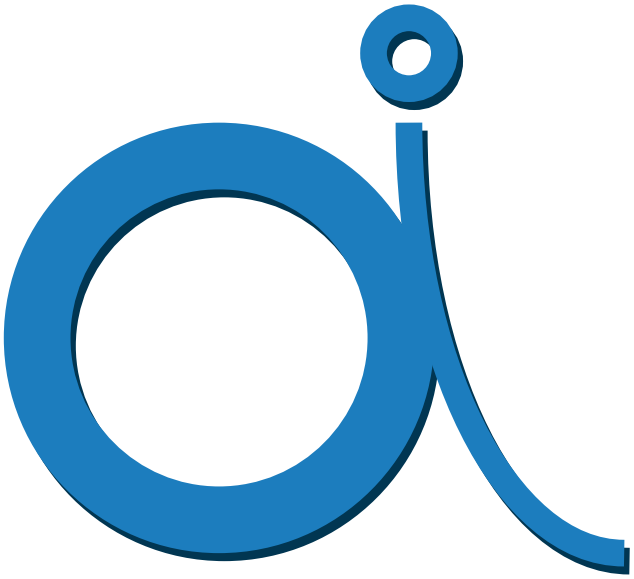
Desgenetez Clément
MMI 2015/2016



# Online Image Annotation

4 avril - 10 juin 2016

Gérard Subsol
Marc Chaumont
Jérôme Azé

## Table des matières :

# Présentation et Remerciements

# Remerciements

Je tiens à remercier :

William Puech (Responsable du projet ICAR), qui a porté ma candidature, ainsi qu'à Gérard Subsol et Marc Chaumont qui quand, le projet est apparu, ont pensé directement à cette dernière.

Sébastien Villéger pour ses retours sur l'application à chaque rapport.

Sébastien Villon qui m'aiguillait en cas de besoins.

Thomas Claverie pour ses avis.

Les trois stagiaires MARBEC en charge des annotations, pour leur compréhension et leur retours sur les bogues de l'application.

David Mouillot qui, après avoir vus mon stage, m'a proposé une place pour un autre projet cet été.

Et pour finir l'IUT de Béziers pour ces deux années qui m'ont permis de choisir mon orientation future ainsi que toutes les autres personnes avec qui j'ai été amené à travailler.

Mon intention est de continuer mes études jusqu'au Master. Ce stage m'a offert la possibilité de rencontrer et de parler avec des personnes issues de ces formations.
De plus les domaines traités sont tous différents et donnent une vision solide et diversifiée, ce qui m'a éclairé sur mes choix

Ce projet s'est déroulé du 4 avril au 10 juin 2016 au sein de l'équipe Icar du LIRMM (LIRMM : Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier) en collaboration avec l'Unité Mixte de Recherche (UMR) MARBEC, MARine Biodiversity, Exploitation and Conservation de Montpellier.

Le Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier - LIRMM - est une unité mixte de recherche, dépendant conjointement de l'Université de Montpellier et du Centre National de la Recherche Scientifique. Il est situé sur le Campus Saint-Priest de l'UM.
Ses activités de recherche positionnent pleinement le LIRMM au cœur des sciences et technologies de l'information, de la communication et des systèmes.
Ainsi, de l'information aux systèmes, de la technologie à l'humain et aux usages, les activités de recherche du LIRMM concernent: la conception et la vérification de systèmes intégrés, mobiles, communicants, la modélisation de systèmes complexes à base d'agents, les études en algorithmique, bio informatique, interactions homme-machine, robotique, etc.
Les travaux sont menés dans trois départements scientifiques de recherche, eux-mêmes organisés en « équipes-projet ».

L'Unité Mixte de Recherche (UMR) MARBEC, MARine Biodiversity, Ex-ploitation and Conservation, a été créée le 1er janvier 2015. Ses autorités de tutelle sont l'IRD, l'Ifremer, l'Université de Montpellier et le CNRS.

MARBEC est l'un des plus importants laboratoires travaillant sur la biodiversité marine et ses usages en France avec environ 230 agents, dont 80 chercheurs et enseignants-chercheurs.

L'unité est implantée à Sète, Montpellier et Palavas-les-flots, ainsi que dans l'océan Indien, en Asie, en Afrique et en Amérique du Sud.

Elle étudie la biodiversité marine des écosystèmes lagunaires, côtiers et hautu-riers, principalement méditerranéens et tropicaux. Ses recherches portent sur différents niveaux d'intégration, des aspects moléculaires, individuels, popula-tionnels et communautaires, aux usages de cette biodiversité par l'Homme.

L'équipe ICAR (Image & Interaction) développe des activités de recherche associant l'interaction et le traitement des données visuelles telles que les images, les vidéos et les objets 3D.
L'équipe ICAR est structurée suivant 3 axes :

- Analyse & Traitement (AT),
- Codage & Protection (CP)
- Modélisation & Visualisation (MV).

L'axe AT s'intéresse à de nouvelles techniques de traitement bas-niveau de l'information représentant, dans un même cadre théorique, l'imprécis, l'incertain et l'incomplet (types d'erreur en traitement des données).
L'axe CP s'intéresse à la transmission et l'archivage sécurisés de données visuelles. Cette protection peut être assurée par tatouage, stéganographie, ou chiffrement et peut nécessiter la robustesse à la compression.
L'objectif de l'axe MV est de modéliser des grands ensembles de données complexes (en dimension et en nature) afin de permettre une visualisation intuitive ou de manipuler ces données pour en extraire des connaissances.
https://www.lirmm.fr/icar/

# Le projet

Ce projet est une collaboration entre MARBEC (David Mouillot et Sebastien Villéger), le CUFR de Mayotte (Thomas Claverie) et le LIRMM (Marc Chaumont, Gérard Subsol, Sébastien Villon)

Il vise à mettre au point un algorithme de type « Deep-Learning » pour identifier des poissons dans des vidéos sous-marines avec comme application prioritaire les récifs coralliens de l'Ouest de l'Océan Indien. C'est un projet très ambitieux et innovant et un des points clés est de disposer rapidement de milliers de vignettes annotées (= photo d'un poisson et son nom).

Le traitement des vidéos se fait par « Deep-learning »
Le « Deep-learning » est un ensemble de méthodes permettant « l'apprentissage » d'une machine.
Pour faire apprendre à une machine, comme à une vraie personne d'ailleurs, il faut des exemples, beaucoup d'exemples.
Le nombre minimum requis d'images d'une même espèce de poisson qui a été défini par l'équipe comme suffisant est de 1000 images sans traitement.
Les vignettes une fois extraites sont copiées et retournées pour multiplier les angles de vue.

Ligne conductrice du stage :

Créer et mettre en place une plate-forme d’annotation d’images, en ligne.

Le cahier des charges qui va suivre est celui qui m’a été donné en début de stage.
Certaines parties ont été traitées comme prévu, d’autre ont été modifiées, ajoutées ou supprimées.

# Cahier des charges (Original) pour le logiciel d'annotations de vidéos

Version de travail du 01/04/2016 : T. Claverie, Q. Calas, S. Villéger, S. Villon

## Objectif principal : Annoter des poissons sur des vidéos

=> Entourer et identifier des individus pour extraire des vignettes pour le deep-learning (LIRMM) et faire des estimations de biodiversité (MARBEC)

## Spécifications techniques et pratiques:

- **être multi-utilisateurs**
  - ⇨ fonctionner sur un serveur (stockage des inputs et outputs)
  - ⇨ utilisation de l'anglais dans le code et les métadonnées

- **traçabilité facile de toutes les opérations**
  - ⇨ tables de métadonnées
  - ⇨ format de fichiers directement lisibles et importables sous R

- **être utilisable par une personne ayant un niveau en informatique moyen**
  - ⇨ opérationnel sur n'importe quel ordinateur (ou tablette ?)
  - ⇨ opérations seulement via interface graphique (pas de code à entrer)
  - ⇨ bonne ergonomie pour faciliter ce travail répétitif

- **être utilisable par une personne ne connaissant pas bien les poissons**
  - ⇨ menu déroulant et aide visuelle pour le choix des espèces ou type d'individus
  - ⇨ possibilité de vérifier les annotations

## Suggestions techniques et pratiques:

Accès au serveur sous 2 modes :

**=> Administrateur du serveur** (identifiant entré a priori + Password requis)**:** T. Claverie, S. Villéger, S. Villon et XX => seules personnes ayant la main sur l'upload des vidéos, la modification de la table des espèces et le téléchargement des résultats

=> **Utilisateurs** (identifiant unique requis, exemple : adresse mail valide ? + password ?)**:** pour 2016 seulement chercheurs de MARBEC et stagiaires encadrés à MARBEC, mais à moyen terme, possibilité d'utiliser l'outil pour faire de science participative à grande échelle.

