



Projet de Génie Logiciel :

Élaboration d'une interface permettant l'utilisation centralisée  
de multiples outils issus des nouvelles générations de  
séquençage.

---

## RAPPORT

---

Auteurs :

**DENET Lola**  
**DESQUERRE Émilie**  
**HUI Tongyuxuan**  
**MEGUERDITCHIAN Caroline**  
**NIU Wenli**  
**PRATX Julie**  
**VU Thao Uyen**

Étudiants en Master 2 de Bio-informatique  
Université de Bordeaux

14 Décembre 2020

# Table des matières

<b>Introduction</b>	<b>1</b>
<b>1 Analyse</b>	<b>2</b>
1.1 Contexte . . . . .	2
1.2 État de l'art . . . . .	2
1.2.1 État de l'art biologique . . . . .	2
1.2.2 État de l'art bioinformatique . . . . .	4
1.3 Analyse des besoins . . . . .	8
1.3.1 Besoins fonctionnels . . . . .	8
1.3.2 Besoins non fonctionnels . . . . .	8
1.4 Objectifs . . . . .	9
<b>2 Conception</b>	<b>10</b>
2.1 Méthodes . . . . .	10
2.2 Outils . . . . .	11
2.2.1 Langages de programmation . . . . .	11
2.2.2 Modules/ <i>Frameworks</i> /Outil de développement . . . . .	11
2.3 Prototype . . . . .	12
2.4 Architecture logicielle . . . . .	15
<b>3 Réalisation</b>	<b>17</b>
3.1 Environnement de travail . . . . .	17
3.1.1 Serveur local . . . . .	17
3.1.2 Environnement virtuel . . . . .	17
3.1.3 Dépôt et mise en ligne . . . . .	17
3.2 Implémentation . . . . .	18
3.2.1 Développement du <i>pipeline</i> . . . . .	18
3.2.2 Développement de l'interface utilisateur . . . . .	19
3.2.3 Intégration . . . . .	19
3.3 Résultats . . . . .	20
3.4 Limites et perspectives . . . . .	24
<b>Conclusion</b>	<b>25</b>
<b>Références</b>	<b>26</b>
<b>Appendices</b>	<b>I</b>
<b>A Guide utilisateur</b>	<b>I</b>
A.1 Installation, configuration et démarrage du serveur Flask . . . . .	I
A.2 Utilisation du site en ligne . . . . .	I
A.2.1 Accueil . . . . .	I
A.2.2 Infos générales . . . . .	II
A.2.3 Séquences . . . . .	II
A.2.4 Alignement . . . . .	III
A.2.5 Phylogénie . . . . .	IV

# Introduction

Ce projet s'inspire du programme "Les sentinelles du climat" porté en Nouvelle Aquitaine<sup>1</sup>. Ce programme a pour but l'étude des changements climatiques et de la biodiversité dans différents milieux. Pour chaque milieu, des espèces sont étudiées durant 6 ans afin d'évaluer l'impact des changements climatiques sur la faune et la flore. Ces espèces sont appelées les sentinelles du climat. L'étude de leurs comportements et répartitions dans les différentes zones permettra l'identification des espèces susceptibles de s'adapter, de migrer ou de disparaître et ainsi d'en mesurer l'impact sur l'environnement et la biodiversité de la région<sup>2</sup>.

La Nouvelle Aquitaine est particulièrement impactée par le réchauffement climatique, en grande partie à cause des kilomètres de côtes qui la composent. D'autre part, la région bénéficie d'une diversité importante de milieux : les dunes atlantiques, les hêtraies de plaine, les zones humides, les pelouses calciales, les torrents des montagnes, les pelouses et rocailles des montagnes. Cette diversité de milieux implique une diversité d'espèces permettant une étude approfondie de l'impact des changements climatiques sur la biodiversité.

Les zones humides<sup>3</sup> ont une importance toute particulière car elles abritent une plus grande diversité de sentinelles, c'est-à-dire une plus grande quantité d'espèces susceptibles d'être impactées par les changements climatiques car ces zones sont plus fraîches. Ainsi, l'augmentation des températures dans ces zones pourrait avoir des effets catastrophiques sur les espèces qui y vivent car elles pourraient avoir plus de difficultés à s'adapter ou à trouver un nouvel habitat propice à leur développement et leur survie.

Six espèces ont été identifiées comme sentinelles du climat dans les zones humides dont une qui a été ajoutée très récemment : la vipère péliade. La libellule, la rainette ibérique et rainette verte, l'azurée des mouillères ainsi que le lézard vivipare sont les autres sentinelles identifiées dans ce milieu<sup>4</sup>.

Compte tenu de ce contexte, il est proposé de développer une interface utilisateur qui combinerait divers outils de bioinformatique afin d'étudier une de ces espèces sentinelles.

Dans ce travail, une analyse du sujet sera tout d'abord réalisée à travers un état de l'art biologique et bioinformatique ce qui permettra d'aboutir à l'identification précise des objectifs de ce travail. Une analyse des besoins fonctionnels et non fonctionnels sera également réalisée afin de préciser les attentes potentielles concernant cette problématique. Dans un second temps, le processus de conception sera abordé et permettra de présenter l'architecture logicielle notamment. Enfin, la réalisation sera explicitée. Le développement ainsi que les résultats obtenus seront alors mis en avant.

Le code source de ce travail est accessible sur GitHub<sup>5</sup>. Il est également possible d'accéder à la plateforme en ligne issue de ce projet<sup>6</sup>.

---

1. <https://www.sentinelles-climat.org>

2. <https://www.sentinelles-climat.org/a-propos/>

3. <https://www.sentinelles-climat.org/milieu/zones-humides/>

4. <https://technique.sentinelles-climat.org/developpement-dindicateurs-sentinelles-climat/>

5. <https://github.com/les-sentinelles-de-la-bioinfo/ProjetPhylogenie>

6. <http://sentinellesbioinfo.pythonanywhere.com>

# Chapitre 1

## Analyse

### 1.1 Contexte

Ce projet s'inclut dans le contexte de recherche précédemment présenté et propose d'étudier la vipère péliade. Cette espèce a été identifiée très récemment en tant que sentinelle des zones humides car elle est difficile à trouver. Elle est susceptible d'être particulièrement impactée par le réchauffement climatique car sa capacité à s'adapter aux chaleurs en raison de sa physiologie serait limitée. De plus, elle ne se déplace que peu chaque année ce qui pourrait être un frein pour une éventuelle migration. C'est une espèce protégée susceptible de s'éteindre dans la région. L'étude menée par le programme "Les sentinelles du climat" va permettre de vérifier ces hypothèses et de prévoir leur impact sur la biodiversité.

Son inclusion récente dans l'étude et les aspects présentés précédemment en font un sujet d'étude pertinent pour ce projet. En effet, peu d'informations ont été recueillies jusqu'à ce jour dans le cadre de cette étude. Il paraît alors intéressant de collecter des informations sur cette espèce et de créer un outil qui permettra d'en faciliter le traitement.

Ce travail a pour but de créer une interface qui permettrait de centraliser plusieurs outils issus des nouvelles générations de séquençage. En effet, les recherches menées sur la vipère péliade pourront permettre d'identifier un groupe de gènes particulièrement étudié par la communauté scientifique. Un autre aspect de ce travail consistera en l'identification d'outils pertinents pour l'étude de ce groupe de gènes afin d'aboutir à l'élaboration d'un arbre phylogénétique.

La bioinformatique est un atout primordial dans ce projet car elle permet de faciliter le travail des chercheurs par la centralisation d'outils complexes sur une même plateforme. En effet, l'une des difficultés de la recherche actuelle est d'identifier les outils les plus pertinents pour un domaine particulier. La multiplicité des plateformes qui ralentit l'aboutissement au résultat est un autre exemple de difficulté rencontrée par les chercheurs. Il paraît essentiel de pouvoir centraliser tout un *pipeline* en un même outil afin de rendre plus efficace l'étude du groupe de gène identifié. De plus, la création de cette interface pourra permettre dans le même temps de collecter des informations sur l'une des espèces sentinelle du climat : la vipère péliade.

### 1.2 État de l'art

#### 1.2.1 État de l'art biologique

##### Présentation de l'espèce

*Vipera berus*, la vipère péliade représentée dans la figure 1.1<sup>1</sup>, est une espèce de serpent venimeux de la famille des *Viperidae*. Son habitat se trouve en Europe et en Asie, dans les zones humides, les tourbières et les forêts. Selon la zone dans laquelle se trouve l'individu, la coloration varie du gris, bleu-gris, marron, vert-brun au noir.

Cette espèce se nourrit de grenouilles ainsi que de petits rongeurs. Les plus jeunes mangent aussi des lézards. Ces derniers ainsi que les grenouilles sont également des espèces sentinelles. La période d'activité

---

1. <https://biodiversite.parc-naturel-pilat.fr/espece/78141>



FIGURE 1.1 – Vipère péliade

de la vipère diffère en fonction de l'altitude et du climat (de mars-mai à septembre-novembre)[Losange, 2008].

Un bocage dégradé, une biologie exigeante, le réchauffement climatique, l'impopularité, tout cela fait que la vipère péliade est, aujourd'hui, une espèce menacée vulnérable (selon l'union internationale pour la conservation de la nature) [CREUX, 2019].

