

UNIVERSITATEA "ALEXANDRU-IOAN CUZA" DIN IASI

FACULTATEA DE INFORMATICA



LUCRARE DE LICENTA

Pixify - aplicație de procesare a imaginilor

propusă de

Mihaela Neculea

Sesiunea: iunie, 2023

Coordonator științific

Conf. Dr. Andreea-Valentina Arusoae

UNIVERSITATEA "ALEXANDRU-IOAN CUZA" DIN IASI

FACULTATEA DE INFORMATICA

**Pixify - aplicație de procesare a
imaginilor**

Mihaela Neculea

Sesiunea: iunie, 2023

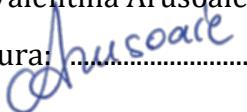
Coordonator științific

Conf. Dr. Andreea-Valentina Arusoae

Avizat,

Îndrumător lucrare de licență,

Conf. Dr. Andreea-Valentina Arusoae.

Data: 20.06.2023 Semnătura: 

Declarație privind originalitatea conținutului lucrării de licență

Subsemnata Neculea Mihaela domiciliată în România, jud. Neamț, Comuna Români, localitatea Goșmani, născută la data de 20 noiembrie 2001, identificată prin CNP 6011120270021, absolventă a Universității "Alexandru Ioan Cuza", Facultatea de informatică specializarea informatică, promoția 2023, declar pe propria răspundere cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art. 143 al. 4 și 5 referitoare la plagiat că lucrarea de licență cu titlu **Pixify - aplicație de procesare a imaginilor** elaborată sub îndrumarea doamnei Conf. Dr. Andreea-Valentina Arusoae, pe care urmează să o susțin în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consumând inclusiv la introducerea conținutului ei într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări și tințifice în vederea facilitării falsificării de către cumpărător a calității de autor ale unei lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data: 21.06.2023.....

Semnătura 

Declarație de consimțământ

Prin prezenta declar că sunt de acord ca lucrarea de licență cu titlul **Pixify - aplicație de procesare a imaginilor**, codul sursă al programelor și celealte conținuturi (grafice, multimedia, date de test, etc.) care însotesc această lucrare să fie utilizate în cadrul Facultății de informatică.

De asemenea, sunt de acord ca Facultatea de informatică de la Universitatea "Alexandru-Ioan Cuza" din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Mihaela Neculea**

Data: 21.06.2023

Semnătura: 

Cuprins

Motivătie	2
Introducere	4
1 Arhitectura Aplicației	6
1.1 Diagrama Use-case a Aplicației	7
2 Tehnologii utilizate	9
2.1 Angular	9
2.2 Flask	10
2.3 OpenCV	11
2.4 SQLAlchemy	12
2.5 Aplicații asemănatoare	13
3 Prezentarea Aplicației	17
3.1 Prezentarea interfeței disponibile utilizatorului	17
3.2 Extragerea metadatelor dintr-o imagine	17
3.3 Atașare de watermark	19
3.4 Detectarea nivelului de claritate	21
3.4.1 Tranformata Fourier	21
3.5 Vizualizarea pixelilor unei imagini pe un graf 3D	24
3.6 Reducerea noise-ului	26
3.6.1 Temporal Noise și Spatial Noise	26
3.7 Convertirea formatului	29
3.8 Eliminarea ochilor roșii din imagini	31
3.9 Login și Register	33
Concluzii	36

Motivație

În această lucrare de licență am abordat domeniul procesării de imagini care are o influență semnificativă asupra multor aspecte din viețile noastre, jucând un rol critic în societate. Producția și consumul de imagini crește pe zi ce trece, ca urmare a utilizării pe scară largă a camerelor digitale, a smartphone-urilor și a platformelor de social media.

Această idee a pornit din dorința de a oferi utilizatorilor, cum ar fi cei pasionați de fotografie sau cei curioși de domeniul procesării de imagini, o platformă care demonstrează mai multe usecase-uri. O persoană pasionată sau curioasă de acest domeniu și-ar putea dori să vadă unele procesări realizate pe imaginile proprii pentru a putea înțelege mai în profunzime unele aspecte, de aceea vreau să ofer utilizatorilor posibilitatea de a aplica aceste tehnici pe imaginile lor proprii în timp real. Realizarea acestor tehnici aduce o experiență mult mai plăcută unui fotograf sau unui artist în procesul de creare a unui produs, determinând și o creștere a numărului de persoane interesate de această parte a informaticii. Platforma pune la dispoziție posibilitatea de a încarcă o fotografie prin intermediul unei interfețe web și alegerea unei acțiuni pentru a fi realizate asupra acesteia.

Contribuții

În ziua de azi există pe piață numeroase de aplicații web sau mobile care se concentrează pe editarea sau procesarea imaginilor, de aceea ideea acestei aplicații nu este una originală. Cu toate acestea, pașii aleși pentru conceperea, modelarea și implementarea acestui proiect reprezintă idei proprii, folosind cunoștințele teoretice și practice dobândite pe parcursul facultății, dar și din studiul individual.

Aplicația pe care am dezvoltat-o aduce unui utilizator mai multe procesări ce pot fi aplicate unei imagini, dintre care țin să menționez: eliminarea ochilor roșii dintr-o poză ce a fost făcută cu blițul, eliminarea noise-ului dintr-o imagine și extragerea metadatelor.

În dezvoltarea acestei aplicații am folosit unelte și librării fară de care reușita acestui proiect nu ar fi putut fi dusă la bun sfârșit. Dintre acestea pot aminti: OpenCV(Open Source Computer Vision Library) care este o librărie open-source ce conține peste 2500 algoritmi folositori pentru procesarea imaginilor, framework-ul Angular care este folosit pentru dezvoltarea de aplicații web. Crearea interfeței pusă la dispoziția utilizatorului este o creație proprie, împreună cu modelul folosit în baza de date și endpoint-urile necesare comunicării de informații.

Prin munca depusă în crearea acestui proiect mi-am largit cunoștințele în aceste domenii și mi-am dezvoltat abilitățile ce țin de crearea unei aplicații web.

Introducere

În domeniul informaticii și tehnologiei, procesarea imaginilor, analiza, manipularea și îmbunătățirea imaginilor digitale fac parte din obiectivele principale și cel mai des utilizate. Înțelegerea modului în care funcționează imaginile și realizarea potențialului lor devin din ce în ce mai esențiale în societatea de astăzi condusă de vizual, unde miliarde de imagini sunt create și schimbate zilnic. Procesarea imaginilor este esențială într-o varietate de industrii, inclusiv realitatea augmentată, imaginile din satelit și imagistica medicală. Imaginile sunt mai mult decât simple reprezentări ale realității; ele conțin o mulțime de cunoștințe și potențial creativ care pot fi deblocate și explorate prin procesarea imaginilor.

Editarea și procesarea [11] de imagini ajută la rezolvarea unor probleme estetice pe care le pot avea imaginile. De exemplu, eliminarea unor efecte inestetice cum ar fi noise-ul (pixeli care fluctuează aleatoriu, producând zone neclare), îmbunătățirea clarității, eliminarea ochilor roșii creați de la lumina puternică a blițului. Acestea sunt doar câteva dintre problemele care pot fi rezolvate, acest domeniu fiind unul larg și în continuă dezvoltare.

Pentru acest proiect de licență am ales să dezvoltăm o aplicație web care se concentrează pe procesarea de imagini datorită multiplelor aspecte esențiale pe care le au imaginile în vietile noastre. În ziua de azi există pe piață o multitudine de aplicații web sau mobile de editare de imagini, ideea aplicației nefiind una originală, totuși pasii aleși pentru conceperea, modelarea și implementarea acestui proiect reprezintă idei personale.

Aplicația numită "Pixify" pune la dispoziția utilizatorilor o serie de operații bazate pe procesare de imagini. Un utilizator își poate face cont pe platformă, are opțiunea de a-și vedea contul și, de asemenea, de a-și edita informațiile personale. Are la dispozibilitate o secțiune unde i se oferă mai multe tipuri de procesări din care poate alege. Aceste opțiuni includ următoarele: extragerea metadatelor dintr-o imagine, adăugarea

unui watermark pe o imagine, aflarea nivelului de claritate a unei imagini, vizualizarea pixelilor unei fotografii drept un grafic 3D, convertirea formatului unei imagini, reducerea noise-ului, eliminarea ochilor roșii creați de la bliț dintr-o fotografie. Uneltele esențiale care au fost folosite în realizarea acestei aplicații sunt: framework-ul Angular care a fost folosit în realizarea interfeței disponibile utilizatorului, iar microframework-ul Flask a fost folosit pentru partea de server.

Această lucrare se întinde pe trei capitole, primul sub numele de "Arhitectura Aplicației" care conține o diagramă use-case ce prezintă interacțiunile existente dintre un utilizator și platformă, iar al doilea sub numele de "Aplicații asemănătoare" ce prezintă trei aplicații care au integrate aspecte din domeniul procesării de imagini și care au fost un exemplu pentru mine. Al treilea capitol, "Tehnologii Utilizate" prezintă uneltele esențiale care au dat formă acestei aplicații, dintre care pot aminte framework-ul Angular folosit pentru partea de frontend și microframework-ul Flask pentru serverul de backend. În ultimul capitol sunt explicate funcționalitățile, fiecare în parte cu exemple și detalii relevante legate de unii algoritmi mai complecsi folosiți.

Capitolul 1

Arhitectura Aplicației

Imaginile sunt reprezentări vizuale ale lumii din jurul nostru, surprinse prin diverse mijloace precum camerele foto sau generate de procese digitale. Acestea pot fi de numeroase tipuri și pot conține numeroase probleme, majoritatea inestetice. Uineltele care fac parte din procesarea imaginilor pot aborda provocări precum îmbunătățirea imaginii, contrastului sau reducerea zgomotului. De asemenea, poate aborda restaurarea imaginii, care implică eliminarea artefactelor sau imperfecțiunilor cauzate de factori precum estomparea mișcării sau limitările senzorului.

În această aplicație am dorit să oferim unui utilizator mai multe uinelte prin care ar putea repara unele probleme, cum ar fi: aflarea nivelului de claritate a unei poze pentru ca apoi să poată compara pozele între ele, reducerea unor aspecte inestetice, dar și aspecte legate de copyright sau de metadatele pe care le conțin imaginile.

