Rapport d’analyse

POC OUTIL LABELLISATION IA

Une image contenant texte, art, Dessin d’enfant, Graphique

Description générée automatiquement

Commanditaire : Nicolas David

Etudiants :

* Frédéric Ye
* Hannick Habdul-Kuthoos
* Mathéo Maréchal
* Vittorio Toffolutti

Sommaire

INTRODUCTION ………………………………………………3

OBJECTIF ………………………………………………………..4

Reformulation du besoin …………………………………..4

Contraintes …………………………………………………….6

Les acteurs ……………………………………………………..7

Description de l'application et Glossaire ……………..7

ANALYSE FONCTIONNELLE ……………………………..11

Base de données et entités de l’application ……….11

Design et chaine de traitement de l’application …..13

ETUDE TECHNIQUE ………………………………………..17

L’architecture du projet ……………………………………17

Choix des technologies …………………………………..17

REALISATION DU PROJET ………………………………..20

Outils et technologies pour le développement collaboratif ……………………………………………………20

Télétravail et suivie de projet ……………………………20

CONCLUSION ……………………………………………….21

**Introduction**

Dans un contexte de transformation numérique et de développement des technologies de l'information géographique, l'Institut Géographique National (IGN) se positionne comme un acteur clé dans la collecte, la gestion et la diffusion de données géospatiales en France. Historiquement reconnu pour son expertise dans la cartographie et la production d'images aériennes, l'IGN s'adapte aux évolutions de son secteur en intégrant de nouvelles sources de données et en adoptant des approches innovantes pour leur exploitation.

L'une des évolutions majeures observées ces dernières années est le passage d'une approche centrée sur les données statiques à une approche dynamique et orientée vers l'analyse en temps réel. Cette transition s'accompagne d'une demande croissante pour des outils et des plateformes permettant la collecte, la gestion et l'analyse de données géographiques de manière efficace et collaborative. Dans ce contexte, l'IGN reconnaît l'importance de développer des solutions technologiques adaptées pour répondre aux besoins diversifiés de ses utilisateurs, qu'il s'agisse de professionnels du secteur géomatique, de chercheurs ou du grand public.

Parallèlement à ces évolutions technologiques, l'IGN s'engage également dans une démarche d'ouverture et de partage de ses données, conformément aux principes de l'open data. Cette politique vise à favoriser l'accès libre et gratuit aux données géographiques produites par l'institution, tout en encourageant la collaboration et l'innovation au sein de la communauté géomatique. Dans cette optique, la création d'une interface web de labellisation de données géographiques revêt une importance stratégique, car elle permettra non seulement de faciliter la collecte et l'annotation de données pour les besoins internes de l'IGN, mais aussi de créer un écosystème ouvert où les utilisateurs externes pourront contribuer à l'enrichissement des jeux de données disponibles.

C'est dans ce contexte dynamique et en constante évolution que s'inscrit notre projet. En développant une interface web de labellisation d'images segmentées, notre objectif est de fournir à l'IGN un outil robuste et fonctionnel pour l'annotation de données géographiques, tout en favorisant l'ouverture et la collaboration avec la communauté géomatique. Une première version a déjà été réalisé sous QGIS où la labellisation d’image est possible mais le besoin d’une interface web open-source pouvant être déployé et où l’ajout de fonctionnalité est possible restait indispensable à l’IGN.

Ce projet de développement d'une interface web de labellisation d'images segmentées est conçu comme un Proof of Concept (POC) pour l'Institut Géographique National (IGN). L'objectif de ce POC est double : d'une part, il vise à démontrer à la direction de l'IGN la viabilité technique et l'utilité opérationnelle de l'outil proposé; d'autre part, il sert à affiner le projet, c'est-à-dire à identifier et à résoudre les principaux défis techniques et organisationnels avant le déploiement à plus grande échelle. Une partie du code a été réalisé en interne par des étudiant de l’école EPITA. Ce code est essentiel pour la chaine de traitement de l’application.