Il est intéressant d'étudier cette espèce pour les différentes propriétés médicales de son venin mais aussi pour sa capacité à réguler les espèces sentinelles déjà introduites. En effet, le venin de *Vipera berus* a principalement une activité hémotoxique. Les hémotoxines peuvent être classées en fonction de leur effet sur l'activation de facteurs de coagulation sanguine, sur des agents anticoagulants, sur des inhibiteurs ou activateurs de plaquettes, sur des agents affectant la fibrinolyse ainsi que sur les hémorragines [Phillips *et al.*, 2010].

La vipère péliade est un serpent des plus dangereux à cause de son venin mortel chez l'Homme si la prise en charge n'est pas rapide. Cependant, celui-ci est composé de nombreux composants protéiques qui sont actuellement testés pour leur utilité dans le traitement de nombreuses maladies (neurologiques, cardiovasculaires, cancers) [Bocian *et al.*, 2016] ainsi que pour la production de sérum antivenimeux pour le traitement des morsures [Netgen, 2011].

D'après Al-Shekhadat *et al.*, *Vipera berus* est une espèce médicalement importante [Al-Shekhadat *et al.*, 2019]. L'équipe scientifique de Malina *et al.* s'est intéressée aux types de dégradation des protéines et ceci pour des individus d'âge et de sexe différents, mais aussi, sur l'effet paralysant du venin. Les échantillons de venin avaient des effets neuromusculaires paralysants variables [Malina *et al.*, 2017].

Une étude réalisée par l'équipe de Czajka *et al.*, traite des situations dans lesquelles l'administration d'antivenin est nécessaire pour neutraliser les propriétés toxiques du venin et ses effets indésirables [Czajka *et al.*, 2013]. L'antivenin ViperaTAB a été utilisé par l'équipe de Hamilton *et al.* car les morsures sont une urgence médicale rare au Royaume-Uni, avec 20 à 50% des 50 à 200 cas estimés par an, nécessitant un traitement avec un antivenin. Ainsi ils ont démontré que lorsque les effets sont systémiques et locaux, ils nécessitent une réanimation hors hôpital, un soutien vasopresseur et une rééducation prolongée [Hamilton *et al.*, 2019].

L'étude de Hermansen *et al.* a pour but de présenter une vaste série de cas consécutifs de patients mordus par *Vipera berus*, et d'identifier les signes et symptômes indiquant une maladie compliquée[Hermansen *et al.*, 2019]. Malina *et al.* ont étudié les effets que provoque une morsure chez l'Homme. Le patient a développé des troubles nerveux crâniens sans ambiguïté, se manifestant par une atteinte bilatérale caractérisée par une paralysie oculomotrice avec ptose partielle (paupière qui tombe), parésie du regard et diplopie. La somnolence et la photophobie étaient ses symptômes supplémentaires [Malina *et al.*, 2013].

### Composition protéique du venin

L'étude, qui a été réalisée par Bocian *et al.*, a permis de recueillir du venin à partir d'individus mâles et femelles adultes [Bocian *et al.*, 2016]. Tandis que celle de Al-Shekhadat *et al.* a étudié les activités toxiques et enzymatiques et a déterminé la composition protéomique de son venin. Cette étude a égale-

ment été conçue pour évaluer l'efficacité préclinique *in vivo* et *in vitro* de l'antivenin russe Microgen pour neutraliser les principaux effets du venin [Al-Shekhadat *et al.*, 2019].

Guillemin *et al.* ont utilisé une méthode basée sur la PCR pour déterminer les séquences d'ADN génomique codant pour les phospholipases A2 (présentent à 59% [Bocian *et al.*, 2016]) à partir des venins de différentes espèces de vipère dont *Vipera berus*.

Ce travail portera sur ces 3 séquences : AY158636 (phospholipase A2), AY158639 (ammodytin I2), AY159811 (ammodytin I1) où l'ammodytin est une protéine variante de la phospholipase A2 [Guillemin *et al.*, 2003].

## 1.2.2 État de l'art bioinformatique

### Bases de données biologiques

Bien que le séquençage de l'ADN et des protéines ait lieu dans le monde entier, presque toutes les données collectées sont stockées et partagées dans un certain nombre de grandes banques généralistes tels que :

- NCBI (*National Center for Biotechnology Information*<sup>2</sup>) : est un référentiel de ressources médicales et biotechnologiques. Les principales bases de données du NCBI comprennent Genbank (des informations sur les séquences d'ADN) et PubMed (une base de données bibliographiques en biologie et en médecine). Structurellement, Genbank se compose de deux grandes sections distinctes : la base de données de protéines et la base de données de nucléotides, dans laquelle la base de données de nucléotides est utilisée comme un chemin d'accès aux données protéiques respectives.
- EBI (*European Bioinformatics Institute*<sup>3</sup>) : Cette base de données est organisée et gérée dans environ 80 domaines différents, dont le plus important se concentre sur 3 domaines : la base de données des structures d'ADN, la base de données des structures protéiques (TrEMBL et SWISS-PROT) et la base de données des structures macromoléculaires (EBI-MSD).
- DDBJ (*DNA Data Bank of Japan*<sup>4</sup>) : est une base de données sous la gestion de JNIG (*Japan National Institute of Genetics*). Comme Genbank et EMBL, DDBJ recueille des données sur les séquences de nucléotides et fournit des données de séquences de nucléotides et un système de supercalculateur disponibles gratuitement pour soutenir les activités de recherche en sciences de la vie.

Les trois bases de données Genbank (NCBI), EMBL et DDBJ effectuent une connexion directe et échangent quotidiennement des informations, de sorte qu'elles possèdent toutes les informations l'une de l'autre. La collaboration entre ces trois banques les aide par conséquent à se développer et devenir les plus grandes bases de données du monde.

### Avantages et Inconvénients des banques généralistes

#### 1. Avantages :

En général, les banques généralistes offrent de nombreux avantages à la recherche scientifique, en particulier dans le cadre de l'analyse des séquences. Ce sont des outils indispensables à la diffusion rapide des résultats scientifiques. Elles présentent complètement toutes les données, en double publication depuis 1988 mais centralisées en un seul ensemble de données depuis le début des années 2000, cela est important pour la recherche de similitudes avec une nouvelle séquence. D'autre part, la représentation d'une grande diversité d'organismes dans les bases de données généralistes rend possible des analyses phylogénétiques. Un autre bonus de ces bases réside dans l'information qui accompagne les séquences (annotations, expertise, bibliographie) qui peuvent parfois constituer les rares annotations disponibles sur certaines séquences. Enfin, l'inter-référence à d'autres bases de données permet d'avoir accès à d'autres informations non répertoriées.

#### 2. Inconvénients :

Contrairement aux avantages ci-dessus, les bases de données généralistes ont aussi des inconvénients. D'abord, c'est le manque de contrôle des données soumises ou saisies surtout pour les séquences anciennes (pas d'expertise). Ensuite, la variabilité de l'état de connaissances sur les séquences est aussi une lacune des bases de données. La détermination des caractéristiques biologiques et des fonctions des séquences

---

2. <https://www.ncbi.nlm.nih.gov/>

3. <https://www.ebi.ac.uk/>

4. <https://www.ddbj.nig.ac.jp/index-e.html>

demande un travail expérimental et une analyse qui doivent se surajouter à l'étape automatisée et systématique du séquençage. Un autre défaut vient des erreurs dans les séquences. La cause est dérivée de la contamination du fragment original due à la technologie ou à la méthodologie. En outre, le dernier désavantage qu'on ne peut pas ignorer est le biais d'échantillonnage : le biais d'échantillonnage taxonomique (les séquences ont été extraites à partir des organismes qui ont une quantité inégale), le biais d'échantillonnage des séquences (les gènes des génomes étudiés sont inégalement représentés dans chacun d'eux) et la redondance des données (certains gènes extrêmement similaires correspondent à des entrées différentes dans la banque et c'est difficile de savoir si cela concerne au polymorphisme génétique ou à la duplication des gènes ou tout simplement aux erreurs lors de la détermination des séquences).

## Alignement des séquences

Après avoir rassemblé les séquences choisies, il est essentiel de les traiter de façon à pouvoir déterminer leur relation. Pour cela, il faut réaliser un alignement qui servira par la suite à créer des arbres phylogénétiques.

L'alignement de séquences a pour but d'arranger les séquences de façon à mettre en valeur leurs similarités. Cela permet notamment de détecter des relations fonctionnelles ou évolutives entre ces séquences. De nombreux outils existent permettant de générer un alignement de séquences à partir d'un certain nombre de séquences d'ADN.

Il existe deux types de mesure de similarité : l'alignement global et l'alignement local. L'alignement global permet de regarder l'alignement des séquences dans leur entièreté ; il est conçu pour comparer des séquences homologues. Par conséquent il est possible d'y retrouver des zones où une similarité faible sera observée. L'alignement local lui, s'intéresse seulement aux régions relativement bien conservée. Il compare uniquement les régions similaires d'une séquence. Il existe plusieurs outils permettant d'effectuer des alignements locaux.