Arhitectura unei aplicații web cuprinde, în general, structura, interacțiunile posibile dintre componentele esențiale ale aplicației și organizarea ei. Vom enumera criteriile de bază ce trebuie îndeplinite de orice platformă web pentru a oferi utilizatorilor securitate, control și o bună experiență în utilizarea acesteia, pe care le-am introdus în aplicație:

- stocarea persistentă a datelor
- prevenirea accesului user-ilor neautorizați
- interfață grafică ușor de navigat și de înțeles
- timp de răspuns rapid la comenziile utilizatorului
- înregistrarea corectă a erorilor

- operații fundamentale asupra datelor.

1.1 Diagrama Use-case a Aplicației

În această secțiune, pentru a înțelege mai în profunzime funcționalitățile existente ale aplicatiei Pixify vom prezenta o diagramă use-case a aplicatiei pe care o vom explica pe scurt.



Figura 1.1: Diagrama Use-case

După cum se poate observa din Figura 1.1, pentru ca un utilizator să aibă acces la operațiile de procesare acesta trebuie să se logheze cu un nume de utilizator și o parolă. Dacă acesta nu are un cont deja existent va trebui să își creeze cont introducând unele date personale. După crearea contului utilizatorul se poate loga ajungând apoi în pagina principală unde i se oferă la dispoziție tipurile de procesări în partea stângă. Fiecare funcționalitate are un flow asemănător deoarece utilizatorul selectează optiunea sa de procesare, apoi este redirectionat pe o nouă pagină unde trebuie să încarce o poză care dorește să fie editată, urmând să o primească înapoi după câteva secunde. Optiunile pe care le are un utilizator sunt următoarele: extragerea metadatelor dintr-o imagine, adăugarea unui watermark pe o imagine, aflarea nivelului de claritate a unei imagini, vizualizarea pixelilor unei fotografii drept un grafic 3D, convertirea formatului unei imagini, reducerea noise-ului, eliminarea ochilor roșii creați de la bliț dintr-o fotografie.

Capitolul 2

Tehnologii utilizate

În acest capitol vom prezenta tehnologiile folosite în realizarea platformei Pixify și aplicațiile asemănătoare care rezolvă aceleași fel de probleme ca aplicația de față. Arhitectura aplicației pe care am dezvoltat-o a fost realizată cu ajutorul a două părți: front-end și back-end.

Pentru partea de front-end am ales să utilizăm framework-ul Angular deoarece promovează reutilizarea codului, menținabilitatea și testabilitatea, permitând mai multor dezvoltatori să lucreze simultan la diferite module ale aplicației. Iar pentru partea de server vom folosi Flask care este cunoscut pentru că oferă utilizatorilor un API intuitiv și simplu, permitându-le să creeze aplicații web cu ușurință. Simplitatea sa nu numai că accelerează procesul de dezvoltare, dar îmbunătățește și menținabilitatea generală a codului.

2.1 Angular

Angular a fost creat și este în continuare întreținut de Google. A fost disponibil pentru prima dată în 2010 ca AngularJS și a suferit o revizuire completă înainte de a fi disponibil în 2016 ca Angular (cunoscut și ca Angular 2+). Angular este un cadru pentru crearea de aplicații pe o singură pagină (SPA) bazate pe componente, care sunt dinamice, scalabile și de încredere.

De la debut, Angular (a se vedea Figura 2.1) a devenit un cadru major în industria dezvoltării web și este utilizat de mai multe companii din întreaga lume. Cu o selecție largă de biblioteci, instrumente și sprijin comunitar, are un ecosistem înfloritor. Angular folosește TypeScript, un superset de JavaScript tip static, care oferă limbajului o veri-

ficare mai bună a tipului și mai multă funcționalitate. Ca rezultat, dezvoltatorii pot construi un cod mai scalabil și mai ușor de întreținut și pot descoperi defecțiuni mai devreme în procesul de dezvoltare.

Utilizatorii pot scrie cod JavaScript mai ușor de înțeles utilizând TypeScript, care specifică un set de tipuri pentru JavaScript. Orice platformă poate suporta cu ușurință funcționarea fără întreruperi a codului TypeScript datorită compilării JavaScript.



Figura 2.1: Angular <https://www.toptal.com/angular-js/top-18-most-common-angularjs-developer-mistakes>

2.2 Flask

Flask a fost creat de Armin Ronacher, care a fost liderul echipei Pocco, un grup internațional de entuziaști ai Python-ului. Setul de instrumente Werkzeug WSGI și motorul de şablon Jinja2 servesc drept fundație pentru Flask. Ambele sunt proiecte Pocco.

Deși este un microframework, întreaga aplicație nu trebuie să fie conținută într-un singur fișier Python. Cadrul Flask este simplu, dar extensibil deoarece este „micro”. Flask nu va lua nicio decizie pentru dvs., cum ar fi ce bază de date să utilizați sau dacă să utilizați un ORM. Flask oferă un cadru puternic pentru crearea de aplicații web și oferă dezvoltatorilor libertatea de a adăuga anumite biblioteci și extensii în conformitate cu nevoile lor. Datorită adaptabilității sale, Flask este ideal pentru o varietate de sarcini, de la site-uri web personale mici la aplicații complexe.

Flask pune un accent puternic pe existența unui API simplu și intuitiv, ceea ce îl face ușor de înțeles și utilizat de către dezvoltatori. Utilizează un sistem de rutare care face posibilă conectarea punctelor finale URL la funcțiile Python cunoscute sub numele de funcții de vizualizare, care sunt responsabile de gestionarea cererilor HTTP și de furnizarea de răspunsuri. Acest lucru face simplă crearea a numeroase funcționalități, inclusiv procesarea formularelor, autentificarea și integrarea bazei de date, precum și proiectarea rutelor și gestionarea mai multor tipuri de solicitări.

2.3 OpenCV

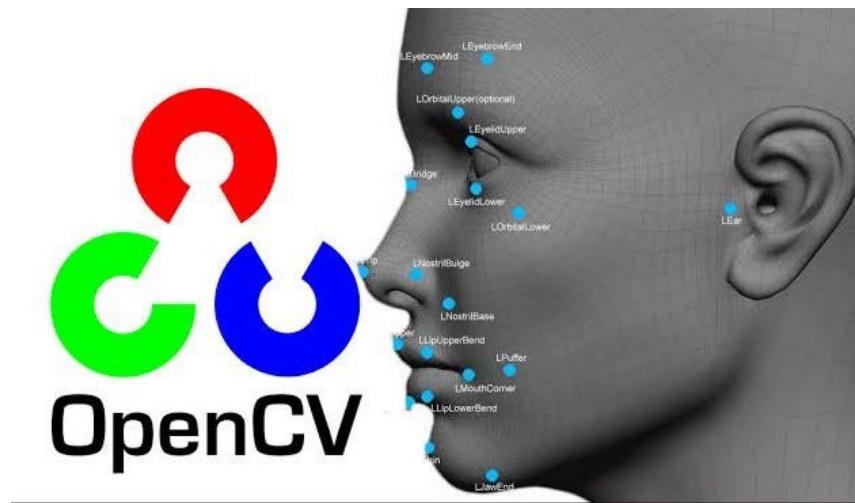


Figura 2.2:

OpenCV

<https://levelup.gitconnected.com/mastering-opencv-with-python-a-comprehensive-guide-for-image-processing-in-100-lines-of-code-100f33a2a2>

În realizarea aplicației am utilizat și OpenCV, o bibliotecă de software pentru vizuirea computerizată și învățarea automată ce este gratuită și open source. Scopul OpenCV a fost de a accelera adoptarea inteligenței artificiale în bunurile comerciale, oferind o bază comună pentru aplicațiile de computer vision. OpenCV face codul simplu de utilizat și modificat de către companii, deoarece este un produs cu licență Apache 2. Suportă Windows, Linux, Android și Macos și oferă interfețe pentru C++, Python, Java și MATLAB.

Peste 2500 de algoritmi optimizați sunt disponibili în colecție, inclusiv o gamă largă de tehnici tradiționale și de ultimă oră de viziune computerizată și învățare automată. Acești algoritmi pot fi utilizați pentru a găsi imagini similare dintr-o bază de date de imagini, pentru a elimina ochii roșii din fotografiile realizate cu bliț, pentru a urmări mișcările ochilor, pentru a recunoaște peisajele și pentru a stabili markeri de

suprapunere. Ele pot fi, de asemenea, utilizate pentru a detecta și recunoaște fețe, identifica obiecte, clasifica acțiunile umane în videoclipuri, urmărirea mișcărilor camerei, urmărirea obiectelor în mișcare, extragerea modelelor 3D ale obiectelor, producerea de nori de puncte 3D din camere stereo, unirea imaginilor împreună pentru a produce o rezoluție înaltă.

2.4 SQLAlchemy

SQLAlchemy este o bibliotecă open-source populară și puternică pentru Python, care oferă un cadru ORM (Object-Relational Mapping), permitând dezvoltatorilor să lucreze cu baze de date folosind obiecte Python în loc să scrie interogări SQL brute. Printre numeroasele sisteme de baze de date pe care le acceptă se numără PostgreSQL, MySQL, SQLite și Oracle. Pentru a facilita dezvoltarea codului eficient și care poate fi întreținut, SQLAlchemy oferă un API de nivel înalt și expresiv, care simplifică interacțiunile cu bazele de date și retrage numeroase complexități. Oferă capabilități precum creația automată a tabelelor, migrarea schemei, interogare flexibilă și simplă și gestionarea tranzacțiilor. SQLAlchemy este un instrument flexibil pentru lucrul cu baze de date în Python [10], deoarece acceptă idei avansate. În aplicația noastră am folosit SQLAlchemy pentru crearea bazei de date.

2.5 Aplicații asemănătoare

În această secțiune vom prezenta câteva aplicații asemănătoare care se focusează tot pe procesarea și editarea imaginilor.

Cloudinary

O platformă pentru gestionarea media în cloud numită Cloudinary are capacitați puternice de procesare de imagini. Poate fi utilizată în aplicații web și mobile, de asemenea oferă API-uri și instrumente pentru încărcarea, modificarea și optimizarea fotografiei.

Imaginiile pot fi modificate dinamic de către dezvoltatori prin decuparea, rotirea și modificarea calității. Oferă posibilități de adăugare de filtre, efecte și suprapuneri și acceptă o serie de tipuri de imagini. Pe lângă suportul pentru manipularea imaginilor, Cloudinary acceptă și transformarea video. Dezvoltatorii au capacitatea de a ajusta mai multe setări video, cum ar fi rezoluția, formatul și codificarea, precum și de a extrage cadre video, de a crea miniaturi video, de a aplica efecte și multe altele.