**Vidéos uploadées**
- couper vidéos à max 5min avant de les uploader
- juste après l'upload, enregistrer (via fenêtre ?) dans un fichier (tableau) les métadonnées sur tournage sous l'eau (lieu, date, profondeur, météo, caméra, angle, résolution) et sur pré-processing (coupes).

**Menu d'accueil :**
- choix entre « videos » ou « projects »

- si « videos » fenêtre de type explorateur de fichiers avec toutes les vidéos disponibles sous forme de miniatures.

⇨ Vidéos pouvant être lues via double clic, s'ouvrant dans une autre fenêtre

- Si « projects » fenêtre de type explorateur de fichiers avec tous les dossiers de projets déjà existant

⇨ Si clic sur un projet, ouverture de l'interface « annotation »

- bouton retour dans chaque sous-menu

**Génération de frames à partir d'une vidéo**

- sélection d'une vidéo par clic droit

⇨ valider choix « créer un nouveau projet »

- ouverture d'une fenêtre « project » permettant de choisir les paramètres pour la génération des frames (valeurs par défaut déjà entrées): fréquence (nb de frames/seconde), format de stockage (jpeg, tiff, ppm)

⇨ création d'un dossier avec comme nom (nomvideo_ freqframe_IDuser)

⇨ Si vidéo déjà traitée avec même paramètres => message d'avis proposant une redirection vers dossier projet déjà existant ?

- clic sur bouton « générer les frames »

⇨ enregistrement de ces métadonnées (nom video, date, user ID, nb frame/sec, format image) dans la table « projects » (= ajout d'une ligne)

⇨ une fois les frames générées, ouverture automatique de l'interface « annotation »

**Interface « annotation » = fenêtre avec 4 éléments**

- barre d'état (en haut), affichage de : IDuser, nom de la vidéo, rang de la frame affichée (exemple : « frame 2/2500 »), nombre total de poissons annotés sur la frame

- menu déroulant pour choix des espèces (¼ droit de l'écran) : construit à partir d'un tableau avec informations taxonomiques stockées dans un fichier txt hébergé en ligne
donc facilement modifiable (pas dans le code du programme), idem pour base de photos.

=> affichage des espèces par ordre taxonomique = 2 niveaux : famille (en gras) + espèces (en italique). Noms latins pour les espèces avec pour celles ayant un fort dimorphisme, précision du sexe (« male » or « female ») ou du stade de vie (« juvenile » or « adult »), exemple : « Homo_sapiens_male ». Pour individu dont l'espèce est non identifiable mais la famille oui  Possibilité de cliquer sur un nom de famille? ou nom générique type « Homo_sp ».

- barre d'outils (en bas) avec 9 boutons : avance/retour de 1, 10, 50, 100 frames + bouton « retour menu »

- photo à annoter sur ¾ gauche de l'écran.

**Processus d'annotation :**

- sélection d'une frame via boutons de navigation
  => génération d'un fichier txt dont le nom est du type « nomProjet_IDframe » pour stocker les informations

- sélection d'une espèce dans le menu déroulant par clic
  => affichage d'une photo type de l'espèce juste à côté

- clic sur l'image
  => affichage du premier coin et du nom de l'espèce juste au dessus

- mouvement de la souris
  ⇨ fait apparaître un carré rouge dont taille= maximum (longueur, hauteur du mouvement)

- 2ème clic fixe le carré qui devient vert
  ⇨ enregistrement des informations dans le fichier txt « nomProjet_IDframe  » : position des 4 coins du carrés, nom de l'espèce, IDuser, date

**Traçabilité :**

Lorsqu'on visualise une frame déjà annotée on voit tous les individus identifiés via des carrés verts. Lorsqu'on passe la souris sur un carré il devient orange et le nom de l'espèce s'affiche (et tous les individus de la même espèce sont aussi entourés en orange ?).
Clic droit à l'intérieur du carré offre le choix « supprimer » (et « modifier nom » ?)
Possibilité d'annoter de nouveaux individus (en particulier si ouverture d'un projet déjà existant).

# Déroulement du stage et l’application

Je suis arrivé le lundi 4 avril avec, en début de journée, réunion de l'équipe ICAR où j'ai pu suivre les présentations de projet d'autres stagiaires.
L'après-midi, j'ai eu une réunion avec MARBEC pour le projet d'annotation et une première discussion sur l'interface voulue.

Un rapport hebdomadaire est envoyé aux personnes concernées par ce projet pendant tout le stage.

J'ai proposé une plate-forme Web, pour les raisons suivantes :

- Aucune installation nécessaire, il faut seulement un navigateur Web à jour.
- Peut être multi-plates-formes (Ordinateur, tablette, etc.)
- Adapté pour l'interaction avec l'utilisateur.
- Comme demandé, tout est sur un serveur.

C'est donc sur le concept d'un site que j'ai travaillé.

Une autre idée m'est ensuite venu sur le principe du Google drive où les fichiers peuvent être modifiés en direct et à plusieurs sans avoir à les enregistrer. Permettent ainsi l'interaction à plusieurs sur la même image.

Pour la mise en place de la plate-forme Web le WebSocket s'est avéré être la meilleure solution et apporte un coté dynamique à l'application.

Pourquoi WebSocket ?

- Seul recquis : un navigateur à jour.
- Permet une communication en direct très simple avec le serveur.
- Invisible et rapide pour l'utilisateur.
- Peu complexe à mettre en place avec NodeJS et socket.

Pour mettre en place WebSocket, je suis parti sur une technologie JavaScript avec NodeJS coté serveur.

Bien que l'utilisation des web sockets soit possible avec PHP ce n'est pas vraiment son point fort et n'est pas pratique à l'utilisation.

NodeJS peut être combiné avec une solution de package (« npm » en occurrence), l'un des packages qui permet entre autre la grande simplification des web sockets c'est Socket.IO
En effet NodeJS utilise du JavaScript de façon asynchrone, c'est à dire qu'il peut gérer plusieurs actions en « même » temps.
En comparaison le PHP, lui, lit le code de haut en bas dans l'ordre pour l'exécuter.
Mais PHP n'a pas été abandonné pour autant puisque c'est lui qui génère et gère le « site » à proprement parler.