- Blast (*Basic Local Alignment Search Tool*) est une méthode de recherche heuristique qui se base sur l'alignement local de séquence par paires. Il permet la recherche dans une banque de données de séquences similaires. Son fonctionnement est le suivant : il décompose tout d'abord la séquence en segments (appelé k-uplets) qui se chevauchent. Puis chacun de ces k-uplets est comparé aux séquences cibles définies par l'utilisateur. Quand une homologie est détectée pour un segment, Blast étend l'alignement en amont et en aval pour chercher une zone de similarité plus étendue. Une fois cette étape effectuée, il évalue la probabilité que la similarité soit due au hasard (*E-value*) [Altschul *et al.*, 1990].
- Il existe également des outils tels que Water et SSearch, qui se basent sur l'algorithme Smith-Waterman. Cet algorithme repose sur l'utilisation de matrice de similarité. Il cherche à optimiser l'alignement de deux séquences en insérant des indel (insertion ou délétion) afin de maximiser le nombre de nucléotides positionnés de manière identique dans les deux séquences. Un score permet ensuite de quantifier le résultat selon le nombre de nucléotides identiques et le nombre de substitution [Olsen *et al.*, 1999].

Il existe également de nombreux outils permettant d'effectuer des alignements globaux.

- Certains d'entre eux se basent sur la méthode de l'alignement progressif, qui utilisent le regroupement d'alignements deux à deux pour construire un alignement multiple. C'est le cas par exemple de T-Coffee, MAFFT ou Pileup. Le programme le plus populaire utilisant cette méthode est Clustal Omega. Son algorithme possède 5 étapes : tout d'abord, les séquences sont alignées deux à deux en les segmentant comme vu précédemment avec l'algorithme Blast. Ensuite, les distances entre ces séquences sont calculées. Puis un algorithme de clustering (*k-means*) est utilisé afin de regrouper les séquences. Une matrice de distance est calculé pour chaque *cluster*. Grâce à cet matrice, un arbre guide est construit en utilisant la méthode UPGMA : à chaque étape les deux *clusters* les plus proches sont combinés jusqu'à ce qu'on obtienne l'arbre final. Cet arbre déterminera l'ordre dans lequel les séquences sont alignées. Enfin, les séquences sont alignées grâce au *package* HHAlign. [Sievers et Higgins, 2014].
- Il existe également un outil appelé Muscle qui se base sur un algorithme itératif. A la première étape, un alignement progressif est construit en utilisant un arbre guide, construit en mesurant la similarité des séquences deux à deux puis en estimant leur distance. Ensuite, la deuxième étape améliore l'arbre construit en réalisant un nouvel alignement progressif basé sur l'arbre. Enfin, dans une troisième étape, l'alignement est à nouveau affiné en réalisant encore un nouvel alignement qui sera gardé si son score est supérieur au précédent. La troisième étape est répétée plusieurs fois. [Edgar, 2004].

Dans le contexte de ce projet, il sera nécessaire de réaliser des alignements d'espèces proches, afin de situer la vipère péliade par rapport à d'autres espèces d'intérêt. Il est donc préférable de choisir une méthode d'alignement global. Des alignements multiples devront être réalisés puisqu'il s'agit de comparer plusieurs espèces entre elles. Ainsi, Clustal Omega ou Muscle, des programmes d'alignement multiple global semblent les plus adaptés pour réaliser cette tâche.

## Phylogénie

Après avoir sélectionné, nettoyé puis aligné les séquences choisies, une étude phylogénétique approfondie d'un gène cible sera effectuée. Celle-ci se fera sur 2 niveaux différents : une plus générale qui servira à situer l'espèce dans le monde du vivant et une plus spécifique sur le plan génique. Cette étude phylogénétique aboutira donc à la réalisation de minimum 2 arbres phylogénétiques :

- Un arbre des espèces : celui-ci représente les relations évolutives entre les espèces en général, celui-ci peut-être déduit à partir de molécules, toutefois cela reste risqué, car de nombreux biais existent. Il y a en effet plus de risques d'obtenir des paralogies (duplication d'un gène entre 2 espèces différentes ou un transfert horizontal de gènes). C'est pour cela que des données génomiques plus larges seront utilisées pour les construire. Celui-ci permettra ainsi de situer la vipère péliade par rapport à d'autres espèces d'intérêts notamment les 6 espèces sélectionnées dans le programme « Les sentinelles du climat » et donner une idée générale de ses relations évolutives.
- Un arbre des gènes : celui-ci représente l'histoire évolutive des molécules apparentées (gènes, protéines etc ... par exemple). Cette représentation peut différer de manière plus ou moins importantes de l'arbre des espèces. Pour cela une sélection sur des gènes d'intérêt sera effectuée

Un arbre phylogénétique est composé de plusieurs éléments : des OTUs aussi appelés feuillettes qui représentent les unités taxonomiques opérationnelles (les espèces/molécules choisies), des HTUs ou nœuds internes qui représentent les unités taxonomiques hypothétiques (un ancêtre hypothétique commun) et finalement des branches représentant les liens de parenté entre les unités. Tous ces éléments sont organisés de façon à former des branchements ce qui correspond à la topologie de l'arbre. Celui-ci peut aussi être enraciné ou non, même si un arbre non-enraciné ne correspond pas réellement à un arbre phylogénétique puisqu'il ne met pas en avant les relations entre les OTUs. La racine, elle, symbolise le dernier ancêtre commun de toutes les OTUs.

Actuellement, de nombreux outils bio-informatiques ont été développés afin de faciliter la création de ces arbres. Des *softwares* gratuits ou encore de nombreux *webserver* sont actuellement disponibles pour faire ces tâches. Chacun de ces outils se basent sur des algorithmes tirés de méthodes de construction pré-existantes :

- méthodes cladistiques : elles se basent sur une étude des états de caractères avec par exemple le maximum de parcimonie.
- méthodes de distances : basées sur des mesures de distances pour cela l'alignement multiple est utilisé conjointement avec le calcul d'une matrice de distance entre chaque paire de séquences. Ici, l'horloge évolutive est prise en compte et l'arbre peut donc être enraciné ou non. On a plusieurs méthodes très utilisées telles que : UPGMA et le NJ (*neighbor joining*).
- méthodes statistiques : elles comprennent à la fois l'étude des états de caractères et les distances. Il y a par exemple : le maximum de vraisemblance.

### Maximum de parcimonie :

C'est donc une méthode dite cladistique, elle se base sur l'étude des états de caractère. Elle consiste principalement à l'identification de la topologie d'arbres impliquant le plus petit nombre de changements évolutifs pour rendre compte des différences que l'on peut observer entre les OTUs étudiés. Pour cela, il va falloir construire tous les arbres possibles, pour chaque caractère le nombre l'arbre qui en demande le moins est conservé.

Avec cette méthode, il n'y a pas de bonnes solutions à proprement parler, puisque plusieurs arbres sont possibles avec un nombre de substitutions identiques. De plus, la longueur des branches ne tient pas compte de la distance évolutive. L'arbre obtenu est non-enraciné.



Celle-ci présente donc plusieurs désavantages :

- Le nombre d'arbres augmentent de manière exponentielle avec le nombre d'OTUs.
- Pas de prise en compte de l'horloge moléculaire = pas de données évolutives.
- Ne fonctionne qu'avec des régions/protéines très conservées.

Cette méthode ne sera donc pas utilisée pour la construction de l'arbre des gènes, mais pourra être utilisée pour celui des espèces.

Outils bio-informatique disponibles :

#### **UPGMA (*Unweighted Pair-Group method by arithmetic averaging.*)**

Cette technique se base sur un principe de distance, il va y avoir un regroupement des séquences par ordre de similarité. L'arbre obtenu sera enraciné et tiendra donc compte de l'horloge évolutive. IL possèdera un ancêtre hypothétique commun à toutes les espèces. Cette technique nécessite un alignement multiple des séquences, il y aura ensuite construction d'une matrice de distance par itérations successives et celle-ci sera utilisée pour construire l'arbre correspondant. Sa construction se fait des feuilles vers les noeuds.

Cette technique possède des avantages :

- Rapide à mettre en place et rapidité dans l'obtention des résultats.
- Simple à comprendre et à mettre en œuvre.

Mais aussi de nombreux désavantages :

- Égalité des taux d'évolution entre les lignées (= horloge moléculaire biaisée).
- Distance évolutive sous-estimée.
- Les branches longues avec une évolution rapide sont considérées comme des groupes extérieurs.

C'est pour cela que cette technique est assez peu utilisée actuellement.

#### **Neighbor Joining.**

Cette méthode se base sur le même principe que la méthode UPGMA et se construit à partir de matrices de distance. Cependant, contrairement à celle-ci, la méthode NJ tient compte du biais des différentes vitesses d'évolution entre les différentes branches de l'arbre en conservant l'additivité des distances. L'arbre obtenu sera, alors non-enraciné. Il ne mettra donc pas en avant les similarités globales entre les différentes espèces. Ce n'est pas une méthode dite phénéticiste : celui-ci reflétera leurs relations de parenté. Cette technique est particulièrement utilisée pour la construction d'arbre des gènes, toutefois il requiert de connaître la distance entre chaque OTUs.

Avantages :

- rapide.
- simple à mettre en place.
- permet de travailler sur un grand nombre de taxa en même temps.

Désavantages :

- Manque de précision.
- Ne fonctionne qu'avec des séquences dont la distance évolutive est connue.

Cette méthode pourrait être utilisée lors du projet.

Les outils bio-informatiques disponibles sont SYLVA, T-Rex.