Funcționalitate bazată pe inteligență artificială: Cloudinary folosește învățarea automată și inteligența artificială (AI) pentru a oferi funcționalități de ultimă oră. Are funcții precum recunoașterea facială pentru a găsi și analiza chipuri, etichetare și clasificare automată a imaginilor, moderarea conținutului pentru a găsi informații delicate și multe altele. Securitate avansată și control al accesului: pentru a garanta confidențialitatea și integritatea activelor media, Cloudinary acordă o prioritate ridicată securității și oferă caracteristici precum jetoane de acces securizat, autentificare și criptare. În plus, oferă dezvoltatorilor funcții detaliate de control al accesului, care le permit să gestioneze cine poate accesa și lucra cu activele media.

Brandfolder

O platformă DAM (Digital Asset Management) bazată pe web, numită Brandfolder, este dedicată gestionării, distribuirii și organizării activelor mărcii, cum ar fi imagini, videoclipuri, documente și alte fișiere media. Deși procesarea avansată a imaginii nu este punctul principal al Brandfolder, aceasta oferă mai multe funcții legate de imagine care ajută la distribuirea și optimizarea eficientă a informațiilor vizuale. Utilizatorii Brandfolder pot eticheta și adăuga metadate la elementele lor de imagine.

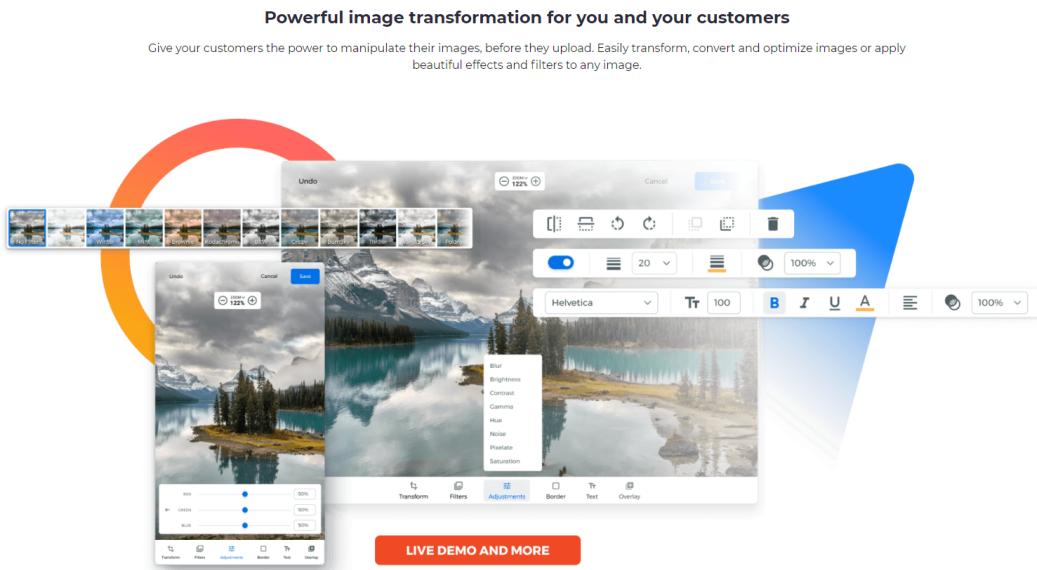


Figura 2.3: Clouddinary <https://www.filestack.com>

Acste metadate pot conține date precum descrierii de imagini, cuvinte cheie, drepturi de utilizare și informații despre drepturile de autor. Etichetele fac mai simplă găsirea unui or activ în depozitul Brandfolder, ajutând la căutarea imaginilor, filtrarea și clasificarea.

Pentru a asigura performanță maximă pe toate platformele și dispozitivele, Brandfolder optimizează și convertește automat fotografiile încărcate în diferite tipuri și dimensiuni de fișiere. Are capacitatea de a crea imagini în miniatură, de a redimensiona și de a comprima imaginile pentru a reduce dimensiunea fișierului fără a sacrifica în mod semnificativ calitatea. Transmiterea rapidă și eficientă a imaginilor este posibilă prin această funcționalitate, în special pentru aplicațiile bazate pe web.

Adobe Photoshop

Adobe Photoshop este un software folosit pentru editarea imaginilor digitale pe calculator, produs și distribuit de compania americană Adobe Systems și care se adresează în special profesionistilor domeniului. Are o istorie lungă care datează de câteva decenii și a fost creat de Adobe Systems. Thomas Knoll, doctorat la Universitatea din Michigan, student și fratele său John Knoll, un supervisor de efecte vizuale, l-au dezvoltat pentru prima dată în 1987. Programul, care a fost cunoscut pentru prima dată sub numele de „Display”, a fost creat ca instrument pentru afișarea imaginilor în tonuri de gri pe un monitor în alb-negru. Dar frații Knoll au văzut rapid potențialul

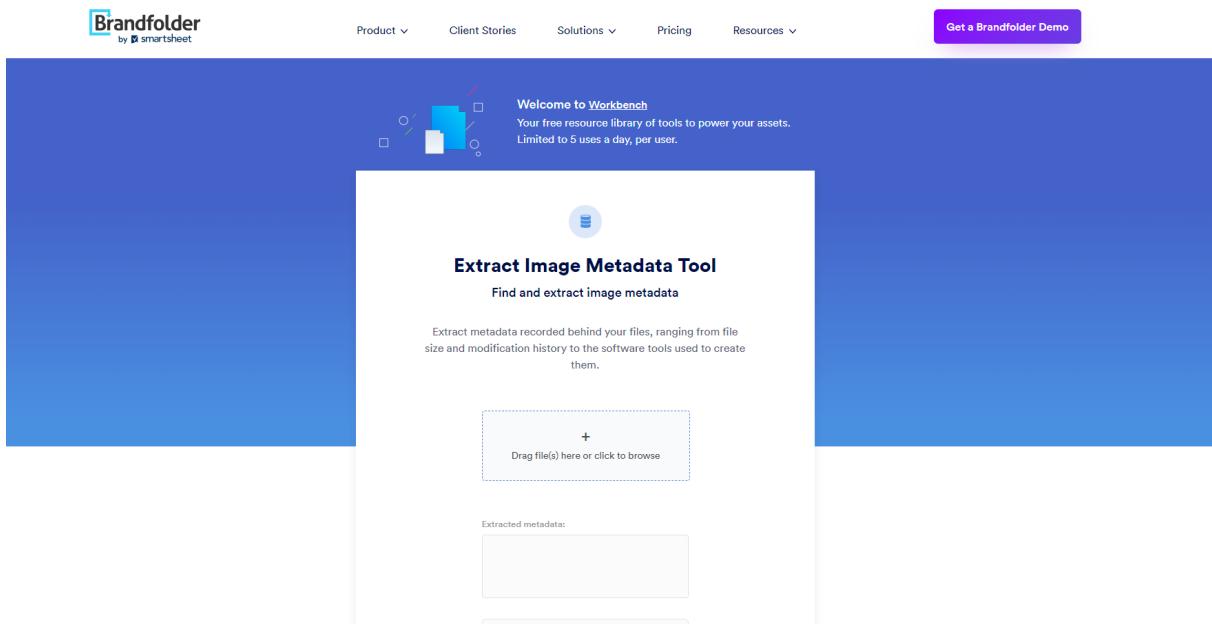


Figura 2.4: Brandfolder <https://brandfolder.com>

invenției lor și au luat decizia de a dezvolta un instrument complet de editare a imaginilor.

Adobe Photoshop este bine cunoscut pentru capabilitățile sale puternice de editare și manipulare a imaginilor. Utilizatorii pot crea, edita și îmbunătăți cu ușurință fotografii digitale folosind gama sa largă de instrumente și capabilități. Editările și modificările independente care pot fi făcute diferitelor aspecte ale unei imagini folosind sistemul de editare bazat pe straturi Photoshop oferă un control și o flexibilitate mai mare. Diferitele instrumente de selecție permit direcționarea exactă a anumitor zone pentru modificare. În plus, funcțiile de editare a imaginilor din Photoshop sunt excelente pentru a elimina defecte, pete și obiecte nedorite.

Software-ul oferă, de asemenea, o gamă largă de filtre și efecte care pot fi folosite pentru a modifica aspectul și pentru a adăuga fler artistic fotografiilor. Folosind diferitele instrumente de ajustare oferite, utilizatorii pot modifica cu precizie tonurile și aspectele de culoare ale fotografiilor lor. Îmbinarea netedă a mai multor fotografii și modificarea perspectivelor sunt posibile atât de caracteristicile de compoziție ale Photoshop. Caracteristicile textului și tipografiei fac posibilă încorporarea și formatarea textului, ceea ce le face perfecte pentru dezvoltarea de design-uri vizuale care conțin elemente textuale. Photoshop permite versatilitatea în salvarea și exportul fotografiilor din diverse motive, datorită suportului său pentru o gamă largă de formate de fișiere. În general, Adobe Photoshop oferă o platformă flexibilă de procesare a imaginilor care

permite utilizatorilor să-și exprime creativitatea și să obțină rezultatele vizuale dorite.