En mettant en place ce POC, l'IGN pourra évaluer de manière concrète les avantages de l'interface en termes de facilitation de la collecte et de l'annotation précise des données géospatiales. Ce test initial permettra également de recueillir des retours d'utilisateurs internes et externes, cruciaux pour ajuster et optimiser l'outil en fonction des besoins réels et des conditions d'usage spécifiques. L'approche progressive adoptée ici minimise les risques et permet à l'IGN de procéder par étapes, en affinant progressivement le projet pour assurer sa mise en place et son intégration.

En ce qui concerne les aspects financiers, le projet de développement de l'interface web de labellisation d'images segmentées est développé entièrement en open-source. Actuellement, le projet est mené par une équipe de quatre étudiants à temps plein pendant une période de quatre semaines pour la création d'un prototype non déployé. Cette équipe sera chargée d'effectuer les ajustements nécessaires tout au long du processus de développement, ainsi que d'incorporer les fonctionnalités supplémentaires demandées par le commanditaire.

Une potentielle estimation du travail supplémentaire serai d'une période d'environ un mois pour finaliser le développement, effectuer des test de sécurité et déployer l'interface à grande échelle. Cette estimation prend en compte les ajustements probables, les itérations, les tests rigoureux requis pour garantir la qualité et la fiabilité de l'outil ainsi que le déploiement.

**Objectif**

1. Reformulation du besoin

L'objectif de cette étude est de concevoir une interface web destinée à la labellisation de zones géographiques, qui serviront ultérieurement de jeux de test pour l'IGN. Cette application se caractérise par plusieurs fonctionnalités majeures. Tout d'abord, elle permet la création de chantiers à partir d'un catalogue d'images. Un chantier représente une sélection d'images dans un catalogue préalablement renseigné, offrant ainsi une flexibilité dans la gestion des données.

Une image contenant texte, diagramme, Police, conception

Description générée automatiquement

Représentation d’une sélection d’image pour un chantier

La deuxième fonctionnalité majeure est la segmentation de l'image en zones regroupées par identification de pixels. Cette étape, réalisée par les étudiants de l'école EPITA, est cruciale pour délimiter les différentes zones d'intérêt à étiqueter.

Une image contenant mammifère, Fauves, tigre, faune

Description générée automatiquement

Représentation d’une segmentation (ai.standford.eu)

Ensuite, l'application assure le découpage de chaque image en patchs à la volée, ce qui permet une meilleure fluidité pour l'utilisateur tout en garantissant une précision accrue dans la labellisation. Ces patchs correspondent à des zones plus petites de l'image, facilitant ainsi le processus de travail.

Une image contenant ligne, diagramme, Rectangle, carré

Description générée automatiquement

Représentation des patchs d’une image

Un autre aspect essentiel de l'application est la possibilité pour l'utilisateur de définir et de charger des nomenclatures pour labelliser les patchs. Cette fonctionnalité permet une adaptabilité aux différentes situations et besoins, offrant ainsi une grande souplesse dans le processus de labellisation. De plus, l'application assure l'enregistrement et la gestion efficace des labellisations résultantes, garantissant ainsi la traçabilité et la cohérence des données annotées.

Une image contenant texte, capture d’écran, carte, diagramme

Description générée automatiquement

Exemple d’un patch avec sa nomenclature qui est enregistré

Des fonctionnalités bonus ont également été évoquées, visant à enrichir davantage l'expérience utilisateur et à améliorer l'efficacité du processus de labellisation. Parmi celles-ci, l'implémentation de différents rôles au sein de la chaîne de traitement de l'image a été envisagée. Ces rôles incluraient un gestionnaire chargé de créer le chantier avec toutes les informations nécessaires à sa réalisation, telles que la nomenclature, les tailles de patchs, ainsi que les autres paramètres requis. Une autre personne serait spécifiquement dédiée à la labellisation des images, tandis qu'une troisième personne serait responsable de l'approbation ou du rejet du travail réalisé.

Par ailleurs, une interface de double vue a été proposée pour permettre la labellisation d'une image en la comparant à une autre. Cette fonctionnalité s'avère particulièrement utile pour l'annotation des changements entre deux millésimes orthophotographiques. En offrant la possibilité de visualiser et de comparer deux images simultanément, cette interface facilite l'identification des modifications et des évolutions sur le territoire au fil du temps, renforçant ainsi la qualité et la pertinence des données géographiques annotées.