#### **Maximum de vraisemblance.**

Cette méthode est dite statistique et va permettre de calculer à partir d'un échantillon observé, la ou les meilleures valeurs d'un paramètre en suivant une loi de probabilité. Ici, il va y avoir une sélection de l'arbre qui maximise la vraisemblance (celui avec la plus forte probabilité de retrouver les données utilisées). C'est principalement utilisé lorsque les taux de changement sont très élevés entre 2 séquences. Chaque base ou AA (acides aminés) des séquences va être considéré séparément. Un *log fit* de vraisemblance est alors calculé pour une topologie donnée en utilisant un modèle de probabilité. Ce *log* est alors

cumulé sur tous les sites et la somme maximisée pour estimer la longueur de branche de l'arbre. Ce processus est utilisé pour toutes les topologies possibles et seule celle ayant la plus haute vraisemblance est conservée.

Cette méthode est particulièrement efficace et possède de nombreux avantages :

- estimation de la longueur des branches.
- permet de différencier les transitions et transversions.

Inconvénients :

- calculs très longs.

Les outils bio-informatiques disponibles sont PhyML, RAxML.

## 1.3 Analyse des besoins

### 1.3.1 Besoins fonctionnels

L'analyse réalisée précédemment permet de comprendre l'intérêt d'étudier le venin de la vipère péliade et principalement les séquences impliquées dans la production de phospholipase A2 (PLA2) (principal constituant du venin). La phylogénie en ressort comme un élément important à étudier.

Compte tenu de ces constats, ce projet devra permettre à l'utilisateur d'obtenir les séquences, de les traiter et de produire un arbre des espèces ainsi qu'un arbre phylogénétique. Ces éléments constitueront la base de l'analyse des besoins fonctionnels.

L'utilisateur devra avoir accès aux différents outils depuis un seul environnement facilement utilisable. Une interface web semble être la méthode la plus appropriée pour faire le lien entre les outils issus de *webservers*.

Cette interface devra permettre à l'utilisateur :

- d'accéder à une page principale qui présentera le contexte et expliquera les différentes étapes du *pipeline* ;
- d'accéder aux différents outils par les biais d'onglets dédiés sur l'interface web ;
- de récupérer des séquences identifiées depuis une base de données en permettant le choix des espèces d'intérêt ;
- d'obtenir des données et des résultats dans des formats standardisés mais aussi de les afficher directement dans l'interface ;
- d'accéder à différents outils d'alignement et d'en sélectionner les options ;
- d'accéder à un outil de construction d'arbres laissant le choix du type d'arbre et de la méthode à utiliser.

### 1.3.2 Besoins non fonctionnels

Afin de satisfaire les besoins fonctionnels de l'utilisateur, il est nécessaire d'identifier les aspects techniques par l'analyse des besoins fonctionnels :

- L'interface web utilisera une association de langages *front-end* (HTML et CSS), et devra permettre de récupérer les requêtes/choix de l'utilisateur.
- Un *web framework* devra être utilisé afin de construire une interaction entre la partie *front-end* et *back-end* sur la base de Flask qui utilise le langage de programmation Python.
- Un programme devra être conçu à l'aide du langage de programmation Python et de *frameworks* pour permettre la transmission des requêtes aux différents *webservers* détaillés dans l'état de l'art et ainsi obtenir les résultats et les transmettre à l'utilisateur par le biais de l'interface.
- Les différentes interactions décrites précédemment devront être transparentes pour l'utilisateur.
- L'utilisation d'un gestionnaire de versions tel que GitHub sera également nécessaire et permettra d'accéder à la plateforme en ligne.

## 1.4 Objectifs

Compte tenu des recherches exposées précédemment, il paraît essentiel de rappeler le but de ce projet et d'en préciser les objectifs.

Plusieurs difficultés rencontrées par les chercheurs ont été mises en avant et relèvent de la multiplicité des outils et plateformes de NGS et de phylogénétique. L'état de l'art a permis la sélection d'outils pertinents pour ce projet. Il a également permis l'identification de séquences particulièrement étudiées par la communauté scientifique. Il sera alors nécessaire de fournir une plateforme qui permettra aux utilisateurs d'accéder à ces séquences, de les traiter, de les analyser et de les modéliser sous forme d'arbres.

Le premier objectif est donc de concevoir une interface web qui permettra l'intégration de divers outils issus de *webservers*.

Le second objectif consiste en cette intégration. Dans un premier temps, l'interface devra permettre à l'utilisateur d'accéder aux séquences d'intérêt par l'intégration d'une base de données. Ensuite, il sera nécessaire d'intégrer l'outil d'alignement afin de traiter ces séquences afin de les préparer à l'étude phylogénétique. Cette dernière sera également possible depuis l'interface par l'intégration d'outils dédiés.

Le dernier objectif est de permettre à l'utilisateur la sauvegarde des résultats à chaque étape dans des fichiers conformes aux formats standards.

L'intégration est un aspect central dans ce projet. Cela sous-entend un travail sur la compatibilité à différents niveaux : celui de l'interface, des outils, des formats, etc. Il sera donc capital de tester et de vérifier à chaque étape que ces notions sont respectées.

# Chapitre 2

## Conception

### 2.1 Méthodes

Les principaux objectifs ont été identifiés, les prochaines étapes seront d'identifier les méthodes les plus appropriées qui permettront l'achèvement de ce projet.

La première étape sera de rechercher des informations et de choisir les outils de bioinformatique les plus adaptés à intégrer dans l'interface.

Les outils seront sélectionnés en fonction des étapes de formation de l'arbre phylogénétique :

- 1) la sélection de séquences protéiques à partir d'une base de données commune,
- 2) l'alignement des séquences et
- 3) la construction arbre phylogénétique basée sur des tests quantitatifs et sur la comparaison des séquences issues des trois plus grandes banques de gènes au monde : *GenBank*, *EMBL* et *DDBJ*, des séquences d'ADN avec les standards les plus appropriés seront sélectionnées pour la recherche.

Dans ce projet, les séquences d'ADN au format **Fasta** extraites de GenBank seraient les plus appropriées pour être incluses dans la base de données du site Web puisque le nombre de séquences trouvées à partir de GenBank est relativement grand et leurs longueurs sont à peu près égales, ce qui est essentiel pour l'alignement des séquences.

L'étape suivante sera de choisir des outils bioinformatiques pour aligner les séquences. Il s'agit d'une étape vers la réorganisation des séquences et la mise en évidence de leurs similitudes. Dans ce cas, l'alignement est utilisé pour déterminer la relation entre la vipère péliade et d'autres espèces et les comparer entre elles. Par conséquent, les méthodes d'alignement globales et à chaînes multiples seront les plus appropriées. Deux outils seront donc sélectionnés pour ce projet : *Clustal Omega* et *Muscle*. Ensuite, deux méthodes : *Neighbor Joining* et Maximum de vraisemblance seront priorisées pour atteindre l'objectif de construction d'arbres phylogénétiques.

*Neighbor Joining* est une méthode de regroupement dont l'algorithme nécessite la connaissance de la distance entre chaque paire de taxons (par exemple, des espèces ou des séries) pour former un arbre.

La méthode Maximum de vraisemblance utilise des techniques statistiques pour calculer les distributions de probabilité afin d'attribuer des probabilités à des arbres phylogénétiques spécifiques.

Une fois que les outils de bioinformatique appliqués seront choisis, l'étape suivante consistera à identifier les outils informatiques pour créer le site Web et pour y intégrer les outils de bioinformatiques sélectionnés précédemment.

La première étape sera créer le squelette de l'interface. La conception de la mise en page, l'insertion d'images et de contenu textuel ainsi que le paramétrage des menus contenant de nombreux onglets différents se feront en langage **HTML/CSS**.

Une fois le *front-end* de la page Web aura été mis en place, l'étape suivante consistera à le rendre fonctionnel et cela vient bien sûr avec l'aide de **Flask**, un *web framework* qui utilise le langage de programmation **Python**. Il présente une grande flexibilité et permet en outre de sélectionner et de faire correspondre les composants aux souhaits.

A l'aide des packages tels que **Biopython** et **Matplotlib** importés, les scripts *Python* seront déployés avec les fonctions nécessaires pour configurer les boutons fonctionnels de l'interface afin de créer une interaction avec les utilisateurs.

Enfin, pour faciliter la communication lors de la mise en œuvre du projet, le code source sera synchronisé sur *Github* pour faciliter la gestion et l'inspection pendant le travail.

## 2.2 Outils

De nombreux outils et langages de programmation seront utilisés durant ce projet que ce soit pour la gestion ou le développement.

### 2.2.1 Langages de programmation

#### Python

La majeure partie de ce projet est codée en Python 3, c'est un langage de programmation dit interprété, c'est à dire qu'il pourra fonctionner peu importe l'ordinateur utilisé. La première version de ce langage a été introduite en 1991 par Guido Van Rossum. De nombreuses améliorations ont été introduites depuis et la dernière version sortie est la 3.9. Il favorise la programmation impérative structurée (qui possède des structures de contrôle concrètes), fonctionnelle et orientée objet.

Ce langage est particulièrement répandu et possède une large gamme de bibliothèques donnant accès à de très nombreuses fonctionnalités déjà implémentées. C'est, d'ailleurs, ce qui a mené au choix de ce langage pour ce projet : l'existence d'une bibliothèque spécialisée sur divers mécanismes de biologie : par exemple, *biopython* qui sera présentée plus loin. Les outils de ce projet seront donc développés avec la dernière version de python 3.