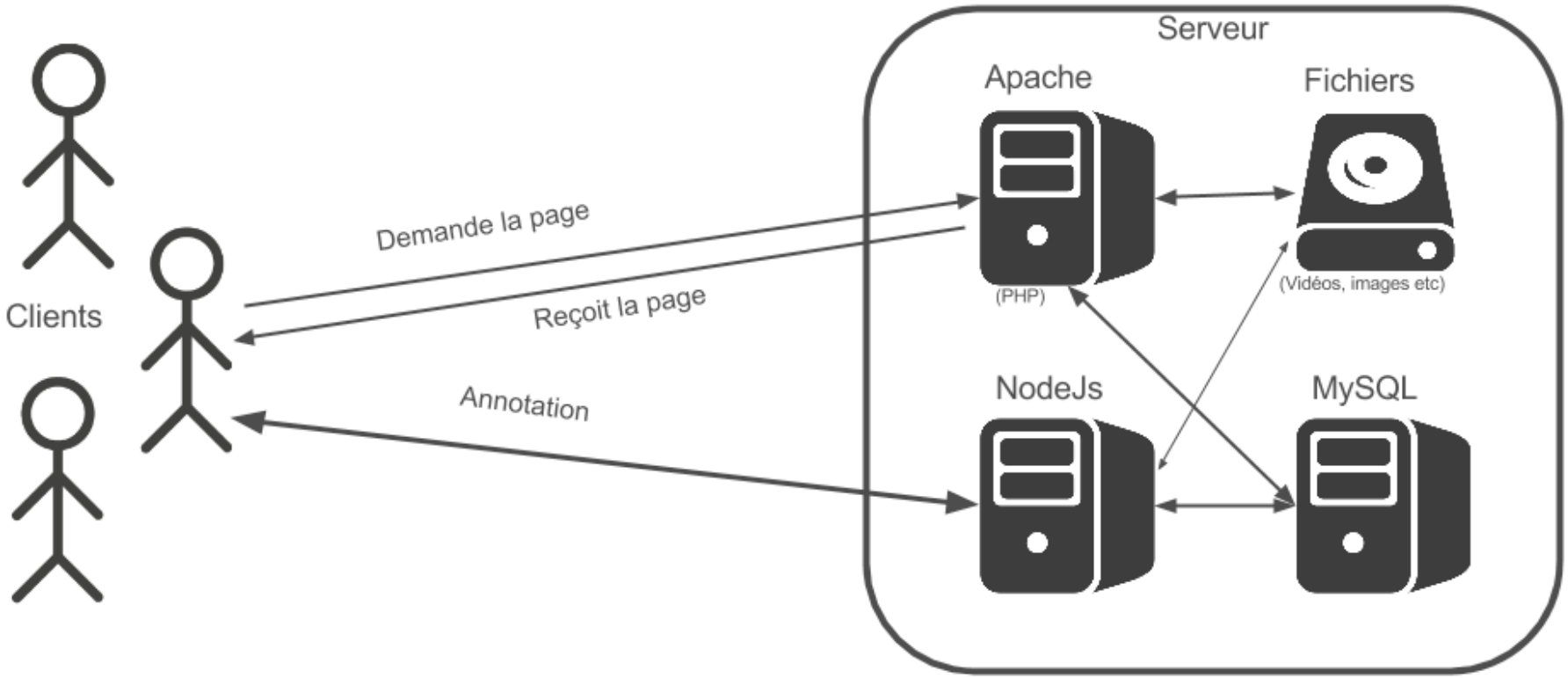



Schéma de l'architecture utilisée

On peut voir sur ce schéma que c'est le serveur MySQL qui fait une jointure entre le serveur Apache et le serveur NodeJS s'il y a besoin de passer des informations entre les deux. (pour l'authentification des personnes par exemple)

L'inconvénient majeur de cette solution est qu'elle demande à ce qu'un serveur NodeJS tourne en permanence, un simple hébergement Web ne suffit donc pas.

Pour reprendre le déroulement du stage, mes premiers jours se sont portés sur la partie « annotation ». C'est la partie identification et dessin.

L'annotation se fait donc avec du JS (Abréviation de JavaSript) mais cette fois-ci coté client.

Pour dessiner, la balise *Canvas* est utilisée.

Cette balise à la particularité de pouvoir dessiner dedans avec du JS (fonction pour créer des rectangles ...).

Ma première approche fut de dessiner dans un seul canvas. Cette approche sera modifiée après.

Bien que la balise *Canvas* permette le dessin, c'est un dessin « définitif ». Par exemple, en prenant la fonction pour dessiner des images (demande le chemin d'une image), si je dessine des pixels dessus, alors les pixels qui étaient dessous sont remplacés.

Rapidement on voit donc venir le problème : Comment faire pour dessiner un cadre de sélection qui lui n'écrase pas les pixels ?

Un petit temps de recherche, de prise en main et d'essais a été nécessaire.
Il faut redessiner en permanence. On écrase tous les pixels par de nouveaux. Il existe des fonctions pour ne traiter qu'une partie de l'image mais cela complique beaucoup la gestion pour peu de gains. Cette action de redessiner en permanence est invisible à l'œil car très rapidement exécutée.
Mais un problème se pose avec les images de très grande taille. Problème qui sera réglé par la suite.

Au départ Socket.IO n'était pas utilisé car une partie de son utilisation m'était inconnue ou incomplète.
C'est donc un package nomé WebSocket, moins pratique qui fut utilisé. Ce qui m'a demandé de recoder une partie de l'application.

Le fait de garder une connexion permanente avec le serveur offre un certain nombre de possibilités :

- Sauvegarder directement les annotations, ce qui évite les pertes suite à de mauvaises manipulations ou actions.
- Plusieurs utilisateurs sur la même image.
- Retour d'informations.

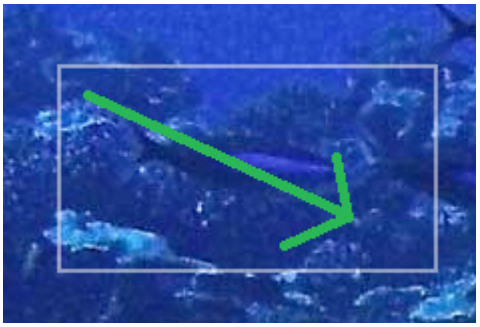
Problème : on ne pouvait transmettre que du texte avec le premier package.

J'ai donc mis en place un protocole en utilisant du JSON pour passer des informations.

Le JSON est un format standardisé pour le JS (mais qui est utilisable dans d'autres langages comme PHP) qui permet la conversion de variables, tableaux, objets en chaînes de caractères. L'opération inverse étant possible : une chaîne de caractères en tableaux, objets...


Résultat après deux semaines de stage

Sur cette image le curseur est dans un cadre. Ce qui change les bords de ce dernier en jaune et affiche le nom de l'espèces de poisson.



Cliquer glisser.

Par défaut l'événement cliquer glisser n'existe pas. La détection des actions est appelée événement. On peut écouter ces événements et leur résultat. En combinant les événements :

- Presser le clique gauche.
- Déplacement dans l'espace.
- Relâchement du clique gauche.

On obtient un cliquer glisser.

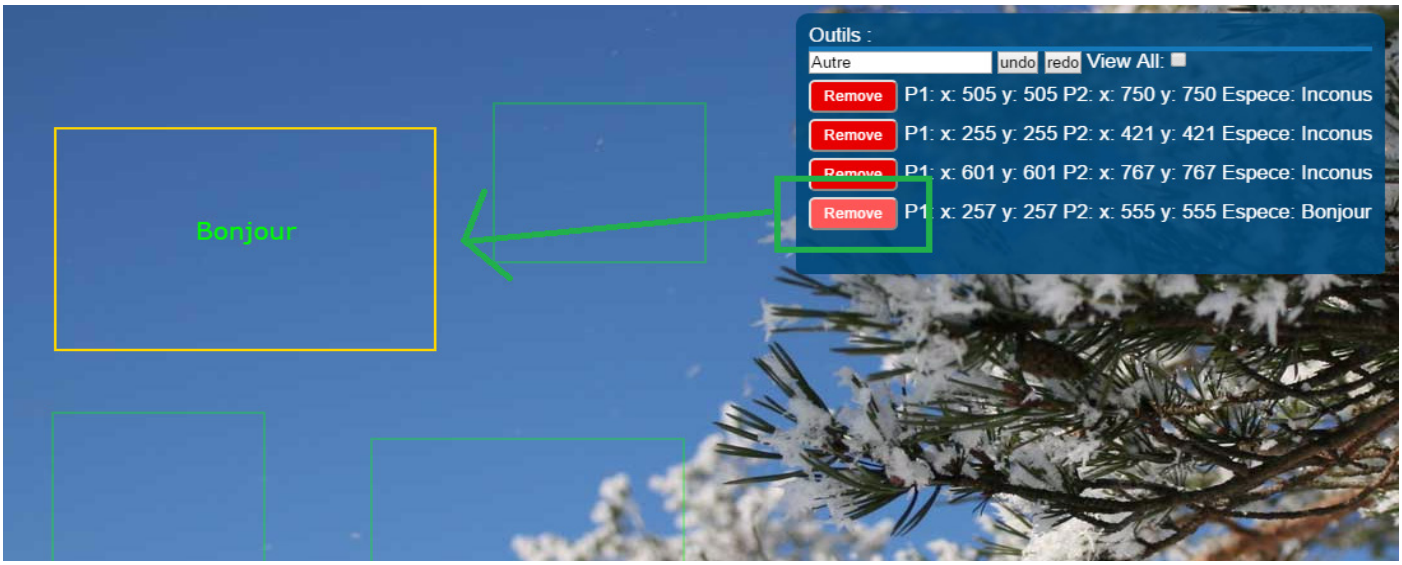
- Prendre la position au premier événement.
- Dessiner pendant le déplacement.
- Prendre la position du relâchement.