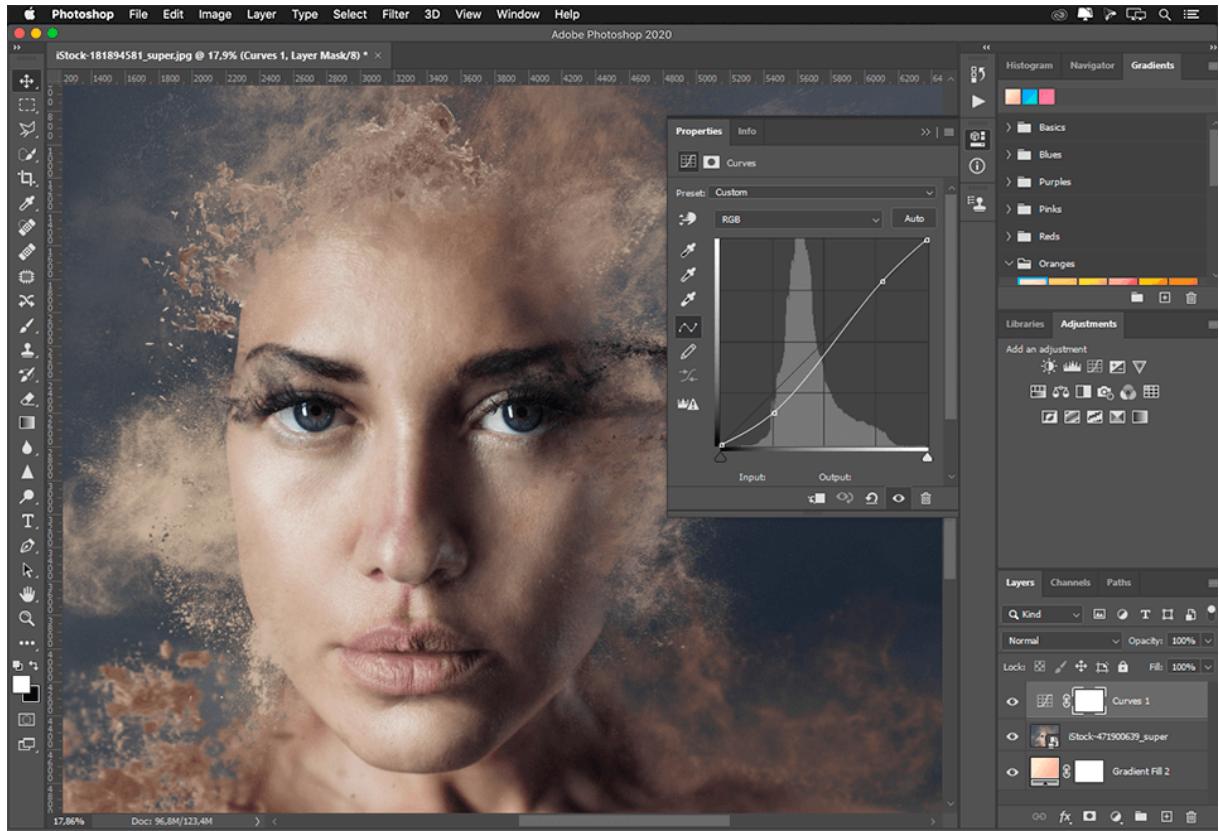


Figura 2.5: Photoshop <https://www.adobe.com/ro/products/photoshop.html>

Capitolul 3

Prezentarea Aplicației

În acest capitol vom intra în detaliu în legătură cu implementarea fiecărui dintr-unul dintre cele două module, a algoritmilor folosiți și a interfeței disponibile utilizatorului. Aplicația oferă unui utilizator mai multe unele cu care ar putea repara probleme pe care le pot avea imaginile, cum ar fi: aflarea nivelului de claritate al unei poze pentru ca apoi să poată compara pozele între ele, reducerea unor aspecte inestetice, dar și aspecte legate de copyright sau de metadatele pe care le conțin imaginile.

3.1 Prezentarea interfeței disponibile utilizatorului

Pentru a avea acces la operațiile din cadrul aplicației un utilizator trebuie să își creeze cont, apoi să se logheze. Odată autenticat, utilizatorul este plasat pe pagina principală unde are acces la toate efectele oferite de platforma. În partea stânga găsește un sidebar cu toate posibilele acțiuni. În partea de sus se află header-ul care conține în partea stânga logo-ul aplicației, iar în dreapta găsim trei butoane, primul "Actions" care duce la secțiunea de editare, un buton de "Profile" care duce la profilul utilizatorului și ultimul de "Logout", după cum se poate vedea în Figura 3.1.

3.2 Extragerea metadatelor dintr-o imagine

Toate imaginile au incorporate anumite informații care nu pot fi văzute cu ochiul liber, acestea se numesc metadate. Pot conține detalii despre modelul camerei, opțiunile de expunere, locația GPS, data și ora fotografierii, precum și detalii despre drepturile de autor. **Exif** (Exchangeable picture File Format) sau **IPTC** (International Press Tele-

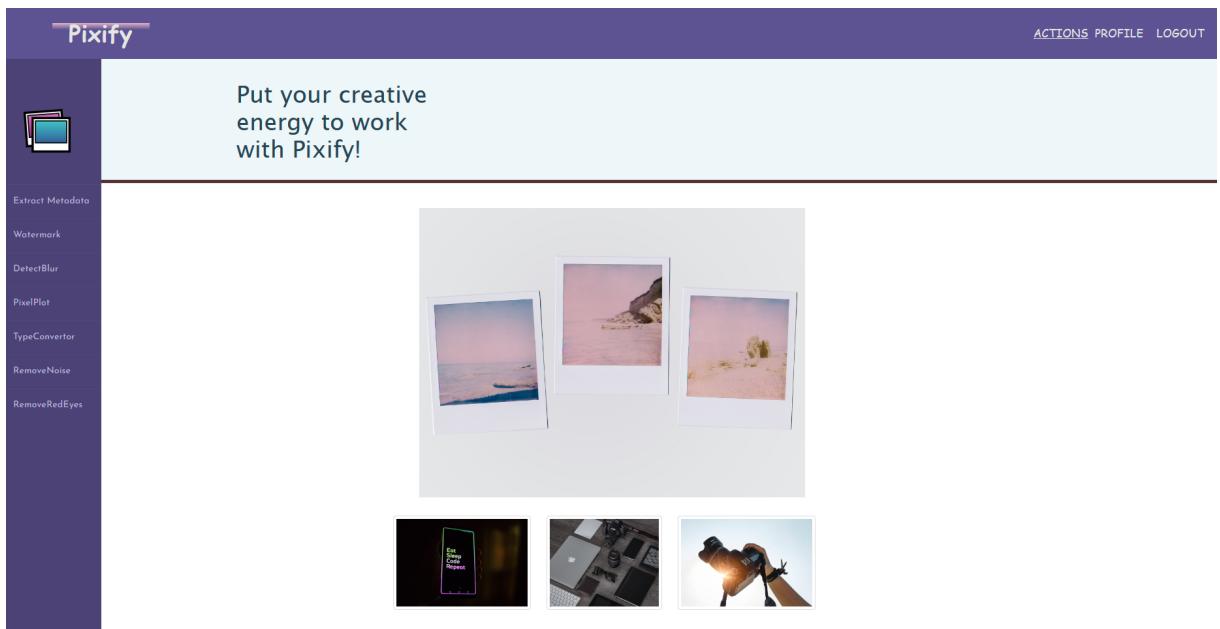


Figura 3.1: Pagina de "Actions"

communications Council) sunt standarde comune care sunt utilizate pentru a păstra metadatele.

Aproape toate camerele digitale creează aceste fișiere de date de fiecare dată când faceți o nouă poză. Exif cuprinde în principal datele tehnice despre o imagine, în timp ce IPTC se concentrează pe descrierea imaginii și a metadatelor sale administrative și de drepturi. Informații legate de copyright pot fi găsite într-o imagine și, de asemenea, despre autorul fotografiei, de exemplu numele acestuia, dispozitivul cu care a fost făcută poza.

Există acest termen de "magic number" care se referă la o specifică secvență de octeți aflată la începutul unui fișier care ajută la determinarea formatului acestuia. Pentru fiecare tip de fișier există o secvență diferită. Octetii aceștia au un rol crucial în extragerea metadatelor deoarece ajută la determinare tipului de fișier, iar extragerea metadatei dintr-o poză diferă de la format la format. În tabela 3.1 se pot observa câteva formate de fișier și respectiv numerele magice asociate acestora. Extragerea datelor de tip Exif a fost realizată cu ajutorul funcției *getexif()* ce ține de libraria Pillow, librărie ce aduce suport pentru deschiderea și manipularea imaginilor.

Pentru a fi redirectionat la secțiunea de extragere de metadate utilizatorul trebuie să acceseze din sidebar-ul pus la dispoziție opțiunea de "Extract Metadata", apoi în noua pagină poate să încarce o poză apăsând pe butonul de "Upload a picture", urmând să i se deschidă o fereastră de unde poate selecta o poză din propriul device.

Tabela 3.1: Magic number based on type

File Format	Typical Extension	Hex digits
JPEG File	.jpg	ff d8 ff e0
PNG	.png	89 50 4e 47
GIF	.gif	47 49 46 38
Bitmap	.bmp	42 4d
TIFF	.tif	4d 4d 00 2a
Xfig	.fig	23 46 49 47
FITS	.fits	53 49 4d 50 4c 45

În cazul în care selectează un fișier care nu este o poză se va afișa eroare "Only images are supported". După ce imaginea a fost selectată, iar imaginea încărcată în aplicație, utilizatorul poate extrage metadata apăsând pe butonul "Get the metadata", apoi după câteva secunde va primi metadatele referitoare la poza selectată, pe care le poate copia în clipboard apăsând pe butonul de "Copy".

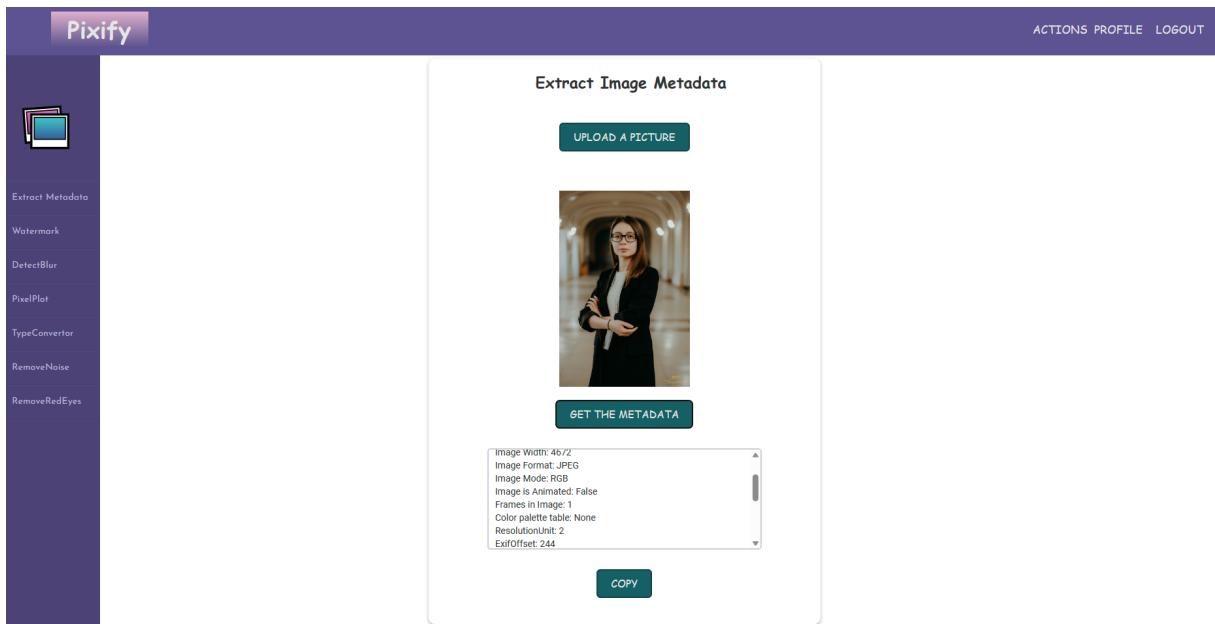


Figura 3.2: Extragerea Metadatelor

3.3 Atașare de watermark

O altă funcționalitate pe care o are aplicația noastră, este cea de atașare de watermark. Watermarking este procesul de atașare a unui logo sau unei bucați de text pe

un document sau pe o poză și este un proces important când vine vorba de protecția copyright sau de marketing. Deși cea mai mare parte a atașării unui watermark în zilele noastre se face digital, cuvântul „watermarking” există de secole. În trecut unei poze i se atașa un watermark când aceasta era umedă, doarece acesta era vizibil doar în anumite condiții de iluminare.