En résumé, cette interface web vise à offrir une expérience utilisateur intuitive et efficace pour la labellisation des données géographiques, tout en intégrant des fonctionnalités avancées pour répondre aux besoins spécifiques de l'IGN et de ses utilisateurs.

1. Contraintes

Les contraintes imposées par le commanditaire ont joué un rôle déterminant dans la conception et le développement de cette solution. Tout d'abord, il était impératif que l'application soit une solution web et open-source, répondant ainsi aux normes d'accessibilité et de transparence prônées par l'IGN. Pour atteindre cet objectif, une des premières approche consistait à reprendre le code des étudiants de l'EPITA et à le comparer avec des solutions déjà existantes, afin de maximiser le temps de calcul et d'optimiser la fluidité de l'application pour l'utilisateur.

Par ailleurs, il était essentiel que la solution soit réalisée en utilisant des technologies récentes, tant pour le front-end que pour le back-end. Cela garantit non seulement la pérennité de l'application, mais aussi sa compatibilité avec les standards actuels du développement web. Ainsi, l'utilisation de librairies récentes a été privilégiée afin de tirer parti des dernières avancées technologiques et d'offrir une expérience utilisateur optimale.

Un aspect crucial des contraintes imposées par le commanditaire est la maximisation des calculs côté client. En effet, il a été spécifié que le moins de calcul et de traitement lié à l'application devait être effectué côté serveur, afin de garantir une plus grande autonomie pour les serveurs de l'IGN. Cette approche permet de décharger les ressources serveur et d'optimiser les performances globales de l'application.

Le commanditaire avait des préférences et nous orientait sur les technologies qu’il considérait être les meilleurs et nous décidions derrière en comparant avec l’existant. Ces contraintes ont guidé le processus de développement de l'interface web de labellisation d'images segmentées, dictant les choix technologiques et architecturaux afin de répondre au mieux aux besoins et aux exigences du commanditaire tout en respectant les standards de qualité et de performance attendus.

1. Les acteurs

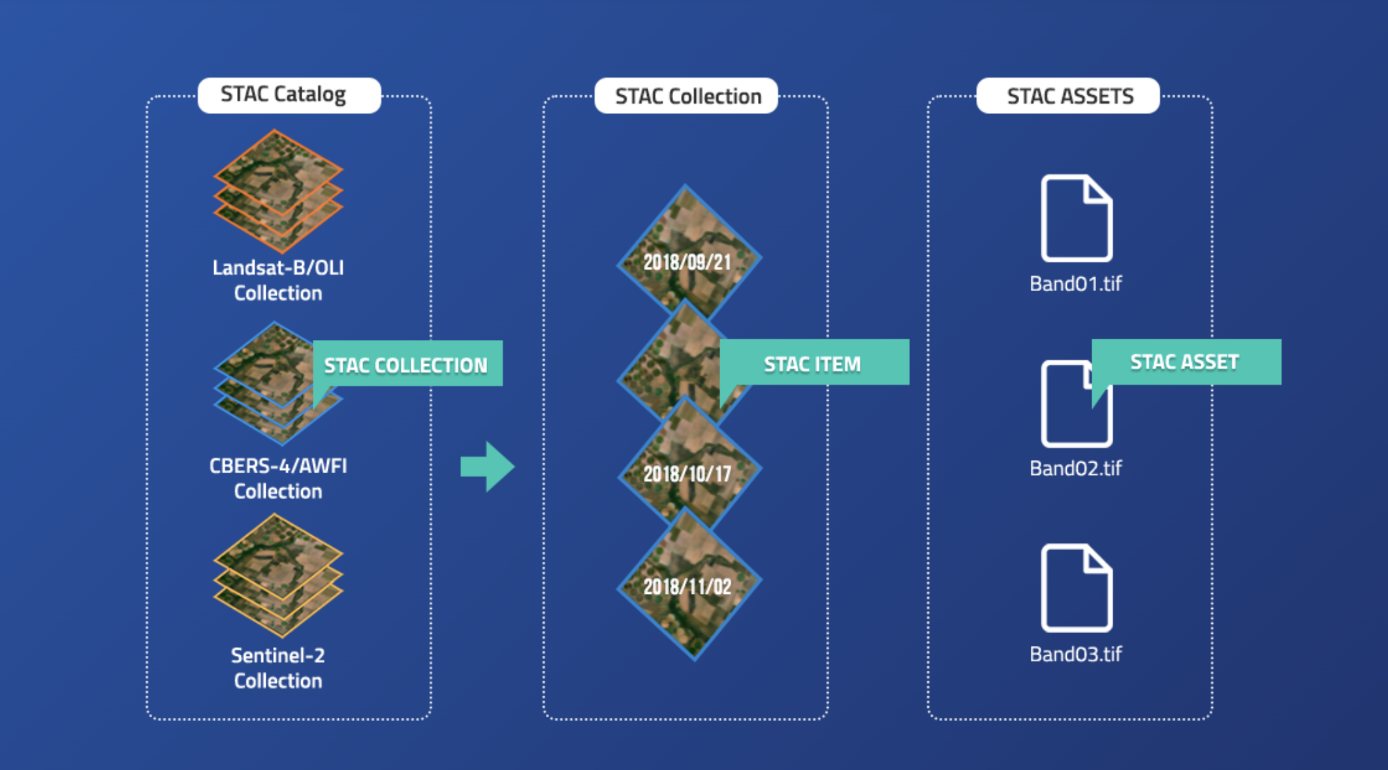
Dans le contexte actuel du projet, les principaux acteurs sont ceux de l'Institut Géographique National (IGN), qui souhaitent labelliser les données à partir de leur catalogue existant pour les utiliser dans des outils d'intelligence artificielle ultérieurement. Cependant, en raison du stade précoce du projet et de son caractère conceptuel, nous n'avons pas encore pu engager de véritables utilisateurs potentiels dans le processus de développement. Pour cette raison, notre principal point de contact et de feedback reste le commanditaire lui-même. Nous nous efforçons de recueillir le maximum de retours sur nos propositions auprès du commanditaire et sur l'existant afin de nous assurer que nos choix et nos développements répondent aux besoins réels de l'IGN et de ses utilisateurs futurs.