Le logiciel qui était utilisé auparavant pour annoter demandait de cliquer aux 4 points ; ce qui n'était ni pratique, ni rapide. Et « echap » pour valider n'était pas logique.

L’emplacement de la liste des objets encadrés se situe dans une « fenêtre » flottante limitée à la page. Le fait de mettre dans une fenêtre déplaçable, permet de libérer de la place pour l’annotation.

Les images dans le *Canvas* sont à taille réelle pour garder une cohérence dans les positions des cadres. Si l’écran n’est pas assez grand, il faut se déplacer dans la page.

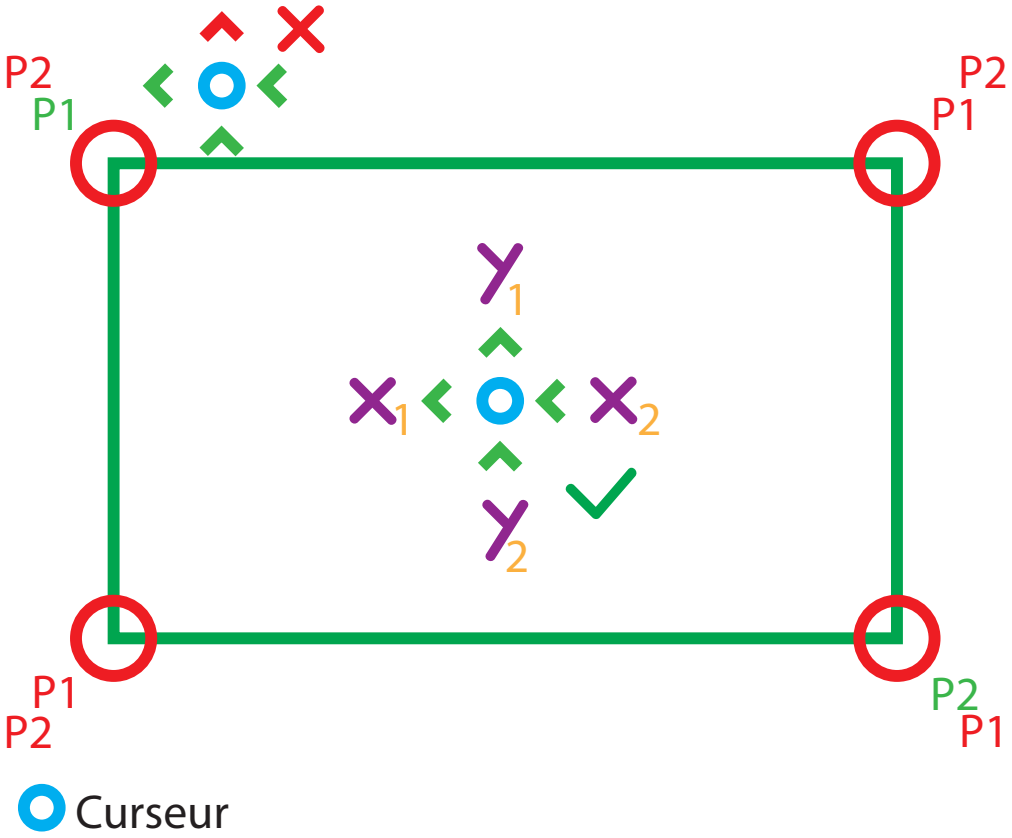
Si la liste était fixe comme pendant les premiers jours de développement, il faut descendre ou monter sur la page plus souvent. Ce qui est peu ergonomique




La transparence de la fenêtre permet de voir ce qui se trouve derrière. Plus de transparance sera appliqué par la suite pour une plus grande visibilité.
Passer son curseur sur le bouton de suppression encadre en jaune le cadre concerné.

En ce qui concerne la détection du curseur pour savoir s'il est dans un cadre, cette fonction n'existe pas il m'a donc fallu la développer.

Les premières fois, lorsque l'on encadrait un objet, les deux points pris étaient sauvegardés dans l'ordre. Ce qui posait des problèmes car p1 pouvait être plus bas que p2 mais aussi plus à droite ...



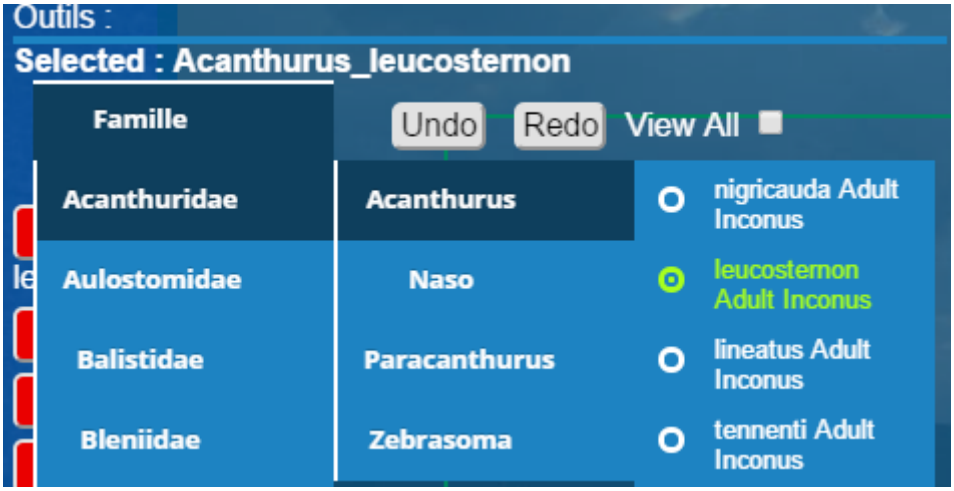

Il aurait fallu tester en permanence toutes les possibilités.

Pour simplifier, avant de sauvegarder, les point sont remis dans l'ordre. P1 est en haut à gauche P2 est en bas à droite. Avec cela, la condition est bien plus simple.

Un autre point qui était assez complexe. Est la possibilité d'annuler une action. Cela peut paraître simple, mais il faut sauvegarder chaque action et faire son opposé.

Et si l'on veut annuler une annulation alors il faut sauvegarder ce qui a été annulé. Simple en théorie mais si l'on supprime un objet particulier on peut vite se retrouver à supprimer le mauvais.


Une case à cocher sert à indiquer si l'on souhaite voir tous les noms en permanence.




Le déplacement de l'historique d'actions côté serveur évite des conflits entre l'affichage et les données reçues qui pouvaient ne pas concorder.

La suppression par clique.

Cliquer sur un cadre le sélectionne en rouge et appuyer sur suppr, le supprime.

Ajout de raccourcis clavier pour le changement de frame avec les flèches.

Du ctrl+z et ctrl+y pour annuler ou refaire plus vite.