Adăugarea unui watermark [7] este un mijloc de a vă promova numele/brand-ul și de a crește recunoașterea acestuia, la fel cum un pictor își poate semna opera de artă. Există mai multe tipuri de a îți pune amprenta pe o imagine, cum ar fi: text, logo, pattern. În aplicația noastră un utilizator, după ce merge la secțiunea numită “Watermark”, trebuie să introducă textul care dorește să fie imprimat pe imagine și să selecteze o imagine apăsând pe butonul de sub titlu. De asemenea și în acest caz i se va afișa o eroare dacă selectează altceva încă din afară de o imagine sau dacă nu introduce un text pentru watermark. După trecerea a câtorva secunde, în partea dreapta utilizatorul va primi poza având atașată pe ea watermark-ul ales de el în trei poziții, acestea fiind colț stânga sus, colț dreapta jos și în mijloc (a se vedea Figura 3.3). Aceste poziții au fost calculate în funcție de lățimea și înălțimea imaginii, iar textul a fost rotit cu 15 grade.

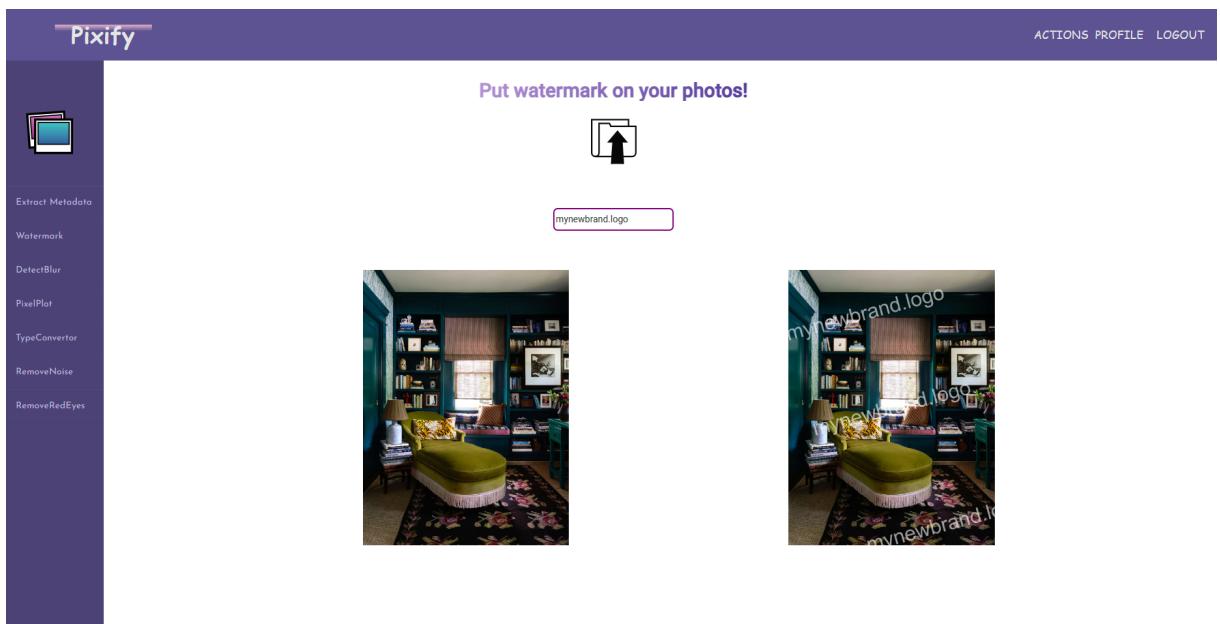


Figura 3.3: Exemplu de watermarking

3.4 Detectarea nivelului de claritate

Această funcționalitate a fost creată cu scopul de a oferi unui utilizator posibilitatea de a afla nivelul de claritate al unei imagini. De exemplu, dacă un fotograf care deține mai multe fotografii dorește să afle care dintre ele este cea mai clară, poate compara media de claritate a pozelor oferite de aplicație.

Una dintre modalitățile de realizare a acestui proces este folosind transformata Fourier care este un instrument matematic utilizat în procesarea imaginilor pentru diverse aplicații. Transformata Fourier ne permite să analizăm conținutul de frecvență al unei imagini și să efectuăm operațiuni precum filtrarea, compresia și îmbunătățirea calității. În secțiunea ce urmează vom prezenta mai detaliat această transformată.

3.4.1 Transformata Fourier

Transformata Fourier [8] a fost numită astfel după matematicianul și fizicianul Joseph Fourier [1] și este o operație care se aplică unei funcții complexe și produce o altă funcție conținând aceeași informație ca funcția inițială, dar reorganizată după frecvențele componente.

În domeniul procesării de imagini această tranformată este de o importanță deosebită deoarece poate fi folosită într-o gamă largă de aplicații, cum ar fi: analiza imaginilor [9], filtrarea imaginilor, reconstrucția și compresia imaginilor. Transformata este folosită pentru a descompune o imagine în componente sale de sinus și cosinus. Rezultatul reprezintă imaginea în domeniul de frecvență, în timp ce imaginea originală reprezinta domeniul spațial. Fiecare punct reprezintă o frecvență în domeniul Fourier. În figura 3.4 se poate observa cum arată o imagine după ce a fost aplicată Transformata Fourier asupra ei.



Figura 3.4: Fast Fourier Transform https://akshaysin.github.io/fourier_transform.html

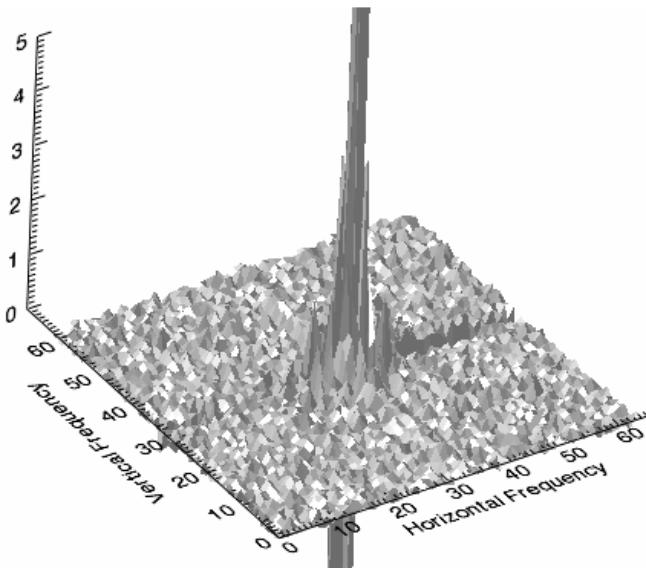


Figura 3.5: Data Frequency <https://www.13harrisgeospatial.com/docs/BackgroundFastFourierTransform.html>

Atunci când se aplică transformata cele mai joase frecvențe sunt adesea afișate în centrul datelor, iar cele mai mari pe margini. În Figura 3.5 se poate observa rezultatul unei funcții FFT(Fast Fourier Transform) reprezentat ca o suprafață, având originea în punctul (0,0), mărimea frecvenței crescând odată cu distanța față de origine.

NumPy [2] este o librărie din Python foarte populară creată în 2005 care oferă utilitarilor o gamă largă de acțiuni asupra vectorilor și multe funcții din domeniul algebrei liniare, matricelor și de asemenea legate de Transformata Fourier. Funcțiile legate de FFT din librărie, cum ar fi "numpy.fft.fft()", "numpy.fft.fft2()" sau "numpy.fft.fftshift()", permit efectuarea de calcule pe matrici unidimensionale și, respectiv, bidimensionale.

În următorul paragraf vom detalia algoritmul care calculează nivelul de claritate al unei poze folosit în această aplicație. Inițial sunt calculate coordonatele centrului imaginii deoarece sunt folosite la detectarea mediei, apoi este calculat vectorul numpy 2D al imaginii după ce a fost aplicată transformata. Elementele matricei sunt rearanjate de-a lungul fiecărei axe, reposiționând componente de joasă frecvență în centru pentru ca apoi acestea să fie înlocuite cu valoarea 0 cu scopul de a reduce frecvențele joase. Pentru a putea fi determinat nivelul de claritate este aplicată inversa FFT pentru reconstruirea domeniului spațial al imaginii. Poza obținută este apoi utilizată pentru calcularea spectrului de magditudine, care deține informații despre puterea sau intensitatea diferențelor frecvențe din imagine $magnitude = 20 * \text{numpy.log}(\text{numpy.abs}(\text{reconstruct}))$. Magnitudinea obținută din coeficienții Fourier

se poate întinde pe o scară largă de valori. Aplicând logaritmul, intervalul dinamic este comprimat, permitând o vizualizare și analiză mai bune. Această compresie poate îmbunătăți vizibilitatea atât a valorilor de magnitudine mică, cât și a celor mari. Înmulțirea cu 20 este o practică obișnuită pentru a scala mărimele în scopuri de vizualizare. Valorile de magnitudine mai mari indică frecvențe mai proeminente, pe când valorile mai mici indică valori mai slabe. În final este calculată media valorilor din spectrum: $mean = \text{numpy.mean}(magnitude)$. Valoarea medie obținute este comparată cu un prag (în cazul nostru în valoare de 30 ales), dacă valoarea este mai mică de prag poza este considerată blurată, iar în caz contrar aceasta este considerată clară.

Pentru a ajunge la aflarea mediei de claritate un utilizator trebuie să selecteze opțiunea de "DetectBlur" din sidebar-ul aflat în partea stângă a ecranului, apoi după ce încarcă o fotografie din dispozitivul personal, urmând aceiași pași explicați și în secțiunile anterioare, va primi înapoi poza și un mesaj dacă aceasta a fost considerată clara sau nu. Ii este afișată de asemenea și media la care s-a ajuns după calcule, astfel putând compara valorile obținute la mai multe poze, cea cu media cea mai mare fiind cea mai clară. După cum puteți observa din captura de ecran de mai jos (Figura 3.6), poza a fost clasificată ca fiind una blurată, având calcului mediei în valoare de: 9.79997.

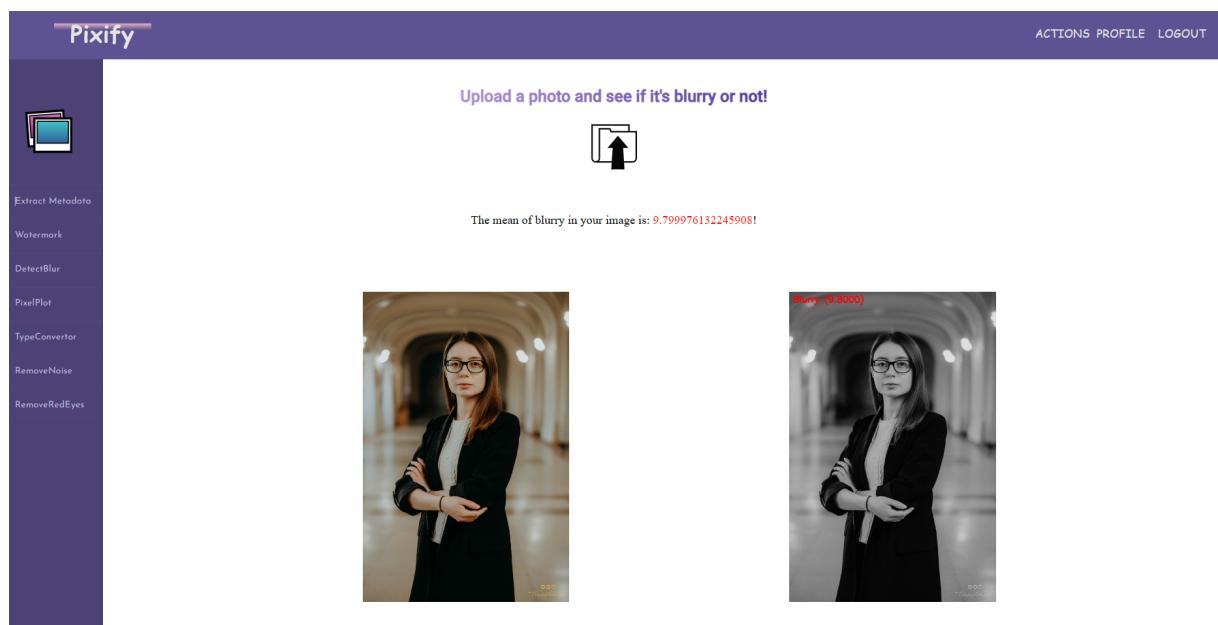


Figura 3.6: Blur Detection

3.5 Vizualizarea pixelilor unei imagini pe un graf 3D

Aplicația Pixify oferă unui utilizator posibilitatea de a vizualiza o poza drept un grafic 3D format din pixelii imaginii. Această parte ar putea fi dezvoltată în proiecte următoare deoarece observarea unui astfel de grafic poate fi utilă în mai multe situații și domenii. Câteva exemple ar putea fi: analiza caracteristicilor unei poze prin abilitatea de a roti, naviga, a mări și a observa diferite tipare în anumite zone din graficul format din pixeli. O altă situație în care ar putea fi de folos această procesare este în domeniul artistic, prin faptul că transformă o poza din spațiul 2D în unul 3D, permitând crearea de noi forme artistice și de vizualizare.

Pentru a ajunge la această transformare a unei imagini un utilizator trebuie să selecteze opțiunea de "PixelPlot" din sidebar-ul pus la dispozitie. După ce poza a fost selectată din propriul dispozitiv, utilizatorul va primi înapoi după câteva secunde graficul 3D având drept valori pixelii imaginii.

Graficul are valorile pentru cele trei axe de coordonate (x , y și z) în intervalul $[0, 255]$. Acest interval este asociat cu modelul de culoare RGB (Roșu, Verde, Albastru), care este utilizat pe scară largă pentru reprezentarea culorilor în sistemele digitale. Fiecare culoare din acest model poate avea o valoare între 0 și 255, reprezentând luminozitatea și intensitatea canalului respectiv. Valoare 0 reprezintă lipsa culorii respective, iar 255 denotă prezența maximă a culorii în intensitate. În Figura 3.7 este reprezentată opțiunea PixelPlot aplicată pe imaginea din Figura 3.8.

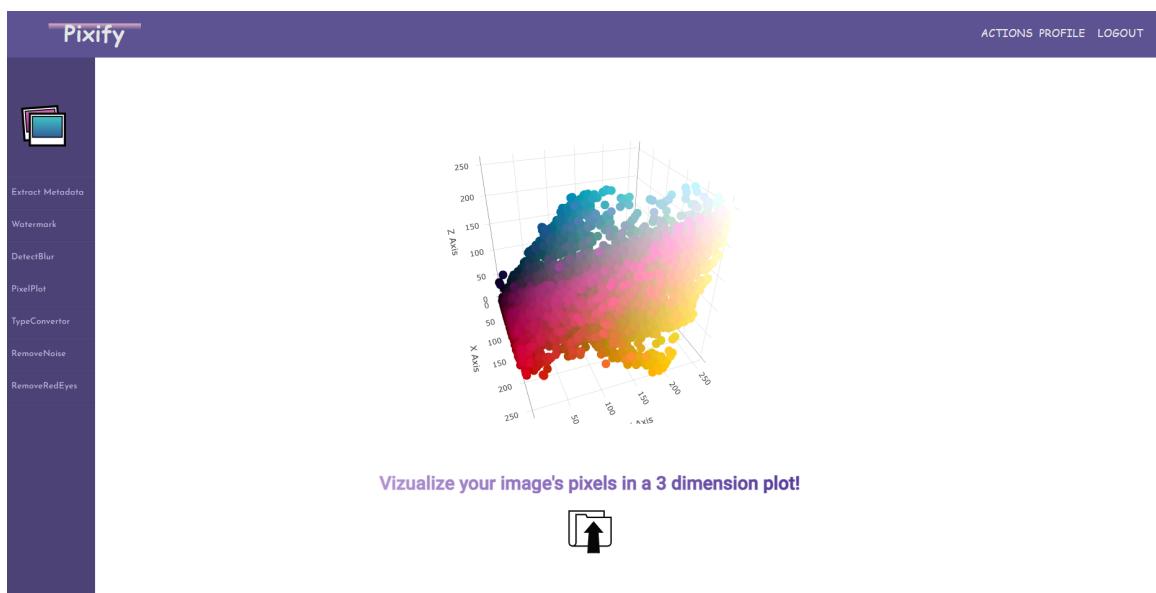


Figura 3.7: Graficul 3D



Figura 3.8: <https://renoguide.com.au/living-room/35-colourful-living-room-ideas-and-modern-designs>

Serverul are rolul de a parurge pixelii unei imagini, urmând ca fiecare să fie descompus în cele trei valori corespunzătoare unui pixel dintr-o poză (roșu, verde, albastru). Graficul oferă utilizatorilor posibilitatea de a mări, micșora și roti pixelii afișați, făcând explorarea și analiza datelor mai plăcută.

3.6 Reducerea noise-ului

Modificările aleatorii sau întreruperile nedoreite care împiedică calitatea și claritatea unei imagini sunt numite zgomot (noise) într-o imagine. Acest efect poate da impresia că valorile pixelilor fluctuează aleatoriu, producând zone neclare. Zgomotul aduce dificultăți în analiza unei imagini, interpretarea și, de asemenea, în estetica vizuală a ei. Zgomotul poate fi cauzat de mai mulți factori, cum ar fi: iluminare slabă, erori de compresie, dispozitivul cu care a fost făcută poză. Imaginile pot conține mai multe tipuri de zgomote, în continuare vom enumera și explica câteva dintre aceste tipuri [3].

3.6.1 Temporal Noise și Spatial Noise

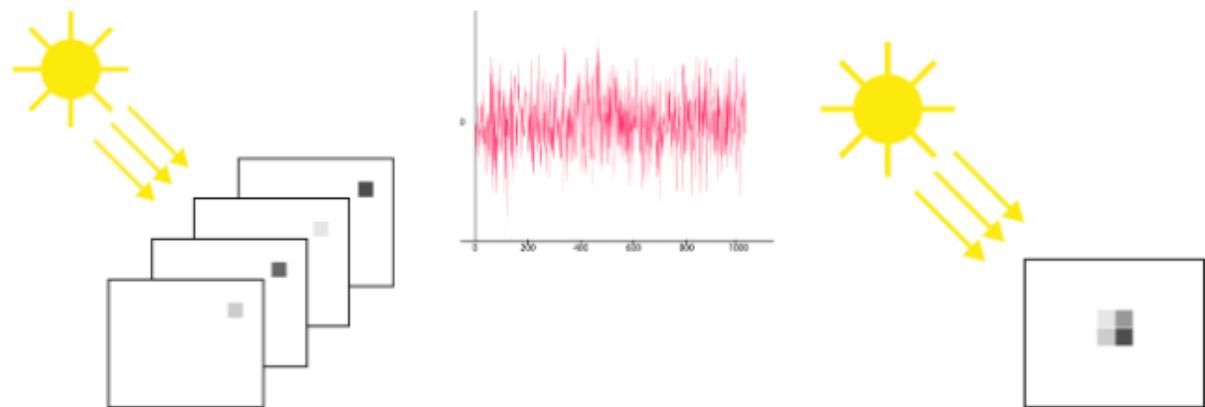


Figura 3.9: Tipuri de Zgomot

<https://www.image-engineering.de/library/image-quality/factors/>

1080-noise

Noise-ul temporal sau random este unul care variază independent de la o imagine la alta atunci când este creată. Aceasta rezultă din variațiile în generarea unei valori de la un singur pixel prin conversia fotonilor de intrare în electroni. De asemenea, numărul de fotoni care lovesc un singur pixel în timpul de expunere va varia. Această procedură este cunoscută sub denumirea de zgomot foton shot. Fotonii de intrare (particule de lumină) sunt transformați în electroni în timpul procesului de captare a imaginii în fotografie digitală (a se vedea Figura 3.9). Acești electroni sunt adunați în fiecare pixel de pe un senzor de imagine dintr-o cameră pentru a reprezenta cantitatea

de lumină prezentă în acea zonă particulară. Dacă ne uitam la un pixel dintr-o imagine care a fost facută de mai multe ori, vom observa că se modifica în timp, chiar dacă cadrul nu a fost modificat.

Zgomotul spațial al unei imagini poate fi definit drept o variație sau intrerupere nedorită care apare pe diversi pixeli. Acesta apare ca fluctuații la nivel de pixel într-o singură imagine, spre deosebire de zgomotul temporal, care fluctuează de la o imagine la alta. În funcție de circumstanțe, pot exista o varietate de cauze ale zgomotului spațial. De exemplu, poate fi creat de defecte ale senzorului de imagine din camerele digitale, cum ar fi pixeli fierbinți sau morți sau zgomotul de tipar cauzat de răspunsul neuniform al pixelilor.

În acest proces de reducere a noise-ului se verifică prima dată ce tip de imagine am primit de la utilizator doarece acest proces este diferit în cazul în care poza este alb negru sau în cazul în care este colorată (RGB).

Biblioteca OpenCV are două metode care sunt folosite pentru a elimina zgomotul din imagini, acestea sunt: cv2.fastNlMeansDenoising și cv2.fastNlMeansDenoisingColored. Aceste funcții performă eliminarea noise-ului din imagini folosind algoritmul "Non-local Means Denoising" [4]. Spre deosebire de filtrele "mean local", care iau valoarea medie a unui grup de pixeli care înconjoară un pixel ţintă pentru a netezi imaginea, filtrarea "non-local" ia o medie a tuturor pixelilor din imagine, ponderată în funcție de cât de asemănătoare sunt acești pixeli cu pixelul ţintă. După ce zgomotul a fost eliminat din imagine, aceasta poate avea în unele cazuri un aspect mai neclar, de aceea am aplicat asupra ei un proces de clarificare folosind un kernel [6].

Un kernel are mai multe usecase-uri când vine vorba de procesarea de imagini, cum ar fi: bluratul unei imagini, clarificarea unei imagini, în procesul de detectare a marginilor și multe altele. Un kernel mai este cunoscut și sub alte denumiri, de exemplu: matrice de conoluție, mască, matrice. În Figura 3.10 se pot observa mai multe tipuri de kernel-uri și efectul lor asupra unei imagini.

Când un nucleu este aplicat unei imagini, nucleul și imaginea sunt convective. Noua imagine este creată prin conoluție, o tehnică matematică care amestecă pixelii imaginii și valorile nucleului aferente. Nucleul [11] funcționează ca un filtru care are capacitatea de a evidenția sau de a ascunde anumite aspecte ale unei imagini. Nucleul este de obicei o matrice de dimensiuni mici care este adesea pătrată, cum ar fi 3x3 sau 5x5. Nucleul specifică zona din jurul fiecărui pixel din imagine care va fi luată în considerare în timpul procedurii de conoluție. Numerele din interiorul nucleului sunt

Operation	Kernel ω	Image result $g(x,y)$
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3×3 (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Figura 3.10: Kernel Effects

[https://en.wikipedia.org/wiki/Kernel_\(image_processing\)](https://en.wikipedia.org/wiki/Kernel_(image_processing))

ponderile sau coeficienții care controlează cât de mult afectează fiecare pixel învecinat pixelul final de ieșire. Procesul se repetă pentru fiecare pixel din imagine, inclusiv pentru pixelii de la margine.

În Figura 3.11 de mai jos este atașată o captură de ecran din aplicație unde se poate vedea realizarea procesului de reducerea noise-ului. Această opțiune se găsește în sidebar-ul din stânga sub denumirea "RemoveNoise".

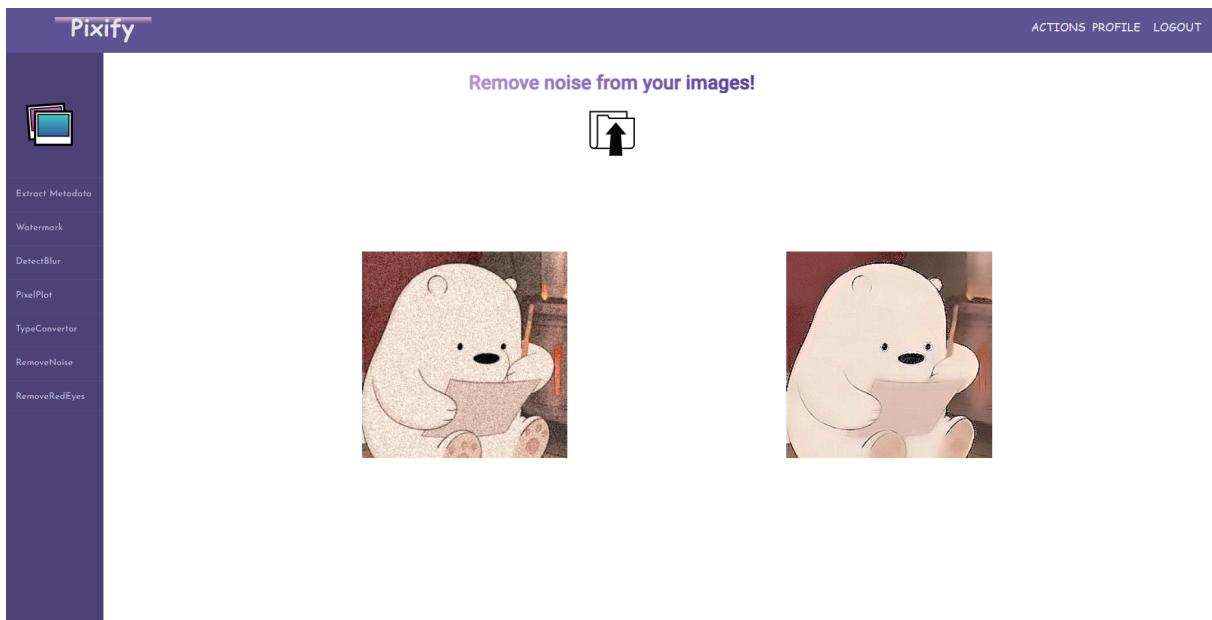


Figura 3.11: Removing Noise

3.7 Convertirea formatului

Diferite dispozitive, platforme sau aplicații software pot avea cerințe specifice sau suport pentru anumite formate de imagine. Un format compatibil asigură că o imagine poate fi afișată sau utilizată corect într-o anumită situație. Ca exemplu, un site web poate solicita fotografii în format JPEG, în timp ce un program de design grafic ar favoriza formatele PNG sau TIFF. Acesta poate fi doar unul dintre motivele pentru care un utilizator ar avea nevoie să schimbe formatul unei imagini.

Această secțiune a aplicației decurge în felul următor: după ce un utilizator s-a logat în aplicație acesta are acces la secțiunea cu procesări de imagini. Această parte este denumită în sidebar-ul disponibil utilizatorului din partea stângă a paginii sub numele de: "Type Convertor". Utilizatorul va trebui să introducă două informații relevante pentru convertirea formatului imaginii, formatul căreia apartine poza și formatul în care dorește să fie convertită. Opțiunile sunt urmatoarele: JPEG (Joint Photographic Experts Group), PNG (Portable Network Graphics), BMP (Bitmap), GIF (Graphics Interchange Format) și WEBP (Web Picture). Utilizatorul primește peste câteva secunde imaginea înapoi în formatul specificat pe care o poate salva.

Cele mai utilizate formate pentru imagini sunt JPG și PNG. JPEG este folosit pe scară largă pentru a stoca fotografii profesionale datorită abilității sale de a obține o



Figura 3.12: Convertirea formatului unei imagini

reducere semnificativă a dimensiunii fișierului prin compresie. Despre PNG putem aminti că este popular pentru compresia sa fară pierderi, făcându-l potrivit pentru grafica web, logo-uri și imagini cu margini ascuțite (imagini care au granițele dintre anumite obiecte clar și bine definite, fară nicio neclaritate).

3.8 Eliminarea ochilor roșii din imagini

Fotografiile în care persoanele au ochii roșii sunt o problemă estetică, dar care ar putea fi remediate prin ajustări ale camerei sau prin soluții software. Această problemă poate fi și o cauză a unor probleme medicale. De exemplu, dacă doar un ochi este roșu persistent în fotografii, pacientul poate avea strabism. Un indicator mai periculos al unei probleme oculare, cum ar fi o cataractă, dezlipirea retinei sau o infecție în interiorul ochiului, este o lumină albă sau galbenie într-un ochi. Această strălucire albă sau galbenie la tineri poate fi, de asemenea, un semn de avertizare al retinoblastomului, o malignitate pediatrică mai puțin frecventă, dar periculoasă.

Acest efect este cauzat de lumina care se reflectă pe retina din spatele ochilor. În general se întâmplă în condiții de lumină slabă când se folosește blițul. Lumina puternică trece rapid și se reflectă în spatele ochilor unde se află mult sânge de la retină. Aceasta este pe scurt motivul efectului de ochi roșii provocat de bliț.

Pentru ca un utilizator să poată elimina ochii roșii dintr-o poză trebuie să meargă la secțiunea "RemoveRedEyes" aflată la finalul sidebar-ului pus la dispoziție în partea stângă a paginii. Pentru a selecta o poză din dispozitivul personal trebuie să apese pe butonul din partea de sus a paginii urmând să se deschidă o fereastră de unde poate alege un fișier. Dacă fișierul ales nu are un format de imagine, acesta va primi un mesaj de eroare: "Only images are supported".

Primul pas în eliminarea ochilor roșii dintr-o imagine este detectarea ochilor prezenti în aceasta. Acest lucru a fost realizat cu ajutorul unui model deja antrenat oferit de librăria OpenCV de tip Haar. Un clasificator Haar sau un clasificator în cascadă Haar este un algoritm de detectare a obiectelor de învățare automată care identifică obiectele dintr-o imagine sau un videoclip. OpenCV oferă un repository public și gratis utilizatorilor ce conține mai multe fișiere de genul care pot fi folosite la detectarea fețelor, ochilor, nasului, gurii și a mașinilor. Aceste fișiere sunt de tipul XML, cel folosit în recunoașterea ochilor se numește "haarcascade_eye.xml".