Les différents rôles supplémentaires envisagés, tels que le gestionnaire de chantier, le labellisateur d'images et l'approbateur, pourraient éventuellement être discutés et intégrés dans les derniers jours du développement. Cette approche permettrait de prendre en compte les retours éventuels du commanditaire et d'adapter les fonctionnalités en conséquence, tout en maximisant l'utilité et l'efficacité de l'application pour les utilisateurs finaux. En somme, bien que les utilisateurs finaux ne soient pas encore directement impliqués, nous nous efforçons de concevoir une solution qui réponde au mieux à leurs besoins potentiels, tout en respectant les exigences du commanditaire et les contraintes du projet.

1. Description de l'application et Glossaire

L’application utilise en entrée qui est une URL, cette URL pointe vers une STAC, spatio temporal asset catalog, qui est « le catalogue d’image » généralement stocker sur internet ou sur la machine de l’utilisateur. Et ce « catalogue d’image » utilise des COG, cloud optimized geotiff qui sont les « images ».

L'élément STAC est l'unité atomique de base, représentant un actif spatio-temporel unique sous la forme d'une caractéristique GeoJSON avec une date et des liens. Le catalogue STAC est un fichier JSON simple et flexible de liens qui fournit une structure pour organiser et parcourir les éléments STAC. STAC permet en effet de créer des collections d'images (ou de tout autre type de données géospatiales) pour les organiser de manière cohérente. Il fournit un moyen standardisé de décrire ces collections, y compris les métadonnées associées telles que la date, le lieu, la résolution, etc. STAC facilite la découverte, l'accès et l'exploitation de ces données en les organisant de manière standardisée et en fournissant des liens vers les données elles-mêmes.