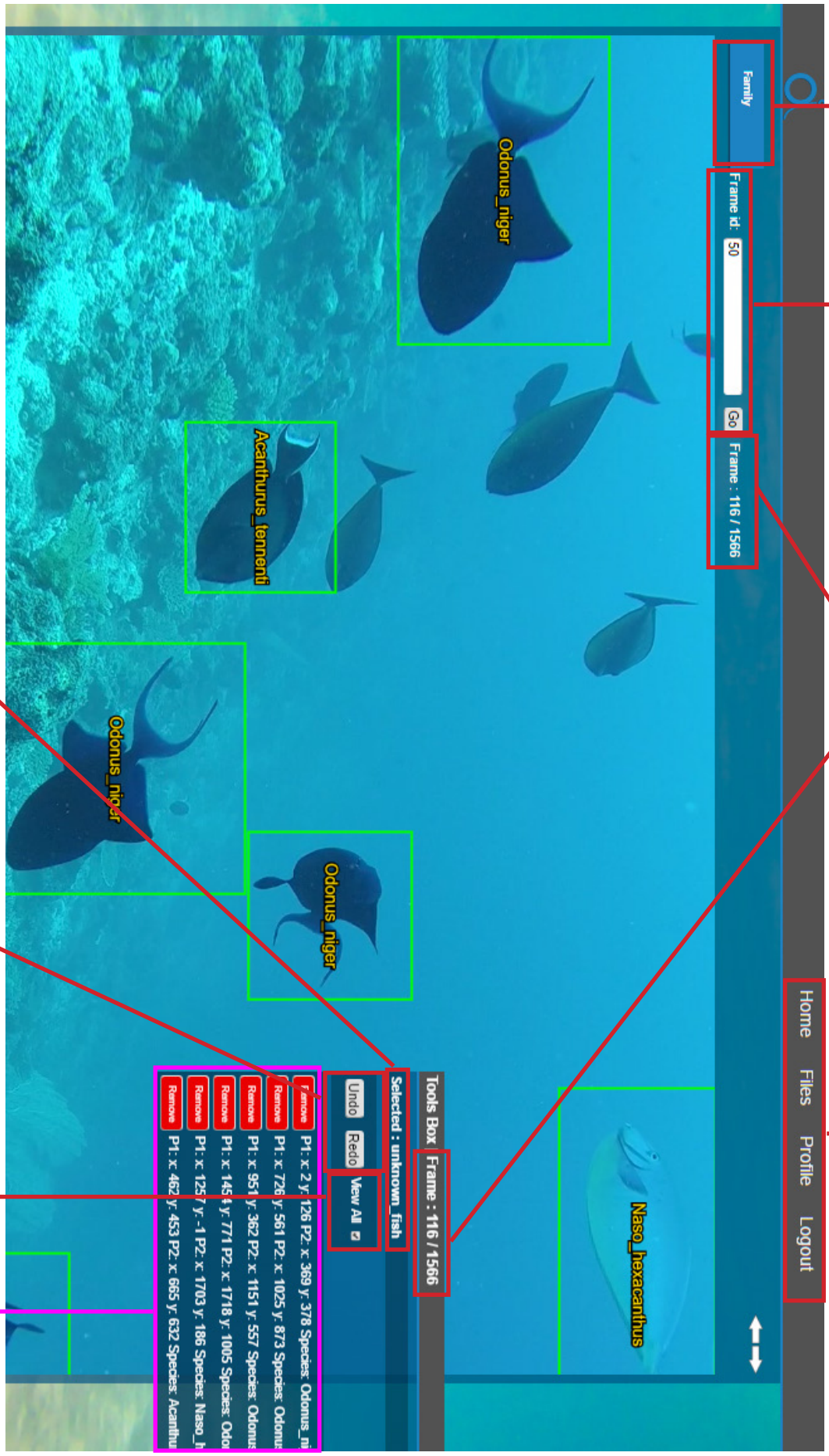
Pour finir et revenir sur le fait que, d'origine un seul *Canvas* était utilisé, après plusieurs essais deux *Canvas* sont désormais utilisés en superposition l'un à l'autre.

Le premier en fond pour l'image.

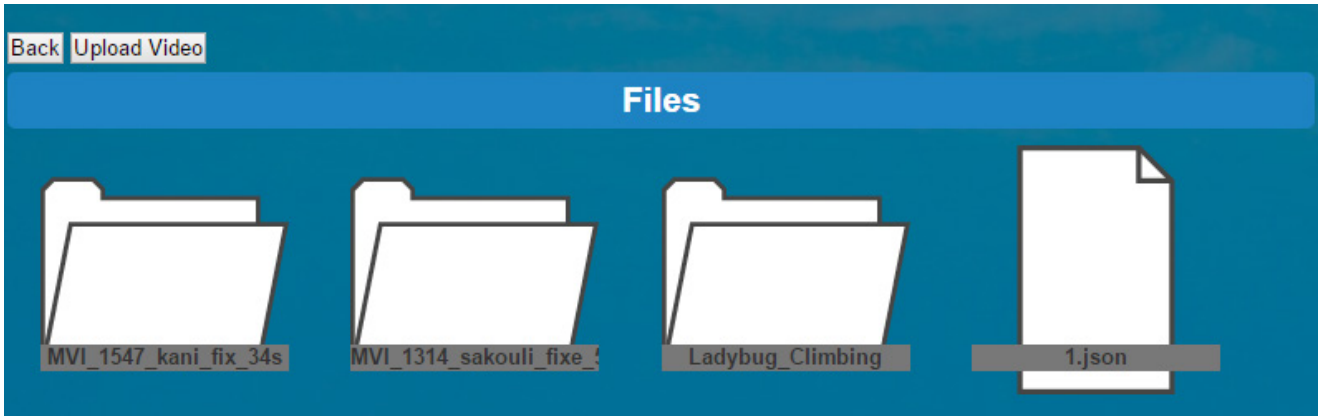
Le second en face pour les rectangles.

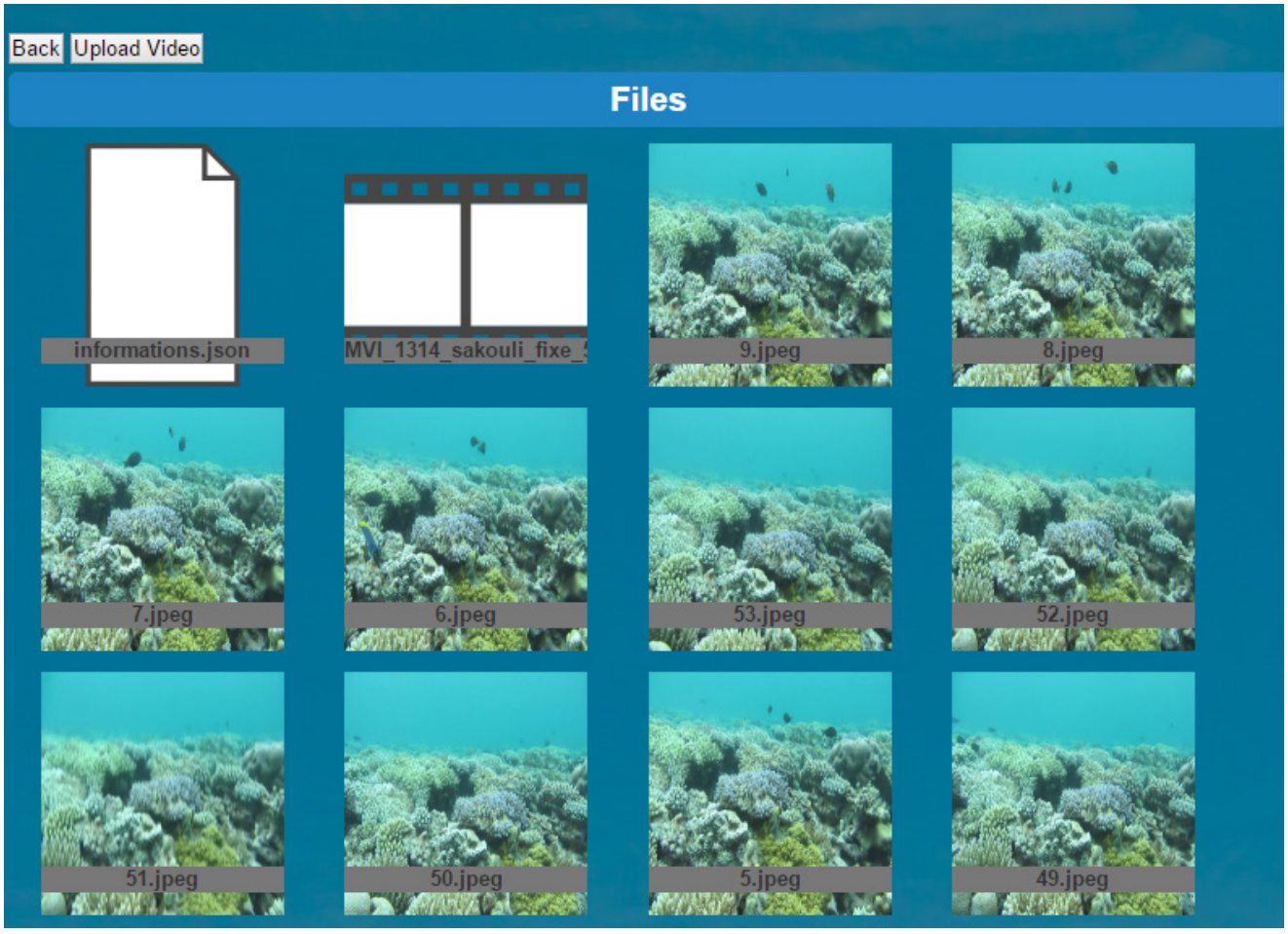
Cette approche apporte un gain de vitesse de traitement pour les images de grand taille car, il n'y a plus besoin de redessiner l'image pour effacer des cadres, l'image est dessinée seulement au changement de frame.

# Application



Pour rendre la recherche et la navigation des images et vidéos plus instinctif je me suis lancé dans la recherche de gestionnaire de fichiers en ligne.
Encore un fois ce n'était pas forcément à jour ou ne convenait pas aux besoins.
C'est donc de ça qu'une interface de navigation très simple est née.



Les icônes présentes sont des images créées pour l'occasion.

(Aucun droit d'auteur)

Si le fichier est une image celle-ci sera affichée.

Cliquer sur une image amène directement sur l'interface d'annotation.

C'est donc aussi avec la création de cette interface que la gestion de multiples fichiers a été implantée.

Pour utiliser l'interface il faut se créer un compte.
Une fois le compte créé celui-ci doit être validé pour pouvoir annoter.
Par défaut on peut se connecter mais l'on ne peut rien faire.
Un administrateur (ou modérateur) doit pour cela changer le rang de la personne.
Quatre rangs existent :

- 0 : Non validé (par défaut)
- 1 : Annotateur
- 2 : Modérateur
- 3 : Administrateur

Un annotateur ne peut qu'annoter.
Un modérateur peut gérer les utilisateurs, uploader des vidéos, ajouter de nouvelles espèces.
L'administrateur lui a presque les mêmes droits que le modérateur. A ceci près qu'un modérateur ne peut pas changer le rang d'un administrateur alors que l'inverse est possible.

La gestion des espèces fonctionne en trois parties :

- Famille
- Genre
- Espèce

Une espèce appartient à un genre qui lui-même appartient à une famille.

Une espèce peut aussi être définie comme adulte ou juvénile, mâle ou femelle. En effet, il existe des dismorphismes pour certaines espèces selon ses critères.

D'autre part il est possible que certaines espèces soit difficilement distinguables entre elles car visuellement très proches. Dans ces cas-là, on ne choisit que la famille où est définit l'id avec les deux noms.

Autre remarque, lors de l'ajout d'une espèce, si l'on a bien choisi le genre, une vérification de son existence ainsi que la récupération d'un numéro unique est faite à partir de la base mondiale de reférence du site WoRMS : http://www.marinespecies.org/

Ce site fournit un semblant d'API pour la récupération. Le nom des espèces est en latin, car le nom peut changer d'un pays à l'autre, voire d'une île à l'autre dans certains archipels.

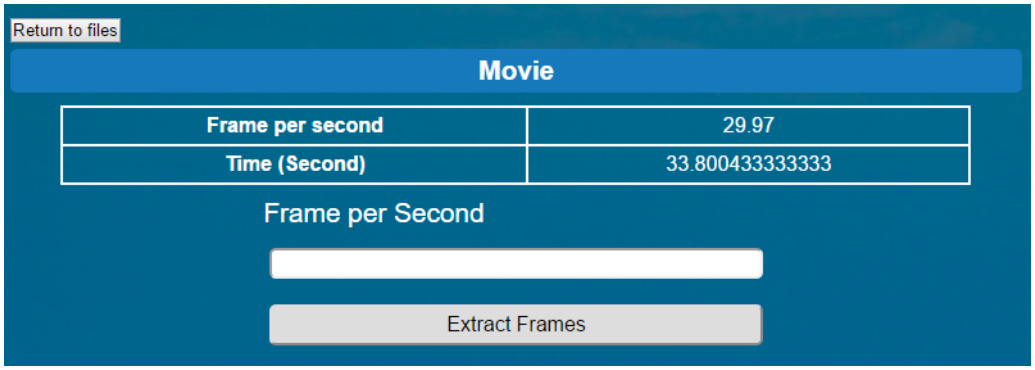
Parfois la recherche retourne -999, ce qui signifie que l'espèce existe en plusieurs fois dans la base. Dans ces cas-là il faut choisir à la main.

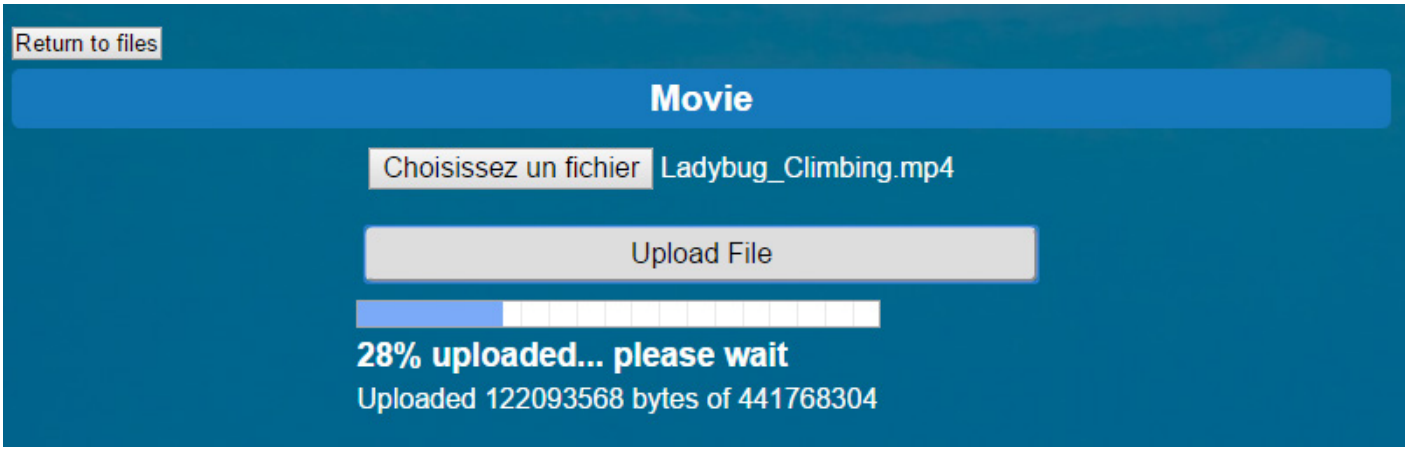
L’upload et la découpe des vidéos.
L’upload des vidéos se fait via un formulaire, il accepte jusqu’à 6Go mais peut-être réglable au besoin.

La découpe est une partie bien plus lourde, tant en ressources qu’en gestion.
Pour découper une vidéo, le logiciel ffmpeg est utilisé.
Il permet entre autre d’extraire des images d’une vidéo par ligne de commande. Cette étape prend du temps.
La découpe se déroule dans un dossier projet.

A l’origine, seul un dossier par vidéo était utilisé mais après discussion c’est un dossier par découpe qui a été privilégié.
Exemple de nom de dossier : GOPR10553_clement_1fps .
Sur une vidéo de 30 secondes, à une image extraite par seconde cela ne fait que 32 images (les deux de plus sont la première image et la dernière)

Mais sur une vidéo de 26 min découpée à 2fps cela devient vite énorme, c’est donc cela qui prend du temps. En effet, on atteint dans se cas environs 3120 images.






L'un des fichiers principaux : Informations.json

Ce fichier contient le nombre d'images qui ont été extraites, le nom de la vidéo et peut contenir toutes sortes d'informations utiles.

Voici un exemple des informations affichées si l'on sélectionne un fichier informations.json

| Name: | MVI_0606.MP4 |
|---|---|
| Frame per second (video): | 25 |
| Time (Second): | 17.84 |
| Extraction frequency (n/sec) | 2 |
| NbFrame extract: | 37 |
| Demo | No |

Enable as a test

Get simplified files

| Id species | Number of annotation |
|---|---|
| Dascyllus_trimaculatus_juv | 338 |
| Archamia_fucata | 97 |
| Thalassoma_lunare | 44 |
| Dascyllus_trimaculatus | 11 |
| Amphiprion_akallopisos | 65 |
| Caesio_varilineata | 1 |
| Chromis_nigroanalis | 2 |
| unknown_fish | 1 |
| Total | 559 |

Il est possible de passer une vidéo comme vidéo de test.

C'est-à-dire que la vidéo peut être annotée avec un fichier d'annotation par utilisateur.

Ce système peut être utilisé dans le cadre d'évaluation de compétence, car bien que l'annotation soit individuelle, un modérateur peut voir et modifier ces annotations mais aussi comparer avec les annotations de référence (toujours modifiables par un modérateur) qui sont toujours conservées.

Côté serveur, dans un premier temps les images n'étaient pas rangées.

(Pas de fichier par vidéo)

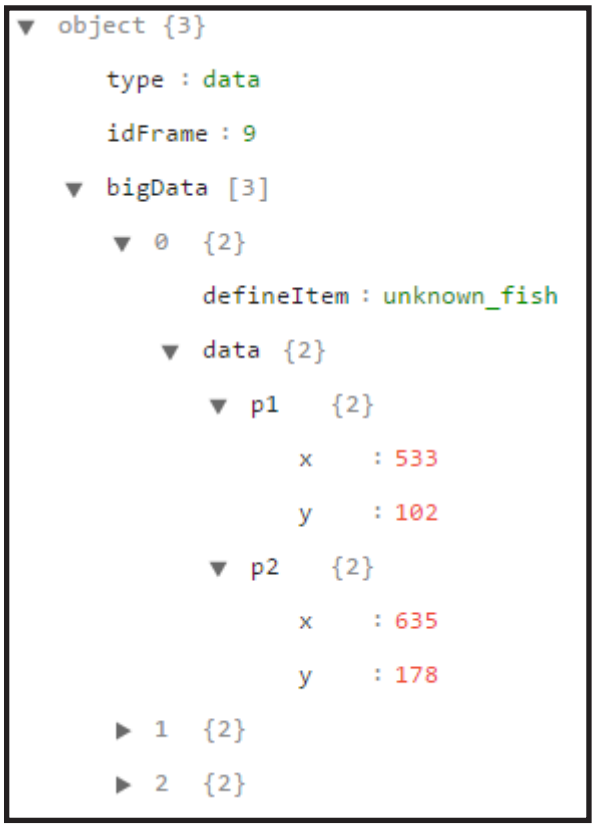
Les informations d'une frame (image prise à un temps T d'une vidéo) sont stockées au format JSON dans un fichier .json par frame.

Ce fichier contient :

- Numéro de la frame.
- p1 (Point 1 qui est le point en haut à gauche)
- p2 (Point en bas à droite)
- defineItem : Ici le nom unique de l'espèce.

Le JSON encore une fois permet un traitement avec un grande simplicité.

{«type»:»da
ta»,»idFrame»:»9»,»bigDa
ta»:[{«defineItem»:»unknown_fi
sh»,»data»:{«p1»:{«x»:533,»y»:102}
,»p2»:{«x»:635,»y»:178}}},{«definel
tem»:»unknown_fish»,»
data»:{«p1»:{«x»:699,»y»:62},»p2»:
{«x»:754,»y»:104}}},{«de
fineItem»:»unknown_fish»,»data»:{«
p1»:{«x»:887,»y»:90},»p2»:{«x»:100
6,»y»:192}}}]}

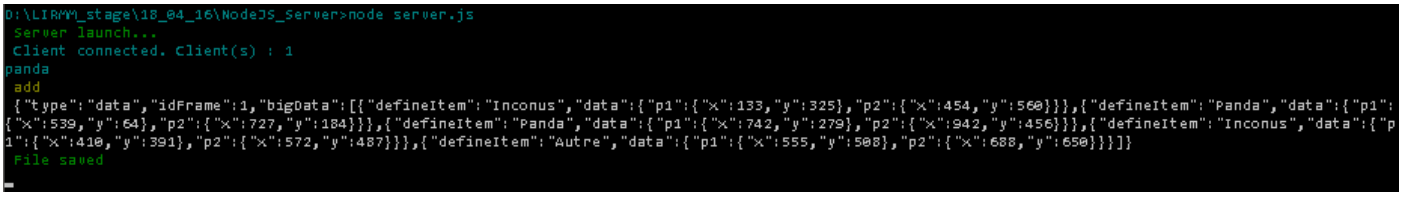



Cette façon de sauvegarder les informations évoluera un peu par la suite.

Comme on peut le voir, de cette façon le format n'est pas très lisible.

Une option viendra simplifier les fichiers pour la lecture.

De plus le serveur sauvegarde toutes les logs d'action. Que ce soit la connexion d'un client, un encadrement, ou un changement de frame.




Capture d'écran de la console serveur

La sauvegarde des actions se fait dans un fichier de log journalier.

```
10:29:26 : Valid Token
10:29:26 : /var/www/html/videos/MVI_1314_sakouli_fixe_51s/10.jsonSaved
10:35:17 : Client disconnect. Client(s) : 0
10:35:33 : Client connected. Client(s) : 1
10:35:33 : Valid Token
10:35:33 : Valid Token
10:35:33 : changeFolder : /MVI_1314_sakouli_fixe_51s
10:35:33 : /var/www/html/videos/MVI_1314_sakouli_fixe_51s/7.json is create
11:00:03 : changeFrame : 8
11:00:03 : Valid Token
11:00:03 : /var/www/html/videos/MVI_1314_sakouli_fixe_51s/8.json is create
11:00:03 : Valid Token
11:00:03 : changeFrame : 9
```

Extrait d'un fichier de log

Les Tokens :

On remarque rapidement dans le fichier de log le retour régulier d'un « Valid Token »

Les Tokens (Jeton) sont des numéros uniques attribués à l'utilisateur lors du chargement de la page d'annotation.
Ces jetons indiquent au serveur d'annotation que l'utilisateur est autorisé à annoter. Il permet de connaître l'identité de la personne qui se connecte, mais aussi de limiter sa connexion.
Un jeton est valide 5h mais ce temps peut être changé en cas de besoin.
Au-delà le jeton est considéré comme périmé et l'accès au serveur d'annotation est refusé. Sans connection au site et sans les droits minimums aucun jeton n'est donné.

Une partie de mes recherches se sont tournées vers la possibilité de crypter les données échangées avec le client.

Mais la majeure partie des modules trouvés étaient obsolètes ou non adaptés au projet. Ces recherches n'ont abouti qu'à peu de résultats et le projet a été laissé de côté car non critique en réseau local.

Pour revenir sur le stockage des données, le changement apporté fut l'arrivée d'un identifiant unique par rectangle.

Cet identifiant permet la suppression et la modification ciblée.

Comme il n'existe pas de fonction de base pour en générer, c'est de la façon suivante qu'est généré cet unique id (Identifiant unique).

Date du jour en millisecondes suivi de 5 caractères aléatoires entre

a-z A-Z et 0-9

La probabilité d'avoir le même étant extrêmement faible, cela offre pour l'utilisation, une certitude de ne pas avoir deux fois le même.

Par la suite, le nom des utilisateurs à été ajouté aux logs pour chaque action de leur part. Avec cela, nous pouvons garder une traçabilité des actions.

Dans les derniers jours ce sont des corrections d'erreurs et une surveillance du système qui ont été mis en place.

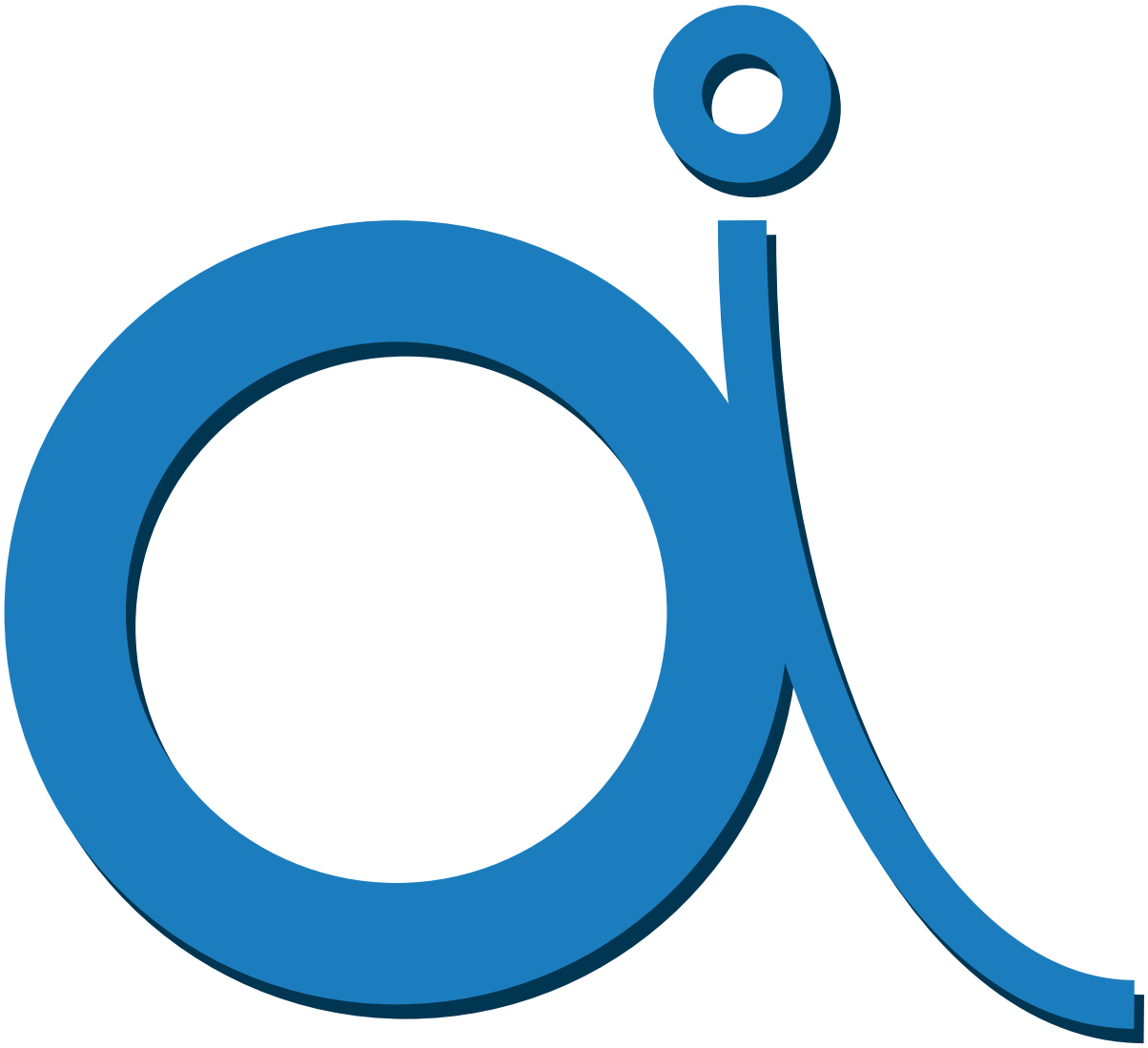
Un exemple d'erreur était la corruption de fichiers contenant les informations d'une frame. Cela supprimait tous le contenu et retournait des erreur à leur lecture.

L'ajout d'un démarrage automatique et redémarrage en cas d'erreur.

Il arrive certaines fois que des informations soit mal transmises et provoquent un crash serveur.

Avec un script de redémarrage en cas d'erreur, cela devient presque invisible au vu de la vitesse de démarrage.

Un logo pour l’application :
Trouvant un peu vide l’interface et ayant une idée en tête je me suis lancé dans la création d’un logo.



Voir dans les lettres OIA pour « Online Image Annotation » sous une forme de poissons très stylisé.

Voilà pour un tour rapide de l’application ainsi que quelques réflexions plus approfondies.

Dans ce stage il m’a été demandé de présenter mon projet a l’équipe lors d’une réunion.

Ma présentation s’est déroulée le mercredi 4 mai.
De bon retours m’ont été donnés sur cette présentation et les échanges qui ont eu lieu après celle-ci m’ont permis de réorganiser certaines parties.

# Conclusion

# Conclusion

Le stage s'est très bien déroulé, de très bons contacts avec les deux équipes (LIRMM et MARBEC).

Il y avait toujours une personne présente en cas de besoin que ce soit un encadrant, une personne de chez MARBEC si une question me venait à l'esprit ou même un autre stagiaire présent dans la salle.

J'ai eu un bon cadre de travail et de bonnes conditions qui ont permis de bonnes avancées et la sortie d'une application fonctionnelle dans la durée du déroulement du stage et ceci en partant de zéro.

J'ai apprecié la liberté permise pour la création de l'application bien qu'un cahier des charges précis ait été donné. Certaines parties n'étaient pas ergonomiques et j'ai pu donner mon avis et faire des propositions sans aucun problème.

Les deux dernières semaines se sont déroulées directement chez MARBEC pour la mise en place et le contact avec les stagiaires en charge des annotations. Cela m'a permis entre autres de corriger les plus gros problèmes et de voir comment se comportait l'application.

Comme tout code n'est pas exempté de bogue et que la personne qui programme ne peut pas tout tester, j'ai pu voir mes erreurs et corriger les plus importantes.

Pour finir sur un avis global de ce stage : Je suis très satisfait et prêt à revenir pour mes prochaines années d'études.

Remarque : A l'issue de ce stage un contrat d'été m'a été proposé par l'équipe MARBEC pour un autre projet. L'avantage est qu'en cas de besoin je pourrais toujours venir corriger l'application d'annotation.

# Glossaire

**API** : Interface de programmation applicative (Application Programming Interface).

**ffmpeg** : Logiciel de traitement vidéo par ligne de commande.

**HTML** : Hypertext Markup Language, langage utilisé pour la sémantique des pages internet (Ex : Hello World le p définit que ceci est un paragraphe)

**MySQL** : Serveur de base de données.

**Package** : Paquet / module complémentaire.

**PHP** : Langage de programmation généralement utilisé pour générer des pages internet côté serveur.

**Objet** : Entité contenant des informations ainsi que des fonctions pour son utilisation. Exemple : un cadre contient deux points qui contiennent eux même une valeur x et y. Un cadre peut-être dessiné.

**Réseau local** : Réseau à portée limitée localement. (Ex : Réseau domestique)

**WebSocket** : Standard du Web désignant un protocole réseau de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par-dessus une connexion TCP.