Nov 23, 2026 · @Patrick

Picasa a disparu en 2016, et des années de retouches et de classement dormaient dans ses fichiers. Voici comment j'ai construit, avec Claude, une photothèque qui les reprend et y ajoute quelques fonctionnalités.

Pourquoi refaire un Picasa en 2026

Picasa s'est arrêté en 2016, mais mes photos, elles, sont restées. Pendant des années, j'y ai recadré, redressé, classé dans des albums, nommé des visages. Tout cela dort dans de petits fichiers cachés, les `.picasa.ini`, un par dossier.

Changer d'outil voulait dire perdre ce travail, ou le refaire. Les alternatives libres, comme digiKam ou Shotwell, sont solides, mais elles ne reprennent au mieux qu'une partie de ces directives. Et je cherchais un outil qui combine ce qui me plaîsait dans Picasa (la simplicité, les retouches qui ne touchent pas l'original) avec ce que les services en ligne apportent aujourd'hui : reconnaissance des visages, tags automatiques, recherche en langage naturel.

J'ai donc conçu ma propre photothèque, pour Windows et Ubuntu, et je l'ai fait développer par Claude, l'IA d'Anthropic. Cet article raconte les choix techniques, les difficultés rencontrées et ce que j'ai appris de cette façon de travailler.

Le cahier des charges

Avant la première ligne de code, j'ai fixé neuf règles. Elles ont guidé tous les arbitrages ensuite.

1. Naviguer par dossier, mais aussi par date, comme dans Shotwell.
2. Créer des albums, c’est-à-dire des dossiers virtuels.
3. Poser des tags et naviguer par tag.
4. Offrir des services de retouche, dont la correction des lignes fuyantes.
5. Ne jamais modifier une photo : les directives sont gardées dans des fichiers à part et l’image est calculée à la volée.
6. Réutiliser au maximum les directives des fichiers `.picasa.ini`.
7. Récupérer les albums définis dans Picasa.
8. Une interface en français et en anglais.
9. Tirer le meilleur parti de la reconnaissance d’image et du tagging automatique.

La règle 5 est la plus structurante. Elle impose que chaque affichage soit un calcul, ce qui pèse sur toute l’architecture et sur les performances.

Architecture : d’Oracle à Python et SQLite

L’architecture finale tient en un seul processus Python, une base SQLite et une interface web locale. Ce n’était pas mon premier choix.

Je travaille au quotidien avec Oracle, et j’avais d’abord imaginé une base Oracle, une API REST avec ORDS, le traitement d’images en Node.js et une interface Bootstrap. C’est robuste, mais lourd pour une application personnelle : une base à installer et à administrer, deux langages côté serveur, et un Node.js mal outillé pour l’analyse d’image.

La version retenue :

Couche	Choix	Pourquoi

Serveur	Python, FastAPI, uvicorn	Un seul langage, et l'écosystème de référence pour l'image et l'IA
Base	SQLite en mode WAL	Un simple fichier, rien à installer, lectures pendant les écritures
Image	Pillow, OpenCV, NumPy	Lecture des formats, géométrie, filtres
IA	ONNX Runtime	Modèles exécutés en local, sans GPU obligatoire
Vidéo	ffmpeg	Lecture, image de miniature, conversion
Interface	Bootstrap 5, JavaScript sans framework	Légère, en français et en anglais, thème sombre

L'application se lance avec `phototheque serve` et s'ouvre dans le navigateur. Les données propres à l'application (base, caches, modèles d'IA, réglages) vivent dans un dossier `.phototheque` du profil utilisateur. Les photos restent là où elles sont.

Les directives de retouche, les tags et les légendes sont écrits dans un fichier `.phototheque.json` par dossier, à côté des photos, sur le modèle du `.picasa.ini`. La base SQLite n'est qu'un index : on peut la supprimer et tout reconstruire en relisant les dossiers.