Représentation d’une STAC

Les STAC, qui sont les catalogues, contiennent les COG, qui sont les images. Un Cloud Optimized GeoTIFF (COG) est un fichier GeoTIFF normal, destiné à être hébergé sur un serveur de fichiers HTTP, avec une organisation interne qui permet des flux de travail plus efficaces sur le cloud. Il découpe les images en tuiles et les rend accessibles via des adresses URL, ce qui permet une visualisation rapide et efficace des données. COG ne se limite pas à la visualisation, mais rend également la manipulation des données plus efficace en permettant l'accès à des portions spécifiques de l'image sans avoir à télécharger l'ensemble du fichier. Lorsqu'une seule grande image est stockée, pour accéder à une portion spécifique de celle-ci, il est nécessaire de charger l'intégralité de l'image. Cela peut entraîner un gaspillage de bande passante et de ressources, surtout si seule une petite partie de l'image est nécessaire. En revanche, avec les tuiles, seules les tuiles pertinentes pour la visualisation sont chargées, ce qui réduit la quantité de données transférées et accélère le temps d'accès. Les images COG sont généralement organisées en plusieurs niveaux de résolution, ou niveaux de zoom. En stockant chaque niveau de zoom sous forme de tuiles distinctes, on peut optimiser l'accès aux données en fonction du niveau de zoom demandé. Par exemple, pour un zoom faible, les tuiles de basse résolution sont utilisées, tandis que pour un zoom élevé, des tuiles de haute résolution sont chargées. Cela permet une expérience utilisateur fluide et une consommation de ressources réduite. Diviser une grande image en tuiles permet également une gestion plus efficace des métadonnées associées à chaque tuile. Plutôt que d'avoir une seule ensemble de métadonnées pour toute l'image, chaque tuile peut avoir ses propres métadonnées spécifiques, ce qui facilite la gestion et la recherche des informations pertinentes.

Une image contenant motif, capture d’écran, art

Description générée automatiquement

Représentation d’un COG

L'authentification sur le site se fait par le biais d'un nom d'utilisateur et d'un mot de passe, assurant ainsi la sécurité et la confidentialité des données.

L’application possède 3 rôles : gestionnaire, annotateur et reviewer.

Un gestionnaire est la personnes qui créer un chantier, choisis les images, la nomenclature et attribut les différents rôles aux autres utilisateurs.

Un annotateur peut annoter le chantier sur lequel il a été assigné, et enregistrer les différents patch lorsqu’il à finis d’en annoter un.

Le reviewer à accès au chantier lorsque l’annotateur à finis son travail. Il à une interface dédié pour comparer l’image segmenté de base aux annotations de l’annotateur. Il peut faire défiler les patchs et accepter ou rejeter avec un message.

Ce processus, tel que décrit, offre une navigation fluide entre les différents rôles, assurant ainsi une continuité dans le travail de l'utilisateur tout en maximisant son efficacité dans la gestion des données géographiques.

**Analyse fonctionnelle**

Pour la partie d'analyse fonctionnelle, nous allons présenter les différentes fonctionnalités et où nous en sommes arrivé après un mois de développement de l’application. Nous avons une interface web, le front-end, reliée à un backend qui lui relie nos bases de données à notre interface.

1. Base de données et entités de l’application

Nous avons deux bases de données différentes, une bdd « data » plus dirigé pour travailler sur les images, et une autre bdd « gestion » pour stocker les informations des nomenclature créer par les utilisateurs et d’autres informations liées aux utilisateurs.

Une image contenant texte, diagramme, capture d’écran, document

Description générée automatiquement

Nous pouvons charger une STAC dans notre application via une URL dans notre interface web. L’application va donc afficher les COG qui sont contenue dans la STAC.

La base de données « data » est utilisée pendant tout le processus de la chaîne de traitement de l’application. Lors de l’authentification, lors de la création d'un chantier, l’utilisateur renseigne l’URL de sa STAC puis choisit les images qu’il souhaite. Cela crée une ligne dans la table chantier pour chaque image sélectionnée, avec une correspondance dans la table image\_sortie. Lorsqu’il choisis les user annotateur et reviewer de même dans la table « user ».Ensuite, dans l’interface de labellisation, les utilisateurs annotent patch par patch, chaque image est découpée en plusieurs patch. Nous faisons ceci image par image. Ensuite, nous segmentons chaque patch à la volée. Cela correspond à segmenter le patch affiché par l’utilisateur et donc ne pas segmenter toute l’image d’un coup, ce qui prendrait plusieurs minutes. Au lieu de cela, chaque patch prend environ 5 secondes à se charger, le nombre de patch peut être variable par image, entre 5 et 800 pour des petites résolutions. Ensuite, chaque patch qui est labellisé peut être enregistré dans la table patch. Des clés primaires assurent l’authenticité de chaque chantier, chaque image et chaque patch. De plus, des clés étrangères permettent de garantir les liens de parenté entre ces trois entités, assurant ainsi l'intégrité et la cohérence des données stockées dans la base de données.

Pour ce qui est de la base données nomenclature, elle sert à enregistrer une nomenclature pour un utilisateur et permettre de réutiliser cette nomenclature par la suite dans un autre chantier.

La base de données « nomenclature » est composée de deux tables : une table "nomenclature" qui contient toutes les informations générales, et une table "style" qui décrit tous les libellés et les couleurs associés à ces libellés.

L’utilisateur dispose de trois choix lorsqu’il s’agit de choisir la nomenclature de son chantier. Le premier choix est de charger une nomenclature en important directement un fichier au format « .csv ». Le deuxième choix est de définir lui-même une nomenclature. Lorsqu’il clique sur le bouton « définir une nomenclature », il peut définir des libellés et choisir les couleurs associées à ces libellés, offrant ainsi une personnalisation et une adaptabilité à ses besoins spécifiques.

1. Design et chaine de traitement de l’application

Nous sommes en pleine transition pour la partie graphique de l’application. Voici un design de l’application qui comprends l’intégration des trois rôles : gestionnaire de chantier, labellisateur et revieweur. Les autres diagrammes ne sont pas parfaitement en accord avec ce dernier ajout pour l’instant.

La page de connexion lorsqu’on arrive sur le site.

Une image contenant texte, capture d’écran, logiciel, Police

Description générée automatiquement

Ensuite la page d’accueil, avec les chantiers qui sont en cours, on distingue trois parties pour les trois différents rôles existant. Chaque utilisateur peut être gestionnaire d’un chantier, labellisateur d’un autre et revieweur d’un dernier. Un rôle n’est pas limitant.

Une image contenant texte, capture d’écran, diagramme, Parallèle

Description générée automatiquement

Ensuite vient la page de création de chantier par un gestionnaire de chantier.

Une image contenant texte, capture d’écran, diagramme, nombre

Description générée automatiquement

Puis vient la page de labellisation par un labellisateur.

Une image contenant texte, capture d’écran, diagramme, Parallèle

Description générée automatiquement

On peut voir en haut à gauche la prévisualisation du patch pour situer ce dernier dans l’image. Le slider de hiérarchie correspond au niveau de segmentation que l’utilisateur peut changer en temps réel.

Une image contenant texte, capture d’écran, diagramme, Parallèle

Description générée automatiquement

L’utilisateur peut sélectionner les images qu’il souhaite, laisser un message de review et accepter ou rejeter la labellisation.

**Etude technique**

1. L’architecture du projet

Une image contenant texte, diagramme, croquis

Description générée automatiquement

L'architecture adoptée pour notre application est une architecture classique en trois tiers. Le premier tier consiste en une base de données serveur, qui inclut deux bases de données distinctes : « data » et « gestion », toutes deux implémentées avec PostgreSQL et déployées via Docker pour garantir la flexibilité et la portabilité. Le deuxième tier est représenté par une API développée avec Flask en Python, qui facilite la création de routes et l'exposition des bases de données. Enfin, le troisième tier est l'interface client, développée avec Vue.js, un framework progressif pour la construction d'interfaces utilisateur dynamiques.

1. Choix des technologies

Nous sommes très reconnaissants envers notre commanditaire pour la liberté accordée dans le choix des technologies, un aspect crucial de notre projet. Les premières réunions, qui ont duré plusieurs heures, ont été consacrées à ce sujet afin de poser des bases solides pour le développement de l'application.

Le premier défi a été la gestion des images. Suivant les recommandations de notre commanditaire, nous avons approfondi notre compréhension des technologies STAC et COG, bien que leur maîtrise ait été complexe au début. Nous avons conclu que l'approche par catalogues avec STAC, combinée à la gestion en pyramide de résolutions multiples avec COG, correspondait parfaitement à nos objectifs.

En parallèle, nous avons exploré les différentes options pour les applications web cartographiques. Parmi les plus connues et récentes, telles qu'OpenLayers et Leaflet, OpenLayers a été retenue. La disponibilité d'une documentation riche sur l'utilisation de STAC avec OpenLayers a renforcé notre choix, assurant une cohérence entre la gestion des images et l'application cartographique choisie.

Une image contenant logo, Police, Graphique, symbole

Description générée automatiquement

Le choix de PostgreSQL a été très rapide de par son efficacité, sa reconnaissance et sa gratuité. Cette technologie de base de données couplé à Docker est très efficace pour avoir une flexibilité et déployer nos base de données rapidement dans plusieurs environnements.

Pour notre projet, le choix de Flask plutôt que Django s'est avéré judicieux pour plusieurs raisons spécifiques à nos besoins. Premièrement, Flask offre une grande flexibilité et légèreté, ce qui est essentiel pour notre application qui nécessite une architecture personnalisée et épurée. Contrairement à Django, qui inclut de nombreuses fonctionnalités intégrées, Flask nous permet de démarrer avec un noyau minimaliste et d'ajouter uniquement les extensions nécessaires, évitant ainsi toute surcharge inutile.

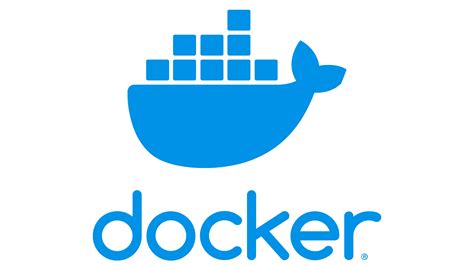
De plus, Flask est reconnu pour sa facilité d'intégration avec d'autres technologies et pour sa simplicité d'utilisation, ce qui est crucial dans un contexte où nous devons souvent tester et modifier rapidement des aspects du back-end en réponse aux exigences du commanditaire ou aux retours des utilisateurs. Cette simplicité nous permet de rester agiles et de nous adapter rapidement aux changements sans être liés par une structure trop rigide.

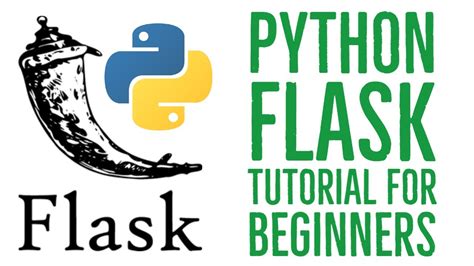
Enfin, la communauté autour de Flask est très active et le support est abondant, ce qui est un avantage significatif pour résoudre rapidement tout problème ou pour intégrer des fonctionnalités supplémentaires. Bien que Django soit également bien soutenu, la légèreté et la modularité de Flask correspondent mieux à notre besoin de maintenir une application facilement modifiable et évolutif.

Pour le développement de notre interface utilisateur, nous avons opté pour Vue.js plutôt que pour d'autres frameworks populaires tels que React ou Angular, principalement en raison de sa simplicité et de sa flexibilité. Ce framework est conçu pour être progressivement adoptable, permettant aux développeurs d'intégrer Vue dans leur projet à leur rythme, partie par partie, sans nécessiter une refonte complète de l'architecture existante.

En outre, Vue.js offre une intégration et une gestion des états très intuitives, avec des fonctionnalités comme les composants dynamiques et le système réactif qui facilitent la création d'interfaces utilisateur interactives et performantes. Le système de composant permet aussi de collaborer facilement entre développeur car nous pouvons ajouter, déplacer et supprimer du code sans remettre en questions toute l'architecture. La documentation de Vue est également extrêmement bien rédigée et structurée, fournissant aux développeurs toutes les informations nécessaires pour exploiter pleinement le potentiel du framework.

Comparativement à React et Angular, Vue.js est souvent perçu comme plus léger et moins prescriptif, offrant une plus grande liberté dans la structure du code sans imposer d'architecture ou de paradigmes spécifiques. Cela permet une plus grande créativité et adaptation aux spécificités du projet, tout en bénéficiant d'une écosystème robuste et d'une communauté active et engagée pour le support et les mises à jour.