În aplicația noastră vom utiliza funcția `cv2.CascadeClassifier("haarcascade_eye.xml")` pentru a încărca clasificatorul Haar, iar detecția ochilor din imagine va fi realizată cu ajutorul funcției `detectMultiScale()` având ca parametri imaginea și clasificatorul. Rezultatul obținut o listă de coordonate (x, y, w, h) ce reprezintă dreptunghiurilor găsite în jurul ochilor din imagine. Această listă de coordonate este apoi parcusă pentru a detecta zonele cu roșu intens din imagine și a forma o mască din ele. Pixelii sunt separați

în cele trei canele posibile (roșu, verde și albastru), iar masca este creată prin comparația canalului de culoare roșie cu un prag de 150 și cu suma pragurilor de verde și albastru adunate: $(r > 150) \& (r > bg)$. Am adăugat mai jos o secvență de cod relevantă celor explicate mai sus.

```
# For every detected eye
for (x, y, w, h) in eyes:
    eye = img[y:y + h, x:x + w]
    #3 channels
    b = eye[:, :, 0]
    g = eye[:, :, 1]
    r = eye[:, :, 2]
    bg = cv2.add(b, g)
    #euristica
    mask = (r > 150) & (r > bg)
    mask = mask.astype(np.uint8) * 255
    # Fill holes
    mask = fillHoles(mask)
```

Aceastăuristică nu se potrivește pentru toți pixelii dintr-un ochi, de aceea după această etapă trebuie umplute găurile lipsă din mască. Vom face acest lucru prin intermediul unei funcții "fillHoles()" adăugată mai jos care crează o copie a măștii noastre și inversează valorile acesteia, urmând ca între masca originală și copie să fie realizată operația de dijuncție pe biți.

```
def fillHoles(mask):
    maskFloodfill = mask.copy()
    h, w = maskFloodfill.shape[:2]
    maskTemp = np.zeros((h + 2, w + 2), np.uint8)
    cv2.floodFill(maskFloodfill, maskTemp, (0, 0), 255)
    mask2 = cv2.bitwise_not(maskFloodfill)
    return mask2 | mask
```

Apoi este calculat un nou canal de culoare pentru a fi înlocuit în zona ochiului roșu din imagine. Noul canal va fi media canalelor de verde și de albastru, care crează un aspect mai spre gri/negru pe regiunea ochiului din poză. În Figura 3.13 de mai

jos se poate observa rezultatul realizării acestor calcule pe o poză în care persoana are ochii roșii din cauza blițului.

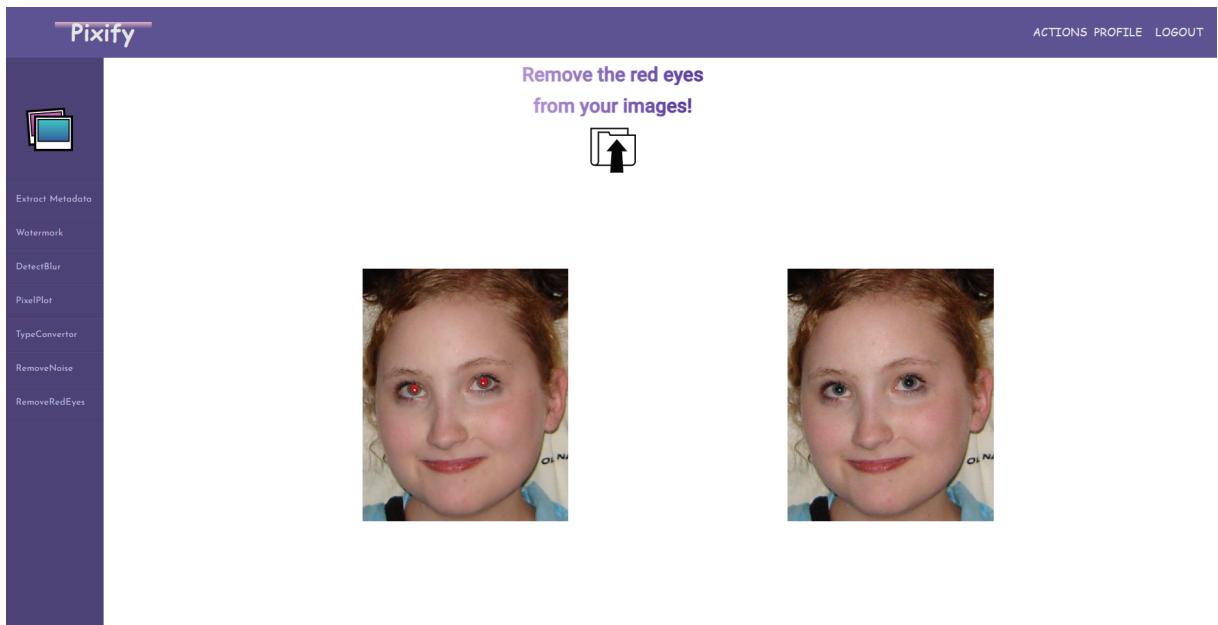


Figura 3.13: Eliminarea ochilor roșii

3.9 Login și Register

Login-ul și registerul fac parte din funcționalitățile aplicației, chiar sunt elemente importante deoarece ajută la securitatea platformei. Ele oferă mai întâi o metodă de autentificare și securitate a utilizatorilor. Aplicațiile pot confirma identitatea persoanelor care folosesc sistemul solicitându-le să se înregistreze și să creeze acreditări speciale de conectare, cum ar fi nume de utilizator și parole. Această procedură de autentificare protejează programul și utilizatorii săi împotriva accesului nedorit sau a utilizării greșite, asigurându-se că numai utilizatorii autorizați pot accesa informații sensibile și pot îndeplini anumite sarcini.

Informațiile pe care un utilizator trebuie să le dezvăluie pentru a-și crea un cont sunt următoarele: numele complet, un nume de utilizator, adresa de email, număr de telefon, o parolă și să confirme parola aleasă. Numele de utilizator trebuie să fie unic, de aceea în cazul în care introduce un nume de utilizator deja folosit de un cont existent va primi o eroare sub câmpul destinat numelui de utilizator. De asemenea toate câmpurile sunt obligatorii, utilizatorul va primi o avertizare în cazul în care nu

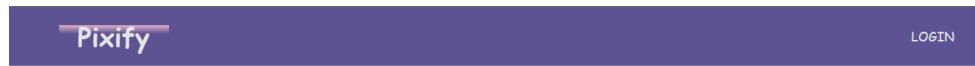


Figura 3.14: Register

le completează pe toate. Parolele trebuie să respecte un anumit format, mai exact o lungime de minim 8 caractere și să conțină cel puțin o cifră, o majusculă și un caracter special. Cu cât este o parolă mai complexă cu atât este mai greu pentru persoanele neautorizate să o ghicească sau să o descifreze.

Un utilizator își poate edita informațiile personale mergând la secțiunea "Profile" (a se vedea Figura 3.15) din header-ul aplicației. Acolo vor fi afișate datele pe care le-a introdus atunci când și-a creat contul. Pe lângă datele deja publicate poate introduce și unele noi, cum ar fi: adresa, data de naștere și gen. El poate modifica aceste date dacă cunoaște parola contului, aceasta fiind o metodă de a asigura autenticitatea persoanei care folosește contul.

Pixify

mneuclea's profile



Username mneuclea

Email m.neculea@yahoo.com

Old Password Old Password

Password Password

Confirm Password Confirm Password

Phone Number 0744205983

Address Judetul Neamt, Piatra-Neamt

Birthdate 06/09/2023

Gender Female

Save Profile

ACTIONS **LOGOUT**

Figura 3.15: Profil

Concluzii

Pentru acest proiect de licență am ales să aplicăm aspecte legate de domeniul procesării imaginilor în crearea unei platforme de editare a lor. Procesările incluse în aplicație țin de aspectul unei imagini și de securitate, cum ar fi punerea unui watermark pe o poză sau extragerea metadatelor. Aplicația a fost implementată cu ajutorul framework-ului Angular care aduce inovații legate de reutilizabilitatea și testabilitatea codului, iar pentru server a fost folosit microframework-ul Flask.

Având în vedere cele prezentate mai sus, crearea aplicației "Pixify" pentru procesare de imagini s-a desăvârșit cu succes. Pentru a duce la bun sfârșit această implementare a platformei am folosit cunoștințele teoretice și practice dobândite pe parcursul facultății, dar și din studiul individual.

Consider că această aplicație ar putea fi dezvoltată în continuare, devenind mult mai complexă. În primul rând, secțiunea cu procesări de imagini ar putea fi îmbunătățită prin adăugarea mai multor unelte de editare, câteva idei în acest sens ar putea fi: detectarea scrisului din imagini, adăugarea de filtre, detectarea diverselor obiecte, generarea unei scene din mai multe poze care surprind diverse cadre. De asemenea, toate aceste efecte pot fi extinse pentru a funcționa și pe fisiere de tip video.

În al doilea rând, încă o îmbunătățire ar putea fi adăugarea unor galerii unde un utilizator și-ar putea salva pozele editate și opțiunea de a le distribui mai departe pe social-media.

În concluzie, consider că lucrarea și-a atins scopul propus la început de proiect.

Bibliografie

- [1] Michael T. Heideman, Don H. Johnson, C. Sidney Burrus, *Gauss and the History of the Fast Fourier Transform*, IEEE ASSP MAGAZINE, 1984.
- [2] Travis E. Oliphant PhD, *Guide to NumPy*, 2006.
- [3] Rohit Verma, Dr. Jahid Ali, *A Comparative Study of Various Types of Image Noise and Efficient Noise Removal Techniques*, Volume 3, Issue 10, 2013.
- [4] Antoni Buades, Bartomeu Coll, Jean-Michel Morel, *Non-Local Means Denoising*, IPOL Journal, 2011.
- [5] Maria Petrou, Costas Petrou, *Image Processing: The Fundamentals*, Wiley, 2010.
- [6] Sandipan Dey, *Hands-On Image Processing with Python*, Packt, 2018.
- [7] C.I. Podilchuk, E.J. Delp, *Digital watermarking: algorithms and applications*, IEEE Signal Processing Magazine, 2001.
- [8] Fast Fourier Transform (FFT) Background.

[https://www.13harrisgeospatial.com/docs/
BackgroundFastFourierTransform.html](https://www.13harrisgeospatial.com/docs/BackgroundFastFourierTransform.html)

- [9] Fourier Transform.

<https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm>

- [10] How to Use One-to-Many Database Relationships with Flask-SQLAlchemy.

[https://www.digitalocean.com/community/tutorials/
how-to-use-one-to-many-database-relationships-with-flask-sqlalchemy](https://www.digitalocean.com/community/tutorials/how-to-use-one-to-many-database-relationships-with-flask-sqlalchemy)

- [11] Image Processing with Python: Image Effects using Convolutional Filters and Kernels

<https://medium.com/swlh/image-processing-with-python-convolutional->