Retouches non destructives et reprise de Picasa

Aucune photo n'est jamais réécrite : chaque retouche est une directive, et l'image affichée est recalculée à partir de l'original. Une photo porte une liste d'opérations appliquées dans l'ordre : rotation, redressement, recadrage, perspective, amélioration, noir et blanc, sépia, netteté. Annuler une retouche revient à retirer une ligne.

[embed: node/dbd6b557-5f88]

Les directives de la photothèque et celles de Picasa s'appliquent à l'original pour produire l'image affichée, qui est gardée en cache.

Recalculer à chaque affichage coûterait trop cher. Les rendus sont donc mis en cache en trois tailles (miniature, aperçu, plein écran), avec une purge automatique quand le cache dépasse sa taille maximale. Pour un JPEG, le décodage se fait directement en taille réduite, ce qui divise le temps de lecture.

Relire les .picasa.ini

Le format des `.picasa.ini` n'a jamais été documenté officiellement. Il a fallu le décoder, puis le vérifier sur ma vraie photothèque :

- Recadrage : la directive `crop64` code les quatre bords du cadre en hexadécimal, chacun sur 16 bits, en fraction de la largeur ou de la hauteur.
- Redressement : la directive `tilt` donne une valeur brute qu'il faut convertir en degrés. Le signe et le facteur se sont révélés inversés par rapport à ma première interprétation : un −3° dans Picasa doit devenir +3°. Un outil de calibration compare les fichiers que Picasa avait sauvegardés dans `.picasaoriginals` pour retrouver le bon facteur.
- Rotation : `rotate(n)` tourne de n quarts de tour, là aussi dans le sens inverse de ce que j'avais supposé.
- Noms de fichiers : Picasa ne respecte pas toujours la casse ni la normalisation des accents. La correspondance se fait donc sans tenir compte de la casse et après normalisation Unicode.
- Filtres : les filtres automatiques de Picasa (J'ai de la chance, Contraste auto, Couleur auto, Réchauffer) sont remplacés par le nouveau bouton Améliorer à l'intensité par défaut. La Netteté, le noir et blanc et le sépia

sont repris tels quels.
- Albums, favoris, légendes, visages : tout est récupéré, y compris les noms donnés aux personnes (les contacts de Picasa), qui servent ensuite de point de départ à la reconnaissance des visages.

Dans la fiche de propriétés, une petite pastille ronde indique qu'une photo porte des directives Picasa. Au survol, elle en donne le détail.

Lignes fuyantes et bouton Améliorer

Redresser un bâtiment sans l'amincir

Je photographie beaucoup d'architecture, et les façades prises d'en bas ont des verticales qui convergent. Les outils grand public corrigent cela en déformant l'image en trapèze. Les verticales redeviennent droites, mais le bâtiment paraît plus mince qu'en réalité.

La photothèque procède autrement, en trois temps :

1. Détecter les segments de droite de l'image (détecteur LSD d'OpenCV) et garder ceux qui sont presque verticaux.
2. Estimer leur point de fuite par un vote robuste (RANSAC), qui ignore les segments parasites.
3. Faire pivoter un appareil virtuel plutôt que déformer l'image. La correction est une homographie de rotation, calculée à partir de la focale lue dans les données EXIF. Les proportions du bâtiment sont conservées, comme avec un objectif à décentrement.

Un réglage permet de ne corriger qu'en partie : une légère convergence

résiduelle paraît souvent plus naturelle. Et la correction peut être appliquée automatiquement à toutes les photos qui portent certains tags, comme « architecture ».

Un équivalent du bouton Améliorer de Google Photos

Le bouton Améliorer analyse l'image et enchaîne cinq corrections :

- la balance des blancs, par l'hypothèse du monde gris ;
- les niveaux, en recalant le noir et le blanc ;
- un gamma pour éclaircir ou assombrir les tons moyens ;
- un contraste local adaptatif (CLAHE) sur la luminance seule, pour ne pas dénaturer les couleurs ;
- une vibrance qui ravive les couleurs ternes sans saturer celles qui le sont déjà, puis une légère netteté.