#### HTML/CSS5

Le HTML (*HyperText Markup Language*) est un langage de balisage conçu pour écrire/représenter des pages web. Il permet plusieurs choses : d'écrire de l'hypertexte (documents contenant des unités d'informations liées entre elles), de structurer sémantiquement une page, de mettre en forme le contenu ou encore d'inclure des ressources telles que des programmes informatiques. La version initiale de ce langage est paru dans les années 1992 et a grandement évolué depuis, il est développé par W3C qui est un organisme de standardisation à but non lucratif. Dans ce projet, il servira à la création d'une interface web donnant accès aux différents outils intégrés.

Le CSS ou encore feuille de style en cascade est un langage informatique qui permet de définir le style d'un document HTML. Ici, il permettra donc de mettre en forme l'interface utilisateur pour lui donner un aspect plus ergonomique et moderne. Créé en 1990, il est actuellement pris en charge par tous les navigateurs web.

#### Jinja2

Jinja est un moteur de *template* pour des applications développées en python ce qui permet aux développeurs de produire des pages web. Il a été basé sur le modèle de *templates* de Django. Il est maintenant l'un des plus utilisés. Il fonctionne sur des principes tels que le *sandboxing* ou encore l'héritage pour permettre à un modèle d'être réutilisable.

### 2.2.2 Modules/*Frameworks*/Outil de développement

#### Biopython

Biopython est un ensemble de modules python qui fournit des fonctions pour gérer les opérations d'une chaîne d'ADN, d'ARN et de protéines telles que le complément inverse d'ADN, la recherche de modules dans la chaîne de protéines, etc. Les fonctionnalités de Biopython incluent des analyseurs pour divers formats de fichiers bioinformatiques (BLAST, Clustalw, FASTA, Genbank, ...), l'accès aux services en ligne (NCBI, Expasy, ...), les interfaces vers des programmes communs et moins courants (Clustalw, DSSP, MSMS ...), Essentiellement, l'objectif de Biopython est de rendre l'utilisation de Python en bioinformatique aussi simple que possible en créant des modules et des classes de haute qualité et réutilisables.

Dans ce projet, **Biopython** a été importé dans des fichiers **Python** pour fournir les fonctions nécessaires à la récupération de séquences de protéines extraites de **GenBank**, à l'alignement de séquences à l'aide de **Clustal Omega** et **Muscle** et à la construction d'arbres phylogénétiques par deux méthodes : *Neighbor Joining* et Maximum de vraisemblance.

## Flask

**Flask** est un *web framework* qui appartient à la catégorie des *micro-frameworks* construits à l'aide du langage de programmation **Python**. **Flask** permet de créer des applications web du plus simple au plus complexe. Il peut créer de petites *API*, des applications web telles que des sites web, des *blogs*, des pages *wiki* ou un site web basé sur le temps ou même un site web commercial.

**Flask** fournit des outils, des bibliothèques et des technologies pour aider les développeurs à faire tout ce qui précède. **Flask** est un *micro-framework*. Cela signifie qu'il est un environnement autonome avec peu de bibliothèques externes. Par conséquent, il a l'avantage d'avoir très peu d'erreurs en raison d'une moindre dépendance ainsi que d'une détection et d'une gestion faciles des erreurs de sécurité.

Ici, **Flask** est utilisé pour créer le *back-end* et établir une relation avec le *front-end* de l'interface.

## Github

**Github** est un système de gestion de projet et de version de code qui fonctionne comme un réseau social pour les développeurs. Les développeurs peuvent cloner le code source à partir d'un répertoire.

**Github** est un service de serveur de répertoire public, chacun peut créer des comptes pour avoir ses propres répertoires afin de pouvoir travailler en collaboration.

Ici, *Github* est utilisé pour synchroniser le code source de l'équipe sur un serveur, aider à la gestion du code source et prendre en charge la manipulation de la vérification du code source durant le développement.

## mod\_wsgi

**Mod\_wsgi** est un module pour les serveurs http d'apache qui fournit ainsi une interface compatible WSGI pour l'hébergement d'application web pour **Python**. C'est une alternative à l'utilisation d'autres modules comme **mod\_python** ou **CGI** pour le déploiement d'une application codée en **python** sur le web. Il est ici utilisé en complément de **Flask**.

## Pythonanywhere

*PythonAnywhere* est un IDE (environnement de développement intégré) en ligne et une plateforme d'hébergement web qui se base sur le langage de programmation **Python**. Sorti en 2012 et créé par Giles Thomas et Robert Smithson, cela permet un accès temporaire (pour la version gratuite) à des serveurs se basant sur des scripts **python** et du **Bash**.

Les fichiers programme peuvent être transférés vers et depuis le service à l'aide du navigateur de l'utilisateur. Les applications Web hébergées par le service peuvent être écrites à l'aide de n'importe quel *framework* d'application basé sur WSGI. *PythonAnywhere* représente une des manières les plus simples de déployer une application web programmée en **python**. Il fonctionne d'ailleurs avec de nombreuses librairies telles que **Cpython**, **Pypy**, **Django**, **flask**, etc ...

Dans ce projet, il sera utilisé pour héberger en ligne l'application.

## 2.3 Prototype

L'idée est d'utiliser **HTML/CSS** pour concevoir le *front-end* de l'interface qui contient un menu avec 5 onglets différents :

- 1) Page d'accueil,
- 2) Informations générales,
- 3) Base de données,
- 4) Alignement,

5) Phylogénie.

Les maquettes des figures 2.1, 2.2, 2.3, 2.3, 2.4 et 2.5 représentent les prototypes des 5 différents onglets qu'il sera possible de trouver sur le site.



FIGURE 2.1 – Page d'accueil

L'onglet "Accueil" présentera les informations contextuelles du projet ainsi que le but et les fonctionnalités de la plateforme créée. En outre, des informations de contact sont également fournies au cas où l'utilisateur aurait besoin d'échanger des informations avec l'équipe de développement.

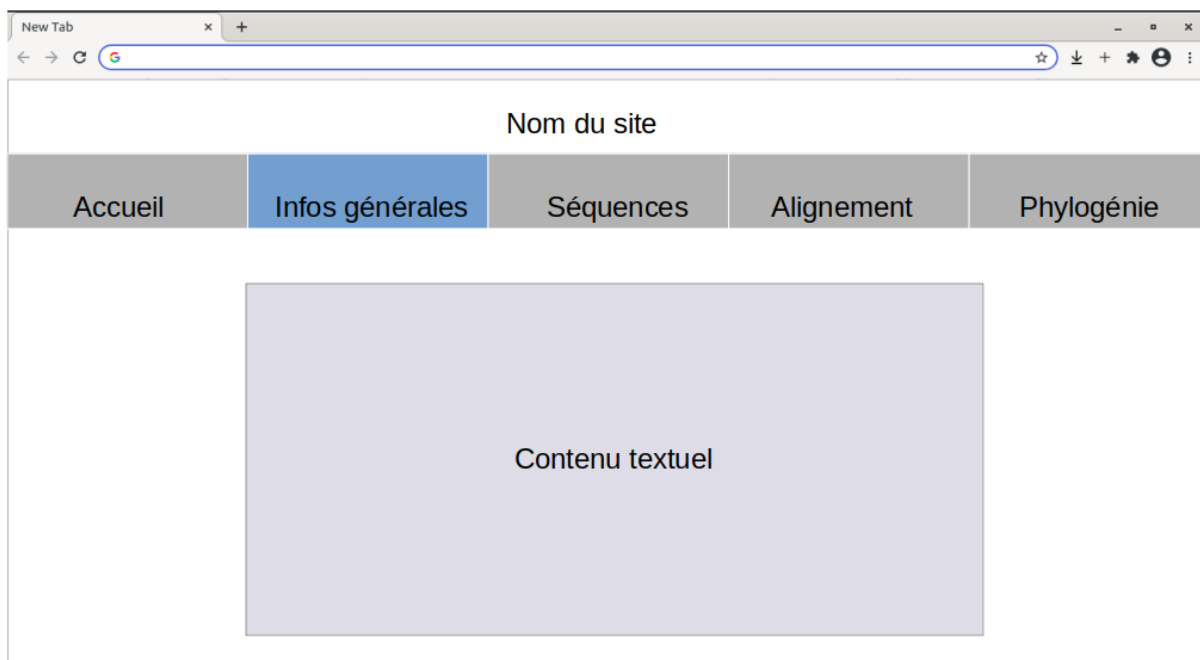


FIGURE 2.2 – Page des informations générales

Ensuite, l'onglet "Informations générales" couvrira les informations générales de l'espèce étudiée : Vipère péliade telles que : nom, classe, caractéristiques morphologiques et biologiques, habitat, taxonomie et arbre phylogénétique représentant la relation entre la Vipère péliade et d'autres espèces identifiées

comme sentinelles du climat.

