo professor ou pelo aluno, sendo que estes usuários no momento não se diferenciam no sistema. Visto que cada um é livre para baixar, executar e alterar a versão do OpenPDI que estiver rodando na sua máquina.

No uso do professor o OpenPDI será uma ferramenta onde ele poderá exemplificar e mostrar diretamente as aplicações do conceito ou filtros que estiver explicando em aula, onde ele poderá exibir a imagem, mostrar o código e ter um conteúdo de apoio sobre a teoria do que está sendo explicado.

Pretende-se que esse modelo auxilie o conteúdo a não ficar vago ou preso somente a conceitos matemáticos que talvez o aluno não consiga conectar ao que está acontecendo.

Uma vez que cada aluno terá a sua versão do OpenPDI, o professor poderá solicitar como atividade complementar de aula, que o aluno faça modificações nos filtros que existem, que os utilize como ferramenta para aplicar filtros e ter a imagem de saída para algum outro fim ou para incluir novos filtros.

³ <https://www.djangoproject.com/>

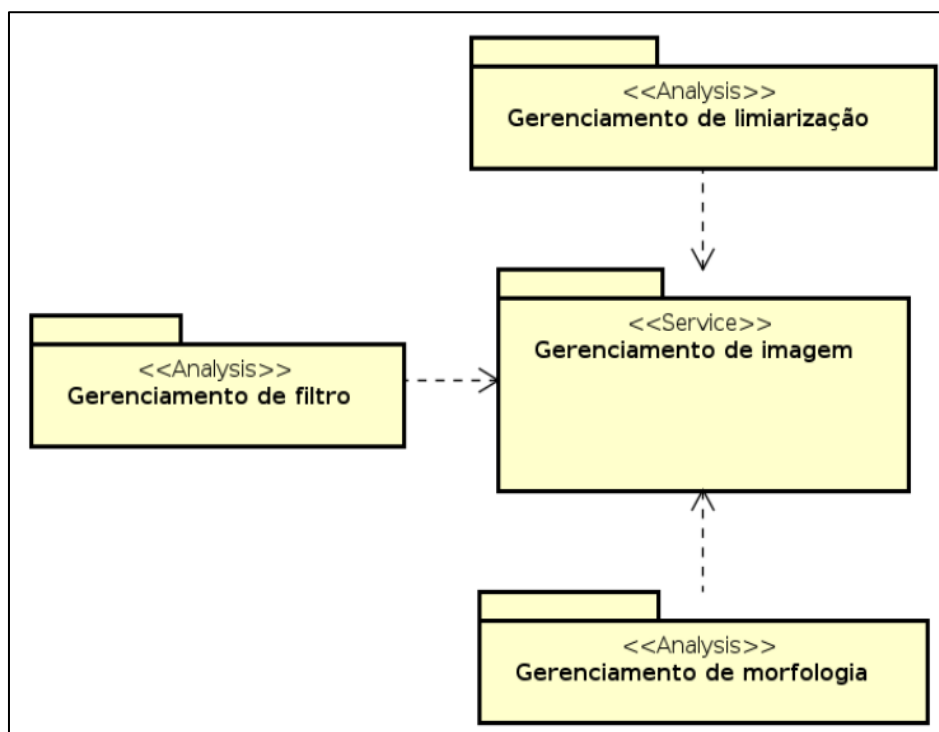
No uso do aluno, poderá ser uma ferramenta de apoio para acompanhar as aulas, para observar na prática as coisas que o professor estiver explicando, para direcionar os estudos, para complementar o ensino tirando possíveis dúvidas, ou para desenvolver as atividades solicitadas pelo professor.

ARQUITETURA DETALHADA DO SISTEMA

A presente seção apresenta uma explicação onde é possível observar com riqueza de detalhes como a arquitetura do OpenPDI está constituída.

Inicialmente a arquitetura relaciona os módulos de gerenciamento de limiarização, gerenciamento de filtros e gerenciamento de morfologia com o gerenciamento de imagens, visto que as imagens são os ativos principais da ferramenta computacional. Isso é possível de se observar por meio das figuras, um (arquitetura inicial) e dois (visão geral das classes).

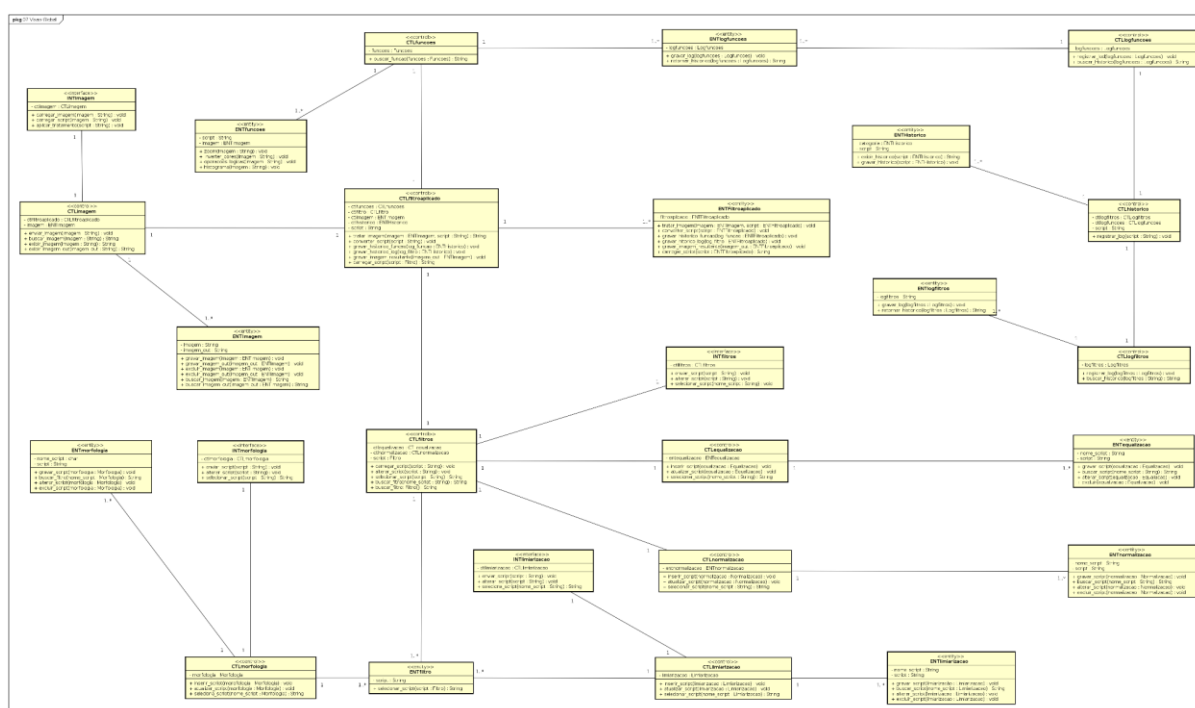
Figura 1 - Arquitetura Inicial



Fonte: Os Autores

Na Figura 2 é possível observar o total de classes que compreendem o sistema OpenPDI.