**Réalisation du projet**

1. Outils et technologies pour le développement collaboratif

Dans la réalisation de notre projet, plusieurs outils et technologies ont été mis en œuvre pour faciliter la collaboration et assurer la qualité du développement. Nous avons utilisé GitHub comme plateforme principale pour la gestion du code source et la collaboration entre les membres de l'équipe. GitHub nous a permis de suivre efficacement les contributions de chaque développeur, de gérer les branches de fonctionnalités et de résoudre les conflits de manière centralisée, ce qui a renforcé la cohérence et l'efficacité de notre travail d'équipe.

Pour gérer les dépendances et assurer la compatibilité des versions de notre environnement de développement, Node.js a été utilisé. Cela nous a aidés à maintenir un environnement uniforme pour tous les développeurs, facilitant ainsi les tests et le déploiement sans accroc de l'application.

En ce qui concerne le testing, nous avons choisi la bibliothèque Vitest pour tester notre code JavaScript. Vitest offre une exécution rapide des tests et une intégration simple avec l'écosystème JavaScript moderne, ce qui nous permet de garantir que notre application fonctionne comme prévu à chaque étape du développement, en réduisant les régressions et en améliorant la stabilité du code.

Pour automatiser notre processus d'intégration continue, nous avons implémenté GitHub Actions. Cette fonctionnalité de GitHub nous a permis de configurer des pipelines automatisés pour exécuter des tests, effectuer des analyses de code et déployer notre application sur des serveurs de test. L'utilisation de GitHub Actions a significativement augmenté notre productivité en automatisant des tâches répétitives et en nous permettant de détecter et de corriger rapidement les problèmes avant la mise en production.

L'utilisation combinée de ces outils a créé une fondation solide pour notre développement, assurant une collaboration efficace et une qualité de code élevée tout au long du projet.

1. Télétravail et suivie de projet

Pendant tout le déroulement du projet, nous avons adopté un mode de travail entièrement en télétravail, une première expérience pour la plupart d'entre nous. Pour garantir une collaboration efficace malgré la distance, nous avons opté pour une approche Agile. Au début du projet, nous avons établi un tableau Trello exhaustif avec toutes les tâches à accomplir, ce qui nous a permis d'avoir une vue d'ensemble claire de nos objectifs et de notre progression. Chaque semaine, nous avons redéfini nos sprints, en ajustant nos priorités et en planifiant les prochaines étapes.

Discord a joué un rôle essentiel dans notre communication quotidienne. Nous avons organisé des rétrospectives à la fin de chaque après-midi pour évaluer notre avancement et discuter des problèmes éventuels. De plus, chaque matin, nous nous sommes réunis pour planifier notre journée, partager nos objectifs individuels et discuter de tout problème nécessitant une résolution. Les salons vocaux Discord ont été particulièrement utiles, car ils nous ont permis d'être disponibles en un clic pour des discussions spontanées ou pour demander de l'aide si nécessaire.

**Conclusion**

Dans ce projet, initié par l'IGN, notre objectif était de répondre à un besoin croissant en matière de labellisation d'images géographiques, notamment pour les cartes anciennes et les images aériennes. Cette initiative s'inscrit dans une volonté d'ouverture des données géomatiques de l'IGN et de diffusion des travaux en intelligence artificielle.

Pour cela, nous avons développé une solution web, permettant la labellisation des images nécessaires à l'utilisation de méthodologies d'apprentissage machine supervisé ou semi-supervisé, spécifiquement adaptées aux problématiques de l'IGN. Cette interface intuitive offre des fonctionnalités avancées telles que la création de chantiers, la segmentation d'images en zones, la gestion dynamique des nomenclatures et la répartition des tâches en différents rôles.

Sur le plan technique, nous avons fait le choix de technologies modernes et adaptées à nos besoins, telles que PostgreSQL pour la gestion des bases de données, Flask pour l'API back-end, et Vue.js pour l'interface utilisateur. L'utilisation de GitHub pour la collaboration, de Node.js pour la gestion des versions, ainsi que de librairies telles que Vitest pour les tests JavaScript et GitHub Actions pour l'intégration continue ont également contribué au succès de notre projet.

Malgré les défis liés au télétravail à 100%, nous avons adopté une approche Agile, avec des mises à jour régulières de notre tableau Trello et des réunions quotidiennes sur Discord pour suivre notre progression, résoudre les problèmes et maintenir notre cohésion d'équipe.