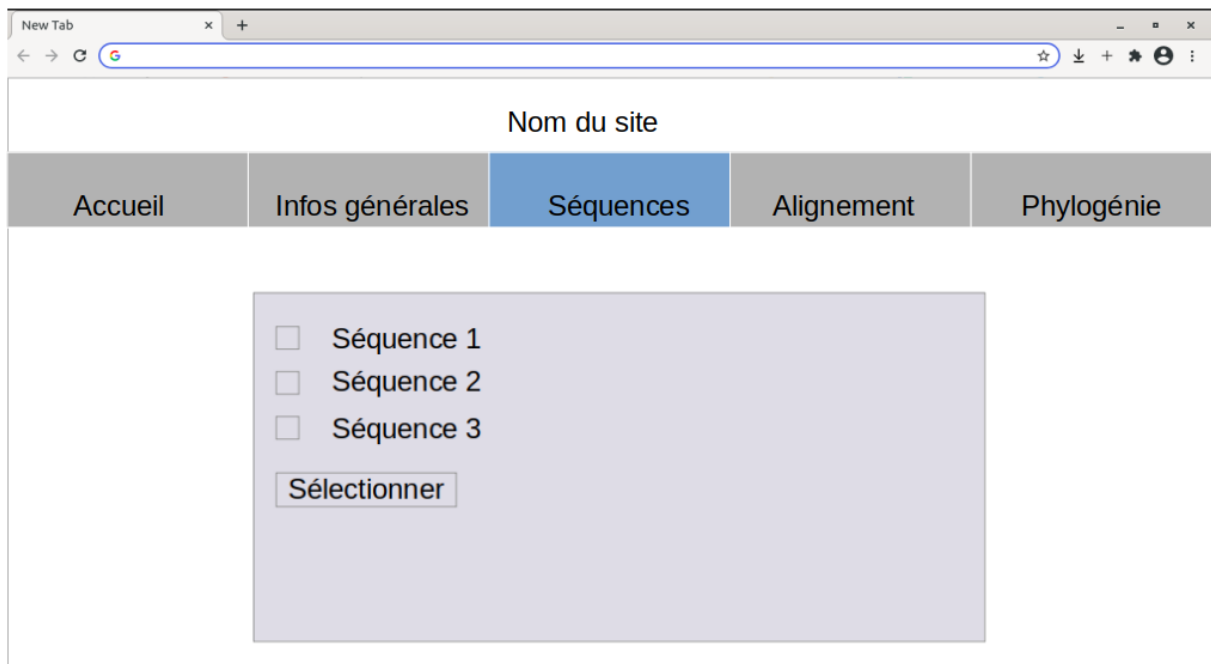


FIGURE 2.3 – Page de sélection des séquences

L'onglet "Séquences" contiendra les noms des séquences protéiques extraites de **Genbank**. Les utilisateurs pourront sélectionner des séquences souhaitées en cochant au minimum 2 cases afin de les comparer et d'observer des relations entre elles. Il y aura aussi un bouton "Sélectionner" utilisé pour rassembler les séquences sélectionnées et passer à l'étape suivante : "alignement".

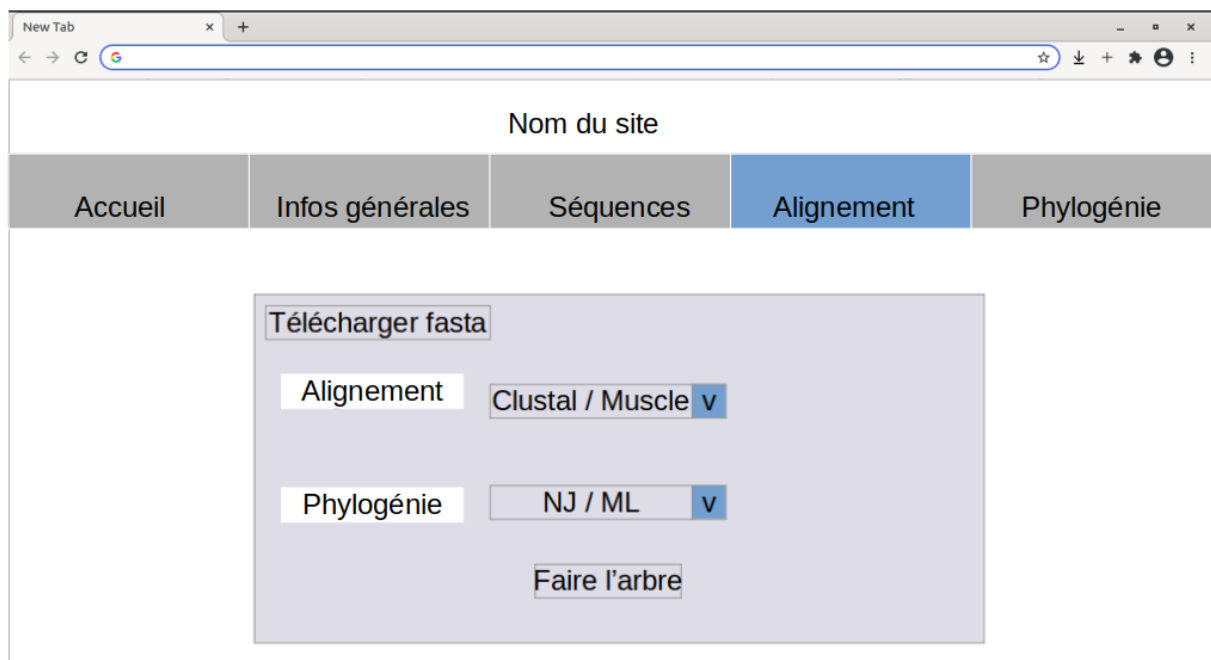


FIGURE 2.4 – Page d'alignement

Le quatrième onglet "Alignement" montre les méthodes utilisées pour aligner des séquences et construire l'arbre phylogénétique. Deux méthodes y seront intégrées pour l'alignement des séquences : **Clustal Omega** et **Muscle**. Deux méthodes seront également disponibles pour l'établissement de l'arbre phylogénétique : **Maximum de vraisemblance** et **Neighbor Joining**.



L'utilisateur pourra choisir une méthode d'alignement pour la réorganisation des séquences pour observer les similitudes entre elles ainsi qu'une méthode de création d'arbre phylogénétique pour l'observation de la relation entre les séquences. Après avoir sélectionné les options pour l'alignement des séquences et la création de l'arbre phylogénétique, le bouton "Faire l'arbre" permettra à l'utilisateur de lancer les processus choisis. De plus, les utilisateurs peuvent trouver dans cet onglet un bouton pour télécharger un fichier au format **Fasta** contenant les séquences précédemment sélectionnées.

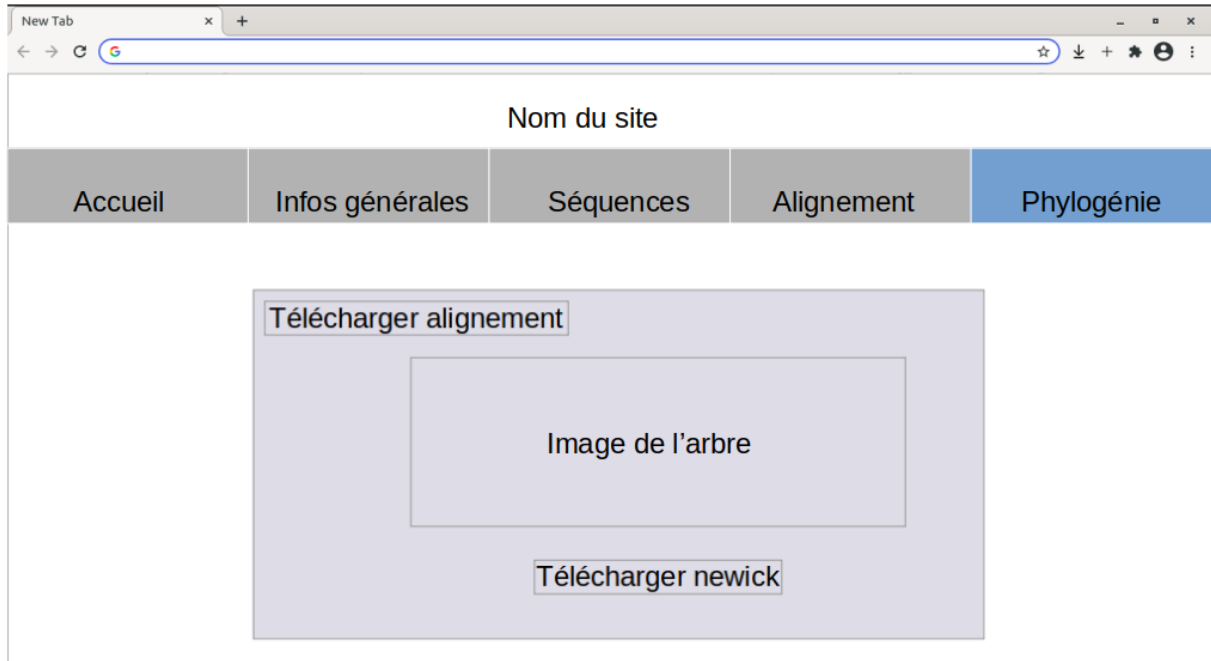


FIGURE 2.5 – Page de phylogénie

Le dernier onglet "Phylogénie" est le lieu de présentation du produit final de l'arbre phylogénétique créé. Ici, l'utilisateur pourra également trouver deux autres boutons : l'un utilisé pour télécharger les résultats de l'alignement des protéines au format **.fasta** et l'autre pour récupérer l'arbre généré au format **Newick**. Il pourra également télécharger l'image de l'arbre par le biais d'un click droit.