Figura 2 - Visão Geral das Classes



Fonte: Os autores.

TECNOLOGIAS UTILIZADAS

A seguir são apresentadas as tecnologias utilizadas para desenvolver o OpenPDI.

- **Linguagem de Programação:** Como base, foi utilizada a linguagem de programação Python⁴, juntamente com algumas bibliotecas de apoio, como:
 - NumPy⁵ é uma das bibliotecas de computação numérica utilizada, entre outros fins, em processamento de imagens.
 - OpenCV⁶ é uma biblioteca para trabalhar com visão computacional.
 - Matplotlib⁷ é uma biblioteca *Python* para criação de gráficos e visualizações de dados.

A linguagem Python foi escolhida exatamente por prover boas ferramentas para trabalhar com imagens.

⁴ <https://www.python.org/>

⁵ <https://numpy.org/>

⁶ <https://opencv.org/>

⁷ <https://matplotlib.org/>

FRAMEWORK

Para auxiliar no desenvolvimento ágil, foi utilizado *framework* de desenvolvimento Web Django⁸. Django é um framework web full stack open source (código aberto) baseado em Python, gratuito e de alto nível. Os métodos de obtenção de atributos (getters) e de modificação de atributos (setters) não estão representados, pois, serão gerados automaticamente pelo referido framework em tempo de desenvolvimento.

O objetivo da escolha foi por ser baseado em Python e por auxiliar o desenvolvimento Web de forma rápida e limpa, *clean code*. Conjunto de boas práticas para obter uma maior legibilidade e manutenibilidade do código.

SOBRE O DJANGO

O Django trabalha com sistema de “template”, separando a lógica da apresentação. Um dos objetivos é por questão de segurança para coibir a inclusão de código malicioso, como comandos que deletam registros de banco de dados. O sistema de Views tem o objetivo de trazer simplicidade, uma “view” é uma função Python e não permite instanciar uma classe nova enquanto uma função pode resolver o que precisa. As “Views” tem acesso a um objeto de requisição, o objeto é passado diretamente para uma função “view”, ao invés da função “view” ter que acessar os dados de uma requisição vindos de uma variável global. O resultado é uma requisição leve, limpa e fácil de testar. O framework faz distinção entre dados de GET e POST.

Para a parte web, o Django gerencia o cache de forma que a aplicação seja a mais rápida possível gerenciando as operações `get()`. Fornece consistência, extensibilidade e segurança. Já provê proteção contra os mais comuns ataques da web, como por exemplo o Cross Site Scripting (XSS), injeção SQL, Cross Site Request Forgery (CSRF) e o Clickjacking. É sustentável se baseando no princípio Don't Repeat Yourself (DRY) para fácil manutenção e reutilização pelo agrupamento de funcionalidades. É portátil, não é necessário estar preso a nenhum sistema operacional ou plataforma, basta ter seu interpretador instalado na máquina.

⁸ <https://www.djangoproject.com/>

BANCO DE DADOS

Foi utilizado o `sqlite3`⁹, o SQLite está incluído no Python, logo, não é necessário instalar nada para dar suporte a este banco de dados. Também é o padrão utilizado no Framework Django e atende aos requisitos do sistema OpenPDI.

PADRÃO DE PROJETO

O Framework Django trabalha com o padrão de projeto Model, Template, View (MTV)¹⁰. Um projeto, Django, com padrão de projeto MTV, significa:

- Model: Mapeamento do banco de dados para o projeto;
- Template: Páginas para visualização de dados. Normalmente, é aqui que fica o HTML que será renderizado nos navegadores;
- View: Lógica de negócio. É aqui que determinamos o que irá acontecer em nosso projeto.

Toda esta arquitetura é interligada e conversam entre si. Uma depende da outra para realizar um determinado serviço e, no final, executar a tarefa que o usuário solicitou.

A camada de visualização de dados são chamadas de templates e a camada de lógica é chamada de view.

No padrão MTV, uma view é uma função que retorna os dados para uma solicitação, ela define apenas quais dados serão apresentados e não como serão mostrados.

Por questões de organização e padronização do código, são separados os conteúdos das apresentações do conteúdo. Com os dados retornados pela view, a camada template fica responsável por definir a forma que esses dados serão apresentados, normalmente em páginas HTML.

No Django, a camada controller é a própria estrutura do projeto. É o mecanismo que envia uma solicitação para a view adequada de acordo com a configuração de rotas do Django. O controlador da aplicação é o próprio conjunto de bibliotecas que compõem o projeto.

⁹ <https://www.sqlite.org/index.html>

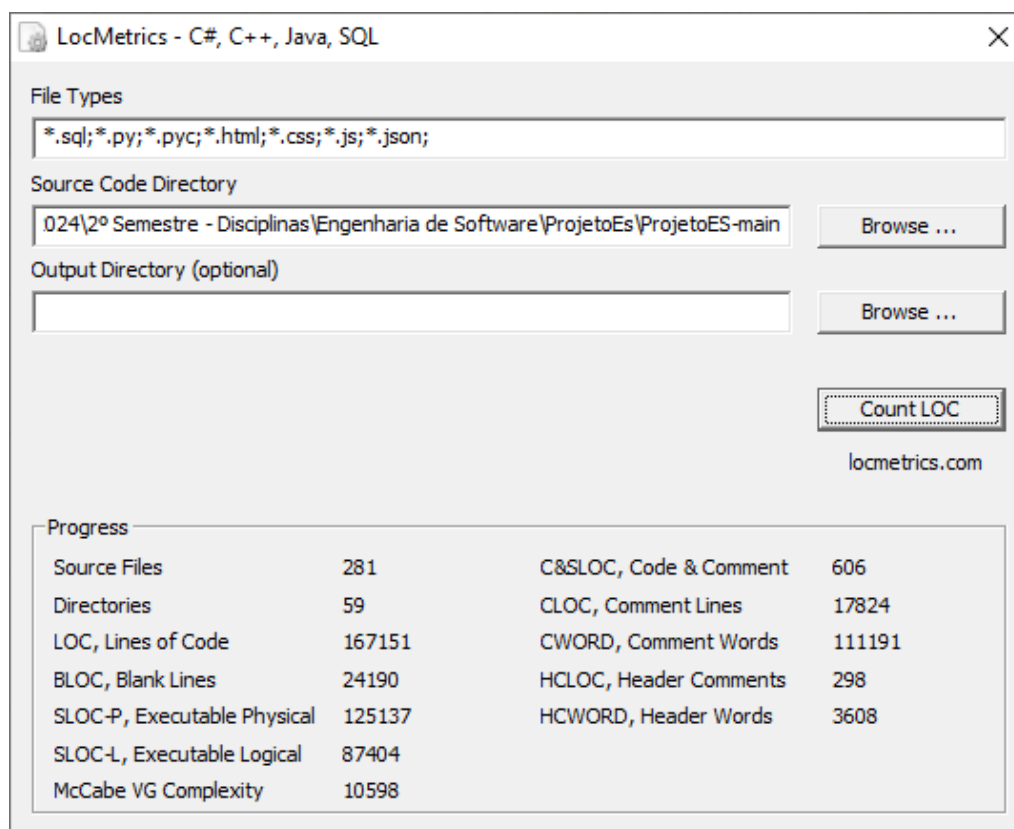
¹⁰ <https://docs.djangoproject.com/en/3.0/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names>

A view retorna os dados a serem visualizados e, serão exibidos ao usuário por meio de templates.

ESTIMATIVA DO TAMANHO DO CÓDIGO (LOC)

A maneira mais simples de medir o tamanho de um programa é contar as linhas. Uma métrica utilizada para avaliar o tamanho de programas é a contagem de linhas, chamada de *Lines of Code* (LoC). Para realizar a estimativa do tamanho do código foi utilizado o programa LocMetrics e a Figura X - Métrica LOC obtida com o programa LocMetrics.

Figura 3 - Métrica LOC



Fonte: Os autores.

Lembrando que a produtividade não pode ser medida por LoC.

FUNCIONALIDADES IMPLEMENTADAS

A interface do usuário foi implementada conforme a evolução dos protótipos apresentados, inicialmente foi prototipada uma proposta de Baixa Fidelidade que apresentava apenas os recursos mínimos necessários. Em seguida um protótipo de Média Fidelidade com funcionalidade de navegação, mas ainda sem função.

Validados este protótipo, foram projetadas as telas que eram inicialmente previstas para serem desktop e devido a decisões de projeto migraram para web, então foram projetadas e aprovadas as telas do protótipo de alta fidelidade que são as atuais do OpenPDI.

OS REQUISITOS FUNCIONAIS:

- RF-A.001 - Abrir -> Abrir um arquivo de imagem existente.
- RF-A.002 - Recentes -> Visualizar os arquivos recentemente utilizados.
- RF-A.003 - Salvar -> Salvar a imagem resultante, bem com suas alterações.
- RF-A.004 - Salvar Como -> Salvar o arquivo permitindo alterar o nome do mesmo bem como fora da pasta padrão.
- RF-A.005 - Sair -> Fechar o programa.
- RF-E.001 - Desfazer -> Permitir desfazer a última alteração.
- RF-E.002 - Copiar -> Permitir copiar o conteúdo selecionado.
- RF-E.003 - Recortar -> Permitir recortar o conteúdo selecionado.
- RF-E.004 - Colar -> Permitir colar o conteúdo da área de transferência.

“Abrir” uma imagem (RF-A.001), foi substituído pela opção “browse” que faz o upload da imagem. Para salvar esta imagem carregada na pasta do projeto, basta clicar em “anexar”. A opção de ‘salvar’ é automática (RF-A.003). A cada operação aplicada, é salva uma nova cópia da imagem automaticamente (RF-A.004).

- A (RF-A.005), basta fechar a aba do navegador.
- AS (RF-E.002, RF-E.003 e RF-E.004) ficaram desnecessárias. Para atender a RF-E.001, basta abrir a imagem anterior que ficou salva automaticamente.

Os requisitos não funcionais atendidos foram:

- Fácil de disponibilizar e executar;
- Transferência de conhecimento;
- Permitir cópia dos "códigos exemplos";
- Disponibilizar código-fonte;
- Fácil usabilidade;
- Executar python com a biblioteca OpenCV;

Os requisitos que não foram possíveis de serem atendidos estão listados abaixo:

- Utilizar atalhos padrão (ctrl+s, ctrl+o)

Pelo mesmo motivo que a funcionalidade do menu Arquivo acabou não sendo implementada, como “outros requisitos”, o:

- Possibilitar adição de novas funcionalidades futuramente, de tal modo a não alterar a estrutura do sistema.

Foi atendido já que é possível adicionar novas funcionalidades no projeto que estiver rodando no computador e o projeto está disponível com código aberto no git.

Nem todos os filtros propostos estão ainda funcionais no sistema, as explicações teóricas também não, mas devido ao curto espaço de tempo para implementação. Porém, foi gerado um documento de apoio com cerca de 60 páginas onde estão presentes os, fontes de todos os filtros, com explicação teórica sobre cada um deles, e o arquivo está disponibilizado juntamente com o git do projeto.

Figura 4 - Material Complementar.



Fonte: Os autores.

Os itens que compreendem o material complementar podem ser observados na Figura 5.

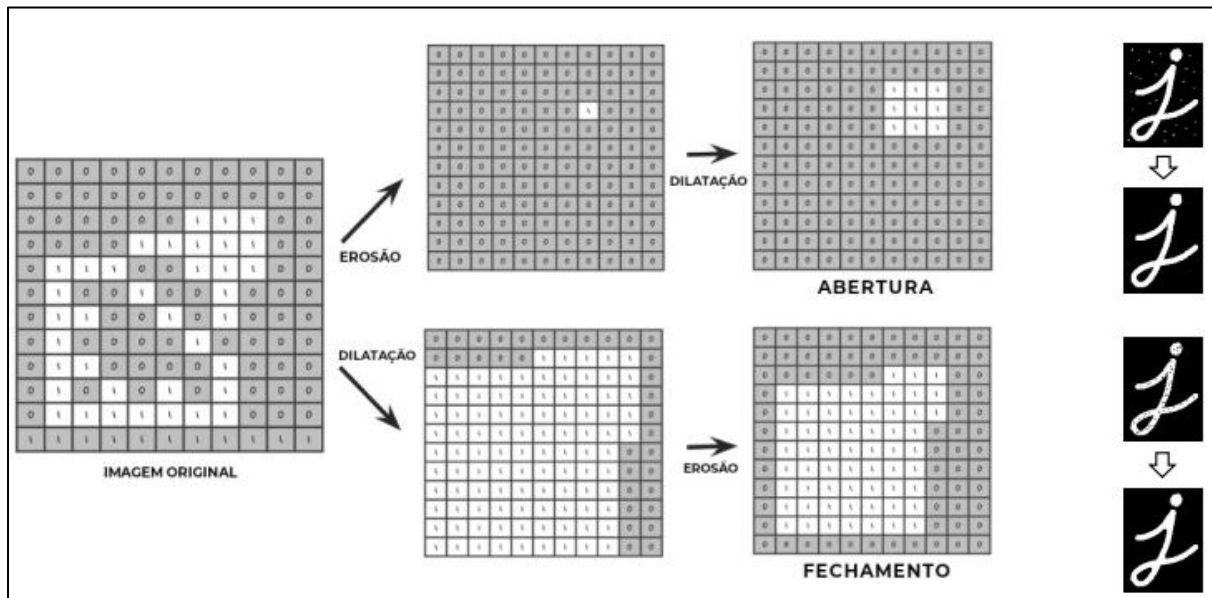
Figura 5 - Material complementar - Conteúdos

Pré processamento de imagens	38
Redimensionamento	38
Redimensionamento no openCV	39
Segmentação	40
Limiarização	40
Limiarização Simples	40
Teste Limiar Simples	41
Método de Otsu	44
Limiarização Adaptativa	45
Primeira atividade de Limiarização Adaptativa	46
imagem de entrada	48
Limiar <u>Ordinário</u>	49
CLAHE Aplicado	49
Limiarização Adaptativa Gaussiana	49
Operações Morfológicas	50
Dilatação e Erosão	50
Erosão	50
Dilatação	52
Abertura e Fechamento	52
Abertura	53
Fechamento	54
Operações morfológicas no processamento de imagens (gradiente)	55
Remoção de Ruído	57
Convolução	57
Remoção de Ruído com Desfoque	59
Desfoque com Média	60
Desfoque Gaussiano	60
Desfoque com Mediana	61
Filtro Bilateral	62

Fonte: Os autores.

Grande parte das operações tem como base a erosão e a dilatação, assim é possível observar um exemplo da operação de fechamento na Figura 6.4

Figura 6 - Exemplo da operação de fechamento.



Fonte: Os autores.

O código em python da operação de fechamento é exposto no Quadro 1.

Quadro 1 - Código Python da operação de fechamento.

```
img = cv2.imread('texto-opencv2.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
cv2_imshow(gray)
```

Fonte: Os autores.

Outro exemplo de processamento de saída da operação de fechamento é ilustrada na Figura 8, a Figura 7 mostra a imagem original.

Figura 7 - Imagem original.



Fonte: Os autores.

Com base na imagem original, aplica-se a operação de fechamento, conforme Quadro 2.

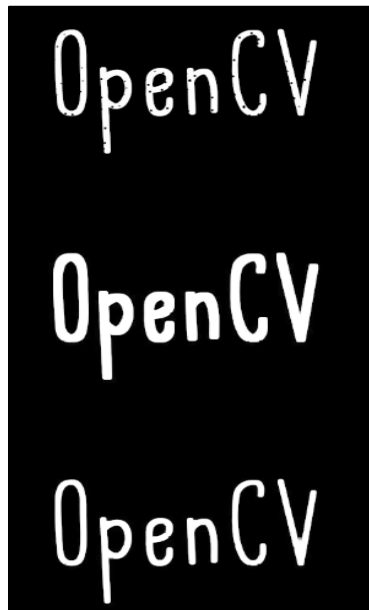
Quadro 2 - Código dilatação, erosão e fechamento.

```
dilatacao = cv2.dilate(gray, np.ones((5,5)))  
fechamento = cv2.erode(dilatacao, np.ones((5,5)))  
cv2_imshow(gray)  
cv2_imshow(dilatacao)  
cv2_imshow(fechamento)
```

Fonte: Os autores.

O resultado desse processamento pode ser observado na Figura 8.

Figura 8 - Resultado do processamento.



Fonte: Os autores.

AVALIAÇÃO DO PROCESSO DE DESENVOLVIMENTO

Desenvolver sistemas não é uma tarefa fácil, as especificações sempre são um desafio a ser encarado pela equipe de desenvolvimento. A análise de requisitos, a modelagem, protótipos, testes e implantação sem dúvidas requerem atenção, comprometimento e conformidade.

Durante o processo de desenvolvimento desse projeto de produto de software utilizou-se inúmeros modelos que foram essenciais para concluir a atividade como um

todo. Os diagramas do workflow de requisitos guiaram o caminho para atingir entender a visão geral das funcionalidades do sistema.

No workflow de análise foi possível entender de que modo os relacionamentos se comunicam entre as entidades.

No workflow de projetos os atributos foram evidenciados, juntamente com os métodos das classes e suas dependências.

AVALIAÇÃO DA CONSISTÊNCIA DOS MODELOS VERSUS IMPLEMENTAÇÃO

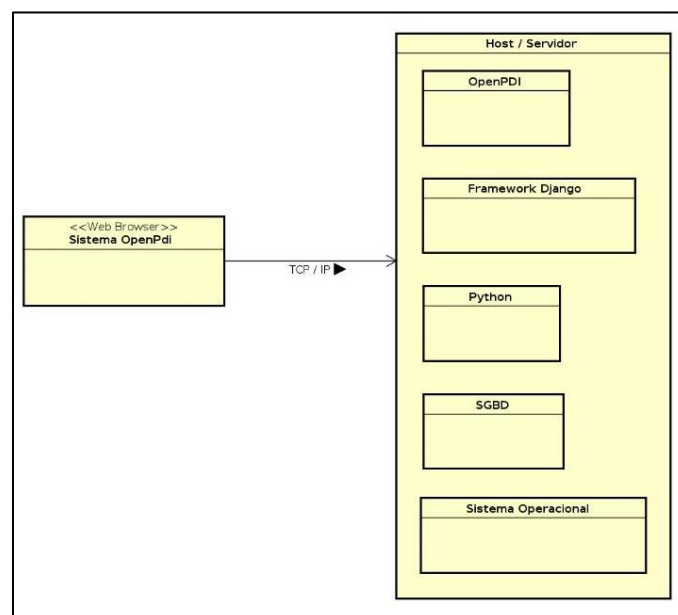
O processo de desenvolvimento de produto de software é algo delicado, as especificações nem sempre ficam claras ou entendíveis por alguma das partes. Assim mesmo não sendo o essencial, é possível que durante o desenvolvimento o escopo sofra alterações, que podem influenciar de maneira temporal, monetária e também em recursos humanos.

Neste projeto a principal mudança foi a alteração de ferramenta desktop para uma plataforma web, possibilitando que o software alcance mais usuários e que não necessite de hardware específico para rodar.

IMPLANTAÇÃO DO SISTEMA

A implantação do sistema, ficou como a prevista no diagrama de implantação, figura 9.

Figura 9 - Diagrama de implantação.



Fonte: Os autores.

O arquivo de especificações dos requisitos se encontra disponível juntamente com o git do projeto.

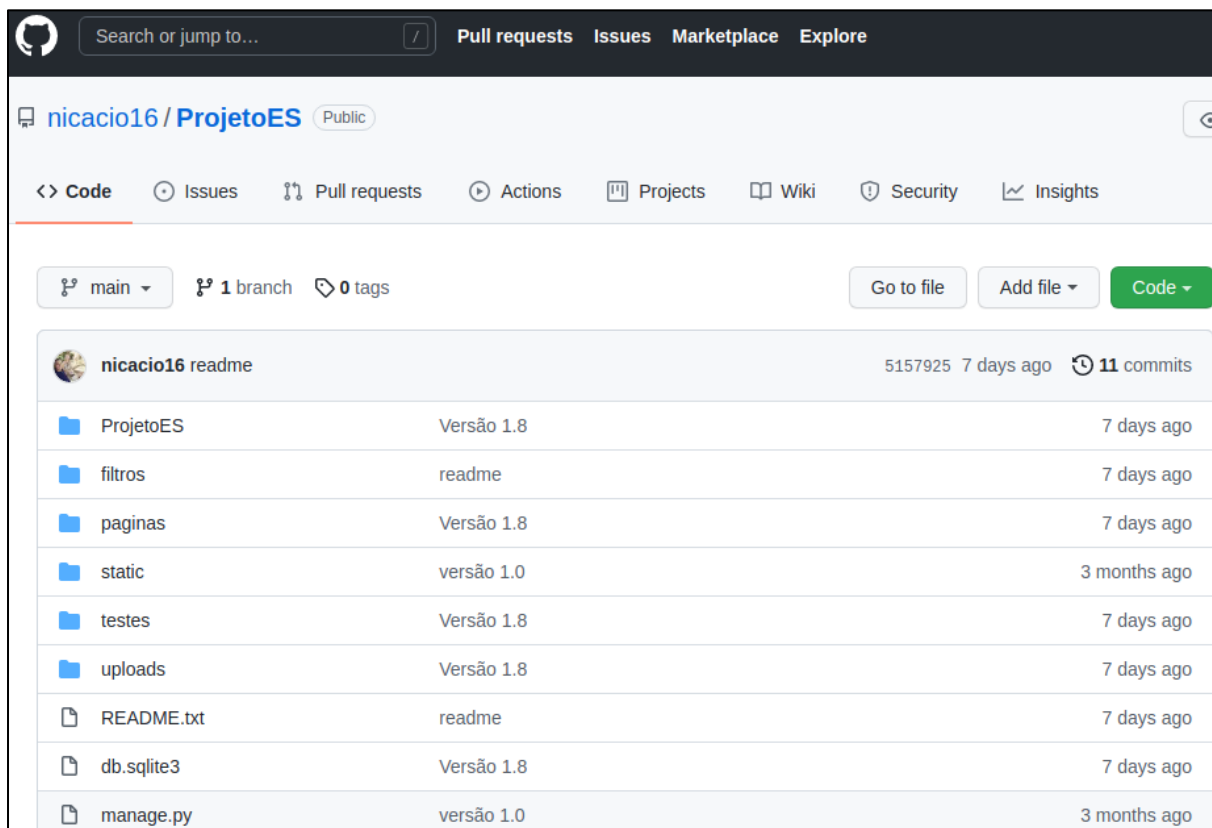
DIFICULDADES ENCONTRADAS

O fator mais importante encontrado e discutido foi a questão temporal para a realização da atividade. Fatores como não estar inserido no mercado de programação também podem ser considerados, além de relacionamento dentro da equipe no sentido de distribuição de tarefas, cumprimento de prazos internos muito em razão de compromissos trabalhistas e disciplinas do mestrado e doutorado.

OBSERVAÇÕES IMPORTANTES: COMO OBTER E EXECUTAR O OPENPDI

Para obter e executar o OpenPDI, inicialmente é necessário acessar o git do projeto, no endereço <https://github.com/nicacio16/ProjetoES>, como mostrado na a Figura 10 - git do projeto OpenPDI.

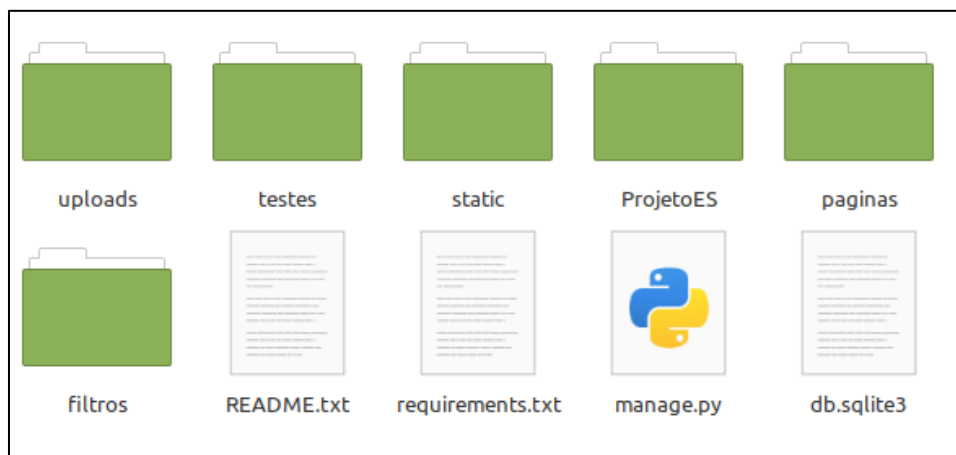
Figura 10 - Local de armazenamento do OpenPDI.



Fonte: Os autores.

Após realizar o download e descompactada a pasta, o conteúdo da pasta descompactada é mostrado na Figura 11 - pasta descompactada do projeto.

Figura 11 - Pasta compactada com os arquivos necessários.



Fonte: Os autores.

O arquivo README.txt, explica como como executar e acessar o OpenPDI, o conteúdo deste arquivo é:

- Rode o comando abaixo no ambiente virtual para instalar os pacotes necessários.
- `pip install -r requirements.txt`
- Para rodar:
 - `python manage.py runserver`
- para acessar:
 - `http://localhost:8000/`

O comando 'pip install -r requirements.txt' irá instalar todas as dependências para rodar o sistema, o comando 'python manage.py runserver' irá executar o sistema e ele estará acessível em qualquer navegador no seguinte endereço: <http://localhost:8000/>. Então:

- O arquivo requirements.txt é responsável por instalar as dependências necessárias para rodar o OpenPDI tanto em sistemas baseados em Microsoft Windows quanto em GNU/Linux.

O conteúdo do arquivo requirements.txt é:

- `asgiref==3.4.1`
- `Django==3.2.7`
- `django-braces==1.14.0`
- `django-cleanup==5.2.0`
- `django-crispy-forms==1.12.0`
- `pytz==2021.1`
- `six==1.16.0`
- `sqlparse==0.4.1`

Executando o requirements, com o comando 'pip install -r requirements.txt', serão instaladas as dependências e a saída será similar a mostrada na Figura 12 - instalação das dependências com o arquivo requirements.txt.

Figura 12 - Instalação das dependências com o arquivo requirements.txt.

```
(base) jefer@HP-EliteBook-Folio-9470m:~/Downloads/ProjetoES-main$ pip install -r requirements.txt
Requirement already satisfied: asgiref==3.4.1 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 1)) (3.4.1)
Requirement already satisfied: Django==3.2.7 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 2)) (3.2.7)
Requirement already satisfied: django-braces==1.14.0 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 3)) (1.14.0)
Requirement already satisfied: django-cleanup==5.2.0 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 4)) (5.2.0)
Requirement already satisfied: django-crispy-forms==1.12.0 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 5)) (1.12.0)
Requirement already satisfied: pytz==2021.1 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 6)) (2021.1)
Requirement already satisfied: six==1.16.0 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 7)) (1.16.0)
Requirement already satisfied: sqlparse==0.4.1 in /home/jefer/miniconda3/lib/python3.9/site-packages (from requirements.txt (line 8)) (0.4.1)
```

Fonte: Os autores.

- Para iniciar o sistema, o comando 'python manage.py runserver' irá executar o sistema e ele estará acessível em qualquer navegador no seguinte endereço: <http://localhost:8000/>. Como mostra a Figura 13 - Executando o servidor do OpenPDI.

Figura 13 - Executando o servidor do OpenPDI.

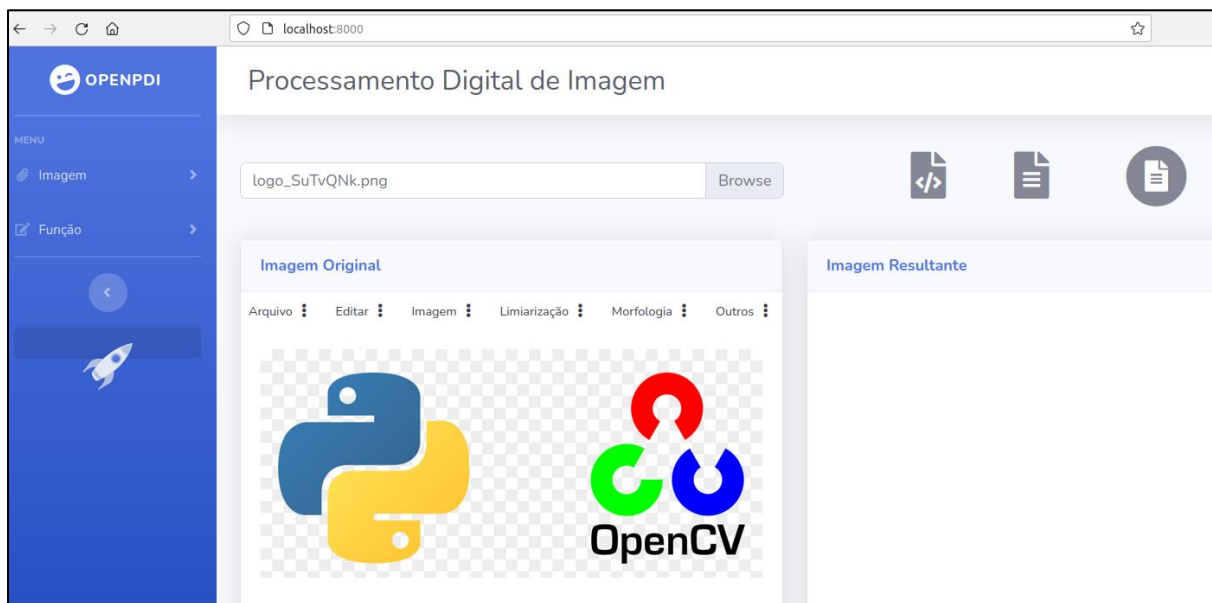
```
(base) jefer@HP-EliteBook-Folio-9470m:~/Downloads/ProjetoES-main$ python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 25, 2021 - 15:06:49
Django version 3.2.7, using settings 'ProjetoES.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Fonte: Os autores.

- Visto que a aplicação está rodando, para acessar o OpenPDI, basta abrir o navegador e acessar o endereço <http://localhost:8000/>, como mostra a Figura 14 - acessando o OpenPDI no navegador, mostra o OpenPDI sendo acessado.

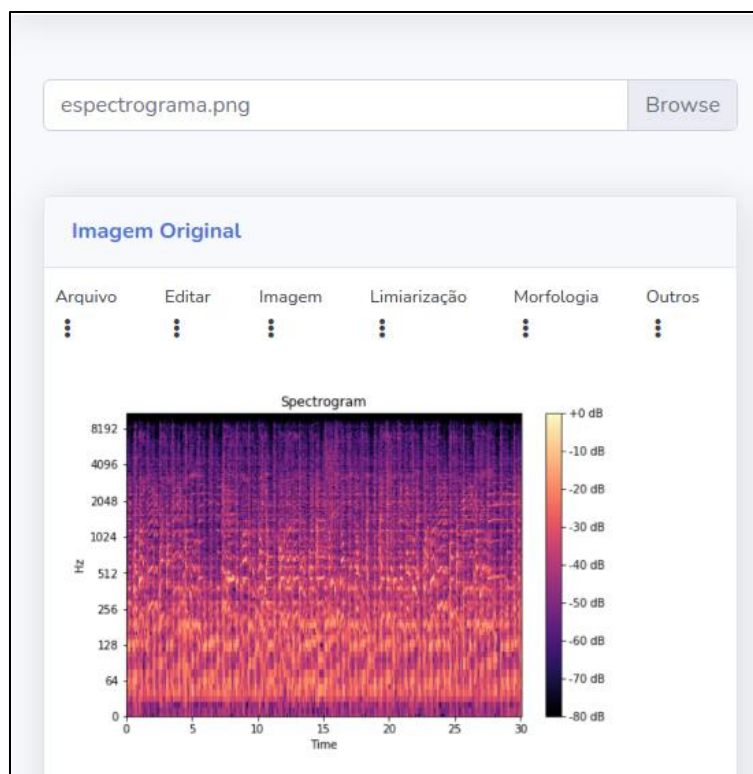
Figura 14 - Acessando o OpenPDI no navegador.



Fonte: Os autores.

Visto que a aplicação estiver rodando, é possível carregar uma imagem nova diretamente do computador pelo botão 'Browse', como mostra a Figura 15 - carregando uma nova imagem.

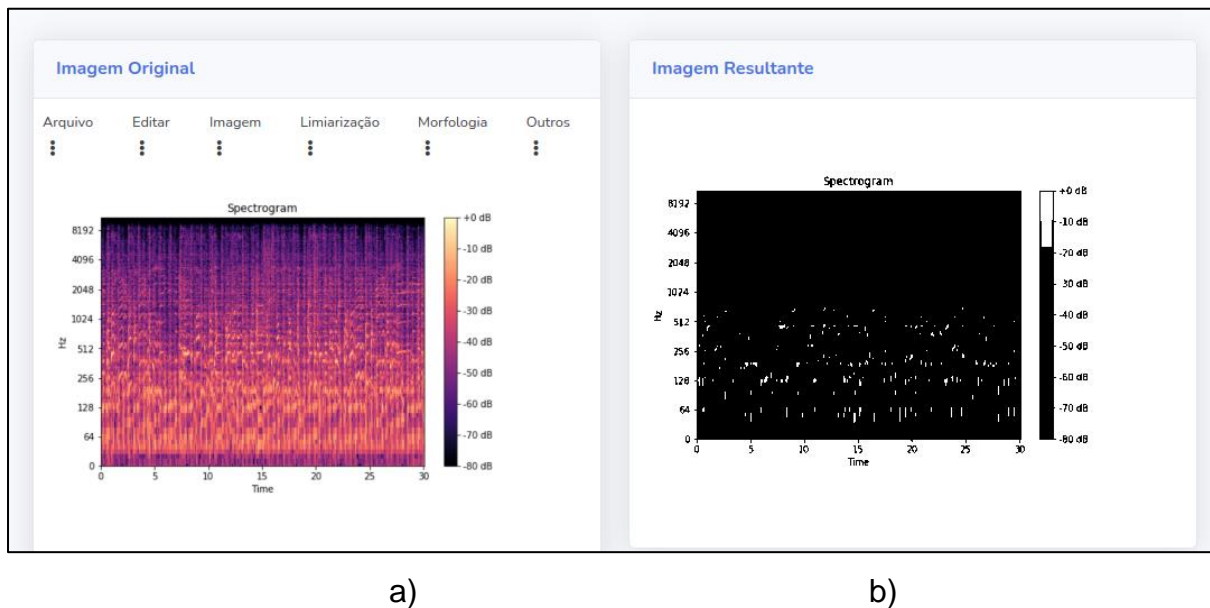
Figura 15 - Carregando uma nova imagem.



Fonte: OS autores.

Com a imagem carregada, é possível aplicar os filtros já existentes, no caso da Figura 16 - Aplicando filtro de limiarização otsu.

Figura 16 - a) Imagem Original; b) Aplicando filtro de limiarização otsu



Fonte: Os Autores


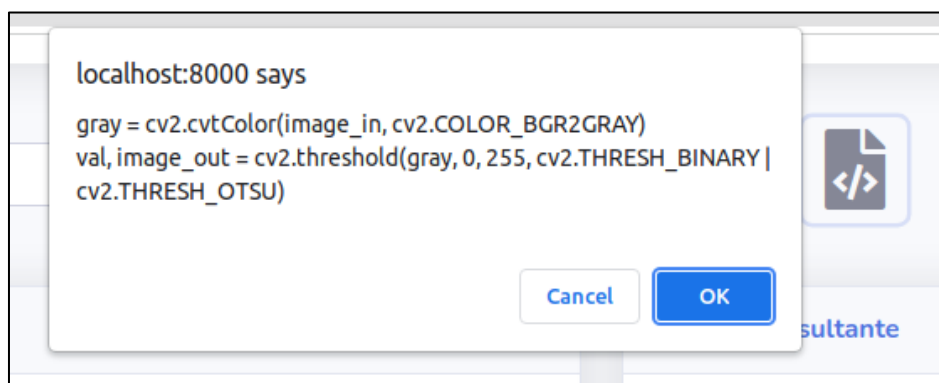
Clicando no ícone de código fonte (), é exibido o comando python que foi aplicado na imagem para gerar a imagem resultante, como mostra a Figura 17 - código aplicado na imagem.

Figura 17 - Código aplicado na imagem



Fonte: Os autores.


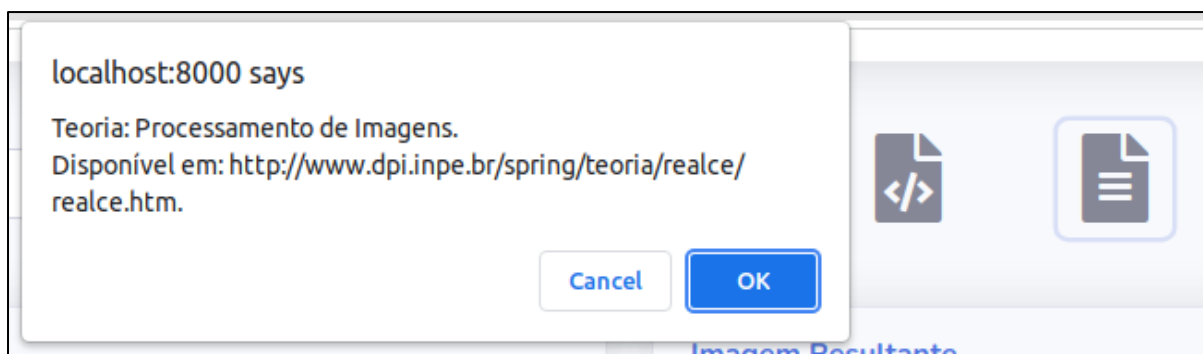
Clicando no ícone (), é exibido um link com mais informações sobre o método aplicado, como mostra a Figura 18 - Informações teóricas.

Figura 18 - Informações teóricas.



Fonte: Os autores.


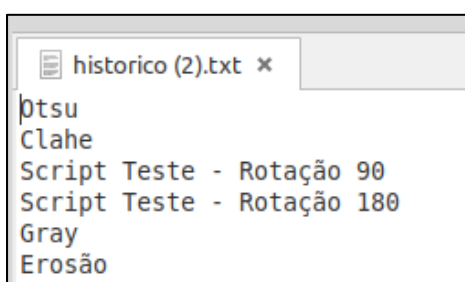
Clicando no ícone (), é realizado o download de um arquivo txt mostrando o que foi aplicado na imagem, conforme Figura 19.

Figura 19 - Histórico da operação.



DEMONSTRANDO OUTROS FILTROS, EROSÃO

A Figura 20 mostra a operação de erosão sobre uma imagem original.

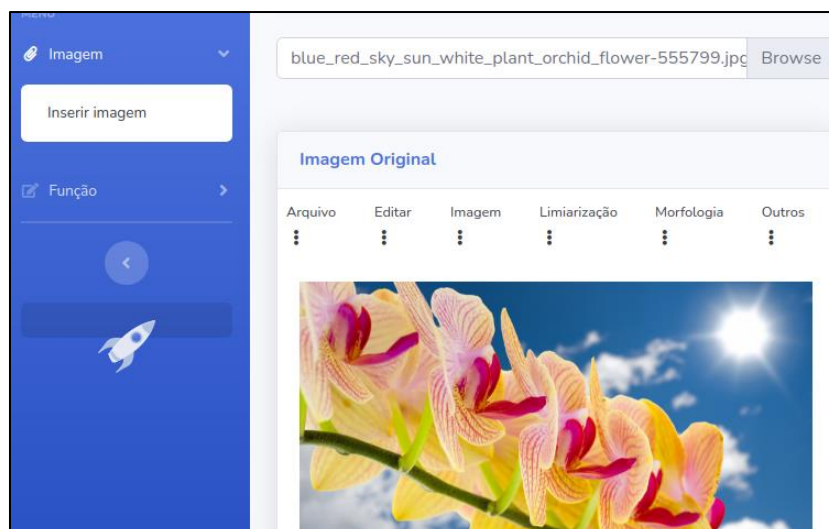
Figura 20 - Erosão.



Fonte: Os autores.

A opção 'inserir imagem' salva a imagem carregada no OpenPDI dentro da pasta do sistema, conforme Figura 21.

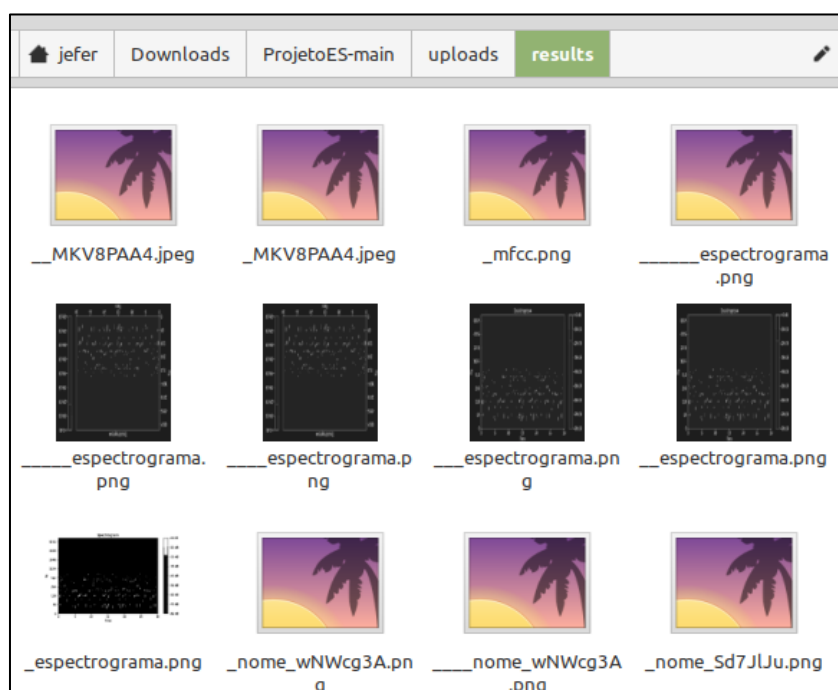
Figura 21 - Inserir Imagem.



Fonte: Os autores.

A pasta 'results' vai salvando cada nova imagem gerada, isso é observado na Figura 22.

Figura 22 - Pasta de armazenamento.



Fonte: Os autores.

Por questão de organização, cada operação aplicada na imagem, ela vai recebendo um incremento no nome do arquivo, assim como na Figura 23.

Figura 23 - Exemplo de incremento.

■	____espectrograma.png	...oES-main/uploads/results	16.2 kB
■	____espectrograma.png	...oES-main/uploads/results	72.9 kB
■	____espectrograma.png	...oES-main/uploads/results	72.9 kB
■	____espectrograma.png	...oES-main/uploads/results	73.0 kB
■	__espectrograma.png	...oES-main/uploads/results	29.7 kB
■	_espectrograma.png	...oES-main/uploads/results	4.3 kB

CADASTRANDO UMA NOVA OPERAÇÃO

A seguir na Figura 24 é possível observar a tela onde é possível adicionar uma nova operação.

Figura 24 - Adicionando uma nova operação.

Processamento Digital de Imagem

Inserir Filtro

Categoria*

Imagem

Escolha entre: Imagem, Limiarização, Morfologia ou alguma outra personalizada

Nome*

resize

Código*

```
maior = cv2.resize(gray, None, fx=1.5, fy=1.5, interpolation=cv2.INTER_CUBIC)
cv2_imshow(maior)
```

Fonte: Os autores.

A funcionalidade inserir filtro é responsável por permitir que seja incluída na ferramenta uma nova opção de filtros e operações. Nessa mesma página insere-se o código python que será aplicado sobre a imagem. Dessa forma é possível observar a padronalidade na ferramenta, onde em um único local é possível iniciar e finalizar uma operação.

Como extra, também foi produzido um Material de apoio intitulado “Complemento Teórico sobre Imagens Digitais”, de cerca de 60 páginas onde são apresentados conceitos de imagens digitais e dos filtros implementados. Está disponível em arquivo PDF com o git do projeto.

<https://docs.google.com/document/d/e/2PACX-1vSHFxWbxcwgVGa4nK-hH3gHQDhrjy5jLS6rcpzOa9-aQzv9INWDyCqJtkZcun3URONoFTajJDUI2koR/pub>

TRABALHOS FUTUROS

O OpenPDI apresentou resultados interessantes, suas funcionalidades já são muito úteis, o que leva os desenvolvedores a pensarem em novas possibilidades para este projeto. Dessa forma, considerando a continuidade deste projeto, propõem-se os seguintes trabalhos futuros.

- Gerenciamento de usuários;
- Permitir que diferentes usuários consigam trabalhar no mesmo projeto;
- Disponibilizar relatórios das ações realizadas na ferramenta;
- Implementação de métrica de avaliação de imagens;
- Desenvolvimento compartilhado da ferramenta por distintas instituições de ensino superior, com base em projetos que envolvam professores e alunos;
- Criação de uma comunidade que auxilie novas possibilidades na ferramenta.