L'intensité par défaut se règle dans les paramètres. Comme le reste, l'amélioration n'est qu'une directive : un second clic la retire.

L'IA, en local d'abord

La reconnaissance d'image tourne sur mon ordinateur : aucune photo ne part dans le cloud, sauf pour l'identification des plantes, que je déclenche moi-même. Les modèles sont téléchargés une fois, puis exécutés par ONNX Runtime sur le processeur. Le traitement se fait en tâche de fond, des photos les plus récentes aux plus anciennes, sans bloquer la navigation.

Tags automatiques et recherche en langage naturel

Le modèle CLIP d'OpenAI, dans une version allégée, place images et textes dans le même espace mathématique. Cela donne deux usages :

- Tags automatiques : chaque photo est comparée à un vocabulaire d'environ 150 libellés en français et en anglais (plage, montagne, fleur, église, chien…). Ceux qui dépassent un seuil deviennent des tags, distingués des tags posés à la main.
- Recherche sémantique : on tape « coucher de soleil sur la mer » et la photothèque classe les photos par ressemblance avec la phrase, même si aucune ne porte ces mots. Le même mécanisme trouve les photos semblables à une photo donnée.

Les visages

Les modèles d'InsightFace détectent les visages (SCRFD), puis calculent pour chacun une empreinte numérique (ArcFace). Les visages dont les empreintes sont proches sont regroupés. Il suffit de nommer un groupe pour que le nom s'applique à tous ses visages, et aux nouveaux qui lui ressemblent. Les noms récupérés de Picasa amorcent ces groupes : une personne déjà nommée dans Picasa l'est dès la première analyse.

Les plantes, avec Pl@ntNet

Pour les plantes, j'utilise l'API de Pl@ntNet, le service d'identification botanique porté par des instituts de recherche français. Seules les photos qui portent le tag fleur, plante ou arbre et qui n'ont pas encore de légende sont envoyées. Le nom trouvé devient leur légende, sous la forme « Églantier (Rosa canina) »,

mais seulement si la confiance dépasse un seuil réglable.

Le plan gratuit autorise 500 identifications par jour. Un compteur local, recalé sur ce que déclare l'API, arrête le traitement au plafond, et une photo déjà envoyée ne l'est jamais deux fois. Il suffit de relancer le lendemain pour la suite.

Tenir 200 000 photos

Les premiers essais sur 65 000 photos sous Ubuntu ont révélé des lenteurs invisibles sur un petit jeu de test. Chacune avait une cause précise, et aucune ne venait de SQLite lui-même.

Symptôme constaté	Cause	Remède
1 à 2 s pour passer d'une photo à la suivante	Chaque JPEG décodé en pleine résolution	Décodage directement en taille réduite, préchargement des photos voisines
Arborescence des dossiers affichée au bout de 5 minutes	La surveillance des dossiers bloquait le démarrage, et les lectures attendaient la fin des écritures	Surveillance lancée en tâche de fond, connexions de lecture séparées de l'unique connexion d'écriture
Visionneuse vide au clic sur une miniature	Une requête de mise en file des tâches d'IA tenait le verrou d'écriture plus de 10 minutes	Index adapté et requête réécrite : 0,26 s
Nouveaux visages jamais détectés	Les tâches les plus anciennes passaient toujours en premier	Alternance des tâches, photos récentes d'abord, détection des ajouts de fichiers

Pour viser 200 000 photos, j'ai aussi fait mesurer chaque opération sur une bibliothèque de test générée à cette taille :

- Rescan : 5,3 s ramenées à 0,9 s, en sautant les dossiers dont la signature n’a pas changé.
- Regroupement des visages : de 107 à 1 300 visages par seconde, avec les centres des groupes gardés en mémoire.
- Vecteurs CLIP : stockés en demi-précision, de 410 à 205 Mo en mémoire, pour une recherche en 0,2 s.
- Tag sur 200 000 photos sélectionnées : 14 s. La sélection est un filtre traité côté serveur, pas une liste de 200 000 identifiants.
- Grille de miniatures : virtualisée, elle ne garde qu’une soixantaine de vignettes dans la page, quelle que soit la taille du dossier.

La leçon : il faut des données réelles, ou au moins à l’échelle réelle, et un journal qui chronomètre les requêtes lentes. C’est ce journal qui a permis de trouver les causes en quelques allers-retours.

Vidéos, HEIC et RAW

Une photothèque de famille contient aussi des vidéos : surtout des MP4 récents dans mon cas, et quelques vieux AVI. Elles sont indexées comme les photos, avec leur date de tournage et leur durée, et rangées dans les mêmes dossiers, dates, albums et tags.

- Miniature : une image prise à 10 % de la vidéo par ffmpeg, avec un badge pellicule et la durée.
- Lecture : un MP4 en H.264 est lu tel quel par le navigateur. Un vieil AVI (MJPEG, DivX, Xvid) est converti à la volée en MP4 720p au premier visionnage, et la copie est gardée en cache. L’original n’est jamais touché.
- Rotation : celle enregistrée par le téléphone est respectée, et on peut

ajouter un quart de tour, seule retouche proposée pour une vidéo.
- Diaporama : une vidéo est lue en entier, puis le diaporama passe à la suite.
- IA : les tags automatiques et les visages sont calculés sur l'image de la miniature.

Les photos HEIC des iPhone et les fichiers RAW des reflex et hybrides (CR2, NEF, ARW, DNG…) sont traités comme des JPEG. Pour les RAW, l'aperçu JPEG intégré au fichier sert aux miniatures, et le développement complet n'est fait que pour le zoom et l'export.

Développer avec Claude : retour d'expérience

Je n'ai pas écrit le code de cette application : je l'ai conçue, testée et pilotée, et Claude l'a écrite. La répartition des rôles a été nette dès le départ.

Mon rôle : fixer le besoin, arbitrer les choix (l'abandon d'Oracle, par exemple), installer chaque version sous Windows et sous Ubuntu, l'essayer sur ma vraie photothèque et rapporter précisément ce qui n'allait pas, avec le journal d'erreurs quand il y en avait un.

Le rôle de Claude : proposer et justifier l'architecture, écrire le code, mais aussi les tests automatiques. Une bibliothèque de test générée, avec de faux `.picasa.ini`, sert à vérifier à chaque version que rien ne régresse. Les services externes sont simulés : un faux serveur Pl@ntNet permet de tester le quota sans consommer un seul appel réel.

Le projet a avancé par petites versions, de la 0.1 à la 0.9, chacune livrée sous forme d'archive à décompresser par-dessus la précédente.

Ce que je retiens :

- Le vrai terrain reste irremplaçable. Le sens du redressement Picasa, les lenteurs à 65 000 photos, les certificats HTTPS de Python sous Windows : rien de tout cela ne se voyait sur les données de test. Il a fallu mes essais pour le révéler.
- Un bon rapport d'anomalie vaut beaucoup. Une ligne de journal ou un symptôme précis (« la photo précédente apparaît une fraction de seconde ») suffisait pour obtenir une correction ciblée et un test qui l'empêche de revenir.
- La conception reste humaine. L'IA va très vite sur le code, mais c'est à moi de dire ce que je veux, dans quel ordre, et ce qui est assez bien.
- L'IA sait aussi dire non. Quand j'ai demandé l'icône officielle de Picasa, Claude a refusé de la reproduire, puisqu'il s'agit d'une marque, et a proposé un pictogramme neutre.

Et maintenant

Mes années de retouches et d'albums Picasa ont survécu au changement d'outil. Les photos n'ont jamais été modifiées : si l'application disparaissait demain, il resterait les originaux et des fichiers de directives lisibles.

J'ai déjà noté les prochaines évolutions, en commençant par la gestion des visages : renommer les groupes en « personnes non identifiées », montrer les visages directement sur la photo, et pouvoir retirer un visage d'un groupe ou le déplacer vers un autre.

Author

- 

  Patrick

  GPM Factory