## 2.4 Architecture logicielle

La figure 2.6 présente l'architecture logicielle de ce projet.

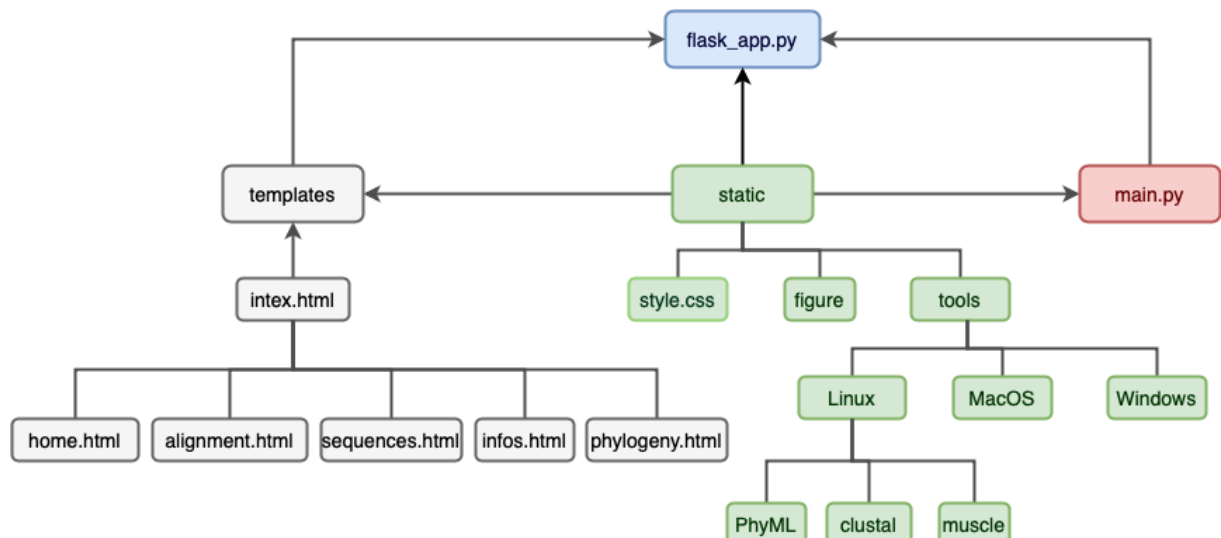


FIGURE 2.6 – Architecture logicielle

L'exécution du fichier `flask_app.py` devra permettre de lancer l'application et d'exécuter le fichier `main.py`. Les fonctions contenues dans `main.py` permettront l'utilisation des outils de bioinformatiques. Les exécutable des outils seront stockés dans le répertoire `/static/tools` pour chaque système d'exploitation. Le répertoire `templates` contiendra tous les fichiers `.html` constituant la partie *front-end* de l'interface utilisateur. Le point d'entrée étant le fichier `index.html`. Les autres fichiers `.html` contiendront respectivement les informations nécessaires au fonctionnement de l'onglet sélectionné par l'utilisateur. Le fichier `style.css` en garantira la mise en page.

Le répertoire `figure` contiendra les figures à afficher dans chaque page web.

Un répertoire `static/data/sauvegardes` sera créé lors de la première utilisation de l'application et stockera toutes les données issues du *pipeline*. Un dossier (avec un nom attribué aléatoirement) par session utilisateur sera créé pour stocker ces données.

# Chapitre 3

## Réalisation

### 3.1 Environnement de travail

#### 3.1.1 Serveur local

Par définition, un serveur local permet de faire fonctionner un site internet sur son propre ordinateur et de simuler les conditions réelles qu'aurait le site sur un hébergeur en ligne traditionnel.

L'utilité de ce genre de serveur réside dans le fait que si tous les utilisateurs d'un même projet, ont la même configuration sur leur propre machine, le site se comportera de la même manière sur toutes les machines et ceci, quel que soit le système d'exploitation (Windows, Linux ou MacOS).

Il existe plusieurs manières de créer un serveur local. La plus connue étant par l'utilisation de Apache. La mise en place d'un serveur local a été expérimentée lors de la phase de développement par l'utilisation de Wamp, Lamp et Mamp. Ce sont des serveurs qui combinent Apache, MySQL et PHP selon le système d'exploitation utilisé (Wamp pour Windows, Lamp pour Linux, Mamp pour MacOS).

Cependant, comme le développement de l'application utilise Flask et que ce dernier sert également de serveur local, il est apparu plus judicieux de se concentrer davantage sur cette alternative pour la suite du développement. En effet, lors de l'exécution de l'application Flask dans un environnement virtuel, une adresse *localhost* est indiquée dans le terminal pour permettre la visualisation et l'utilisation de l'interface depuis n'importe quel navigateur en local. Cela constitue une manière simple pour l'utilisateur de bénéficier des services de la plateforme en local quel que soit le système d'exploitation.

#### 3.1.2 Environnement virtuel

Afin de s'adapter aux différentes versions des outils, le projet doit être exécuté dans un environnement d'exploitation Python isolé. Dans ce cas là, **virtualenv** est utilisé pour créer un environnement Python contenant une version spécifique et garantir que le *package* Python est sain. Cette ligne de commande créera un dossier dans le répertoire courant contenant le fichier exécutable Python et une copie de la bibliothèque **pip**, afin que d'autres *packages* puissent être installés. Lorsque l'environnement virtuel est activé, le projet peut être exécuté et les opérations suivantes peuvent être réalisées.

Un guide utilisateur a été élaboré dans l'annexe A et également dans le fichier **README.md** sur GitHub afin d'explicitier la manière d'installer et de configurer l'environnement virtuel Flask ainsi que son serveur local.

#### 3.1.3 Dépôt et mise en ligne

L'intégralité du programme est accessible en ligne sur un dépôt GitHub<sup>1</sup>. Le gestionnaire de version GitHub a permis de collaborer à chaque étape du développement.

D'autre part, *PythonAnywhere*<sup>2</sup> a été utilisé en combinaison avec GitHub afin de permettre la mise en ligne de l'interface utilisateur gratuitement. La plateforme *PythonAnywhere* est un environnement de développement en ligne et un service d'hébergement web basé sur le langage de programmation Python. Sur cette plateforme, il est possible de créer un environnement virtuel et d'héberger le programme afin

---

1. <https://github.com/les-sentinelles-de-la-bioinfo/ProjetPhylogenie>

2. <https://www.pythonanywhere.com>

de pouvoir l'utiliser en ligne.

Tout d'abord, il a été nécessaire de créer un projet et de configurer l'environnement en précisant la version de Python utilisée ainsi que le *framework* (ici **Flask**). Ensuite, le dépôt GitHub du projet a été lié à *PythonAnywhere* par le biais d'une console intégrée. Un nom de domaine a alors été attribué : **sentinellesbioinfo.pythonanywhere**. L'interface utilisateur est alors accessible en ligne<sup>3</sup> sans que l'utilisateur n'ait besoin d'installer ou de configurer quoi que ce soit sur sa machine.

L'avantage de lier *PythonAnywhere* au dépôt GitHub est de pouvoir actualiser facilement le site web tout au long des phases de tests du développement. En effet, la console intégrée permet de *pull* les modifications apportées au programme par les différents contributeurs. D'autre part, il est aussi possible de modifier le programme directement depuis l'IDE de *PythonAnywhere* et de *push* les modifications sur le dépôt. Ainsi, il est facile de déboguer et de mettre à jour le site web sans avoir à recopier ou télécharger des fichiers à chaque modification.

L'inconvénient de la version gratuite de *PythonAnywhere* est la durée de mise en ligne. En effet, le site est hébergé gratuitement durant 3 mois. Au delà, il ne sera plus visible en ligne. Cependant, une fonctionnalité permet d'être notifié lorsque l'échéance approche afin de procéder au renouvellement de cette durée pour 3 mois de plus. Cela est valable indéfiniment et reste très acceptable dans le cadre de ce projet. Par ailleurs, ce léger désagrément peut être pallié par l'utilisation de la version payante. La version payante pourrait également permettre d'héberger plusieurs projets sur un même compte développeur contre un seul pour la version gratuite. Une fois encore, l'hébergement d'un seul projet est suffisante dans le cadre du travail réalisé ici.

## 3.2 Implémentation

### 3.2.1 Développement du *pipeline*

Afin d'utiliser les outils permettant le traitement des données (recherche de séquences, alignement, phylogénie), un *pipeline* a été réalisé à l'aide d'un programme Python qui utilise principalement le module **biopython** et le module **matplotlib**.

Le programme réalisé permet tout d'abord la recherche de séquences grâce à la fonction **get\_fasta()**. Cette fonction utilise le sous-module **Entrez** du module **biopython**. Ce module permet d'accéder à NCBI et d'effectuer des recherches dans les différentes banques de données ainsi que de récupérer des informations.

Les séquences nucléotidiques d'un certain nombre de gènes peuvent être récupérées en spécifiant l'identifiant *GenBank* du gène recherché. Un fichier **.fasta** par séquence est ainsi créé dans les données de sauvegarde. Ces fichiers sont ensuite combinés pour créer un fichier **multifasta** unique qui permettra par la suite l'alignement.

Ensuite, en utilisant le fichier **multifasta** généré à l'étape précédente, il est possible de procéder à un alignement. Deux outils sont proposés pour réaliser cet alignement : *Clustal Omega* et *Muscle*. Il est possible de les utiliser grâce aux sous-modules **ClustalWCommandLine** et **MuscleCommandLine** du module **biopython**, qui utilisent les programmes *Muscle* et *Clustal*. Ces derniers doivent être téléchargés au préalable. Ces modules permettent d'utiliser des lignes de commandes dans un programme en Python sans avoir à passer directement par le terminal.

Les deux fonctions réalisant cet alignement sont **clustal\_alignment** et **muscle\_alignment**. Elles prennent en entrée le fichier **multifasta** et créent un nouveau fichier **.fasta** contenant l'alignement dont la structure sera différente selon l'outil qui a été utilisé.

Enfin, à l'aide du fichier d'alignement il est possible de créer un arbre phylogénétique.

Deux méthodes sont proposées pour cela : *neighbor joining* et *maximum likelihood* à l'aide de l'outil *PhyML*. Ces deux méthodes utilisent le module **Phylo**, qui permet ici de convertir des fichiers d'alignements, de les lire, et de générer les arbres.

La fonction permettant d'utiliser le *neighbor joining*, **NJ\_tree()**, prend en entrée un fichier d'alignement dont le type (*clustal* ou *muscle*) doit être spécifié. Grâce à la fonction **DistanceTreeConstructor** du sous-module **Phylo** de **Biopython**, l'arbre est créé. L'image de l'arbre est ensuite sauvegardée pour l'afficher par la suite sur le site grâce au module **matplotlib**.

---

3. <http://sentinellesbioinfo.pythonanywhere.com>

La fonction permettant d'utiliser l'algorithme de *maximum likelihood*, `ML_tree()`, utilise quant à elle le sous-module `PhyMLCommandLine` qui permet l'interaction avec le programme PhyML, qui doit être téléchargé.

La fonction convertit d'abord les fichiers d'alignement au format `.phylip`, puis crée l'arbre dont l'image sera également enregistrée pour l'afficher grâce au module `matplotlib`.

L'ensemble de ces étapes permet donc d'établir et d'afficher des arbres phylogénétiques à partir d'identifiants de gènes, en laissant l'utilisateur choisir les outils et méthodes qui lui conviennent.

### 3.2.2 Développement de l'interface utilisateur

Pour mettre en place et en forme l'interface web, les langages de programmation HTML5 et CSS5 ont été utilisés. L'application web est partagée en 6 fichiers `.html` (`home.html`, `alignment.html`, `sequences.html`, `infos.html`, `index.html`, `phylogeny.html`) qui sont stylisés grâce à un seul fichier `.css`.

L'interface se présente à l'utilisateur sous la forme d'une seule page web contenant une barre de menu avec différents onglets (5 au total) : accueil, infos générales, séquences, alignement et phylogénie. Chaque onglet correspond à une page `.html` particulière. Toutefois, cela sera transparent pour l'utilisateur. Il aura réellement de passer d'un onglet à l'autre sans chargement d'une nouvelle page. L'utilisateur pourra savoir de manière claire sur quelle page il se trouve grâce à la coloration de l'onglet utilisé. Pour éviter toute erreur dans l'utilisation du site, seul 3 onglets sont cliquables : accueil, infos générales et séquences. A partir de l'onglet séquences, le basculement sur l'alignement et la phylogénie se fait automatiquement pour respecter l'ordre du *pipeline*

Chaque fichier `.html` est organisé de la même façon et des sections communes sont retrouvées. Le fichier `index.html` contient le squelette général des pages. Il va permettre de mettre en place l'en-tête du site, le menu, le *footer* ainsi que les couleurs de l'interface. Il sera rappelé au début de chaque fichier `.html` avec la ligne de commande suivante `% extend "index.html"%`, afin d'appliquer cet élément à chaque page/onglet.

Grâce à la méthode d'extension de modèle (`% block content% - % endblock%`), il est possible de partager le code `index.html` sur différentes pages Web. Lorsque les mêmes informations ou la même mise en page doivent être utilisées, ou qu'il est nécessaire de modifier le contenu d'un modèle, il permet d'éviter la répétition de code dans chaque fichier. Ainsi, un seul fichier doit être modifié.

Dans les onglets Accueil et Infos générales quelques informations contextuelles du projet, le fonctionnement du site Web ainsi que l'introduction de l'espèce sélectionnée (la vipère péliade : *vipera berus berus*) sont brièvement présentées. Ce sont des pages simples contenant seulement du texte sans réelle interaction avec l'utilisateur.

Dans l'onglet **Séquences**, des `checkbox` sont disponibles afin de permettre aux utilisateurs de sélectionner les séquences cibles fournies par le site Web. Le choix est validé en cliquant sur le bouton "sélectionner".

Dans l'onglet **Alignement**, le bouton "Télécharger Fasta" permet de télécharger le format `.fasta` des séquences cibles. Deux méthodes d'alignement sont proposées (*Clustal* / *Muscle*) et deux méthodes pour construire l'arbre phylogénétique (*Neighbor Joining* / *Maximum Likelihood*). Après avoir sélectionné la méthode, le bouton "Construction de l'arbre" permet de lancer la construction et l'affichage de l'arbre.

L'onglet **Phylogénie** montre la figure de l'arbre.

### 3.2.3 Intégration

Le *microframework* Flask du fichier `flask_app.py` est utilisé dans cette partie pour intégrer le *back-end* développé en Python au sein du *front-end* développé en HTML. Le routage URL est écrit directement au-dessus de la fonction à exécuter et est défini à l'aide du décorateur `app.route` de Flask. A travers cela, un modèle de Jinja2 est renvoyé avec les paramètres requis par le modèle. Cette fonction peut aussi permettre l'utilisation de différentes méthodes de requête HTTP. Les plus couramment utilisées sont *Get*, *Post*, *Put*, *Delete*.

Les cinq fonctions suivantes sont respectivement responsables des cinq routages.

— La fonction `home()` renvoie le modèle `home.html` quand l'utilisateur sélectionne la page d'accueil.

- La fonction `infos()` renvoie le modèle `infos.html` quand l'utilisateur sélectionne la page d'infos.
- La fonction `sequences()` renvoie une liste d'identifiants de gènes dans le modèle `sequences.html` pour remplir les éléments dans les `checkbox`.
- La fonction `getSequences()` peut recevoir les éléments cochés par l'utilisateur dans la page de sélection des séquences. Elle appelle la fonction `main.get_fasta()` pour générer un fichier au format `.fasta` qui contient les séquences sélectionnées. Lorsque l'utilisateur fait une sélection, la fonction retournera un modèle et la page passera automatiquement à l'étape suivante du *pipeline*.
- La fonction `alignmentAndPhylogeny()` s'occupe de collecter les choix que l'utilisateur fait dans le routage `/alignment`. Ces choix peuvent diriger quelles fonctions dans le `main.py` peuvent être exécutées. La figure ainsi que le résultat de ces fonctions est sauvegardé dans le dossier statique `static`. Enfin, cette fonction envoie le modèle `phylogeny.html` pour afficher la figure générée.

À ce stade, le *back-end* et le *front-end* sont bien connectés. L'appel à chaque fonction du fichier `main.py` et pris en charge, et la tâche de collecte des requêtes et d'envoi des résultats depuis la page web l'est également. Le navigateur web possédant une cache, il est donc nécessaire de définir l'heure d'expiration du cache des fichiers statiques avec `APP.CONFIG['SEND_FILE_MAX_AGE_DEFAULT'] = TIMEDELTA(SECONDS=1)`.

### 3.3 Résultats

La finalité de ce projet se présente sous la forme d'un site web qui a été mis en ligne et dont le nom est « Les sentinelles de la bioinfo ». Celui-ci possède une interface web claire et intuitive qui a été divisée en plusieurs onglets :

- Accueil.
- Informations générales.
- Séquences.
- Alignement.
- Phylogénie.

Le style de la page reste assez simple avec une gamme de couleur limitée, afin de donner un aspect plus épuré au projet. Il est possible pour l'utilisateur de naviguer d'un onglet à l'autre en ayant toujours un moyen de savoir de manière claire sur quelle page il se trouve grâce à la coloration de l'onglet lorsque celui-ci est sélectionné.

La page internet s'ouvre par défaut sur l'onglet "Accueil" de l'interface comme présenté dans la figure 3.1. Cet onglet contient diverses informations liées au contexte de ce projet, au contenu du site web et à sa création. Il sert de mise en contexte et permet aussi de mettre en avant le projet initial dont découle cette plateforme ("Les sentinelles du climat"). En bas de cette page, une partie "*User guide*" est accessible afin de présenter de manière simple et claire le fonctionnement de l'interface.



FIGURE 3.1 – Onglet d'accueil

Dans le deuxième onglet s'intitulant « Infos générales », diverses informations sur l'espèce choisie pour ce projet sont présentées : *vipera berus berus*. La figure 3.2 montre un extrait de cet onglet.

Ce site étant spécialisé sur cette espèce sentinelle, il a semblé intéressant d'offrir la possibilité à l'utilisateur de visualiser les informations les plus communes et importantes sur ce reptile. Cette page se divise donc en plusieurs sections donnant des informations morphologiques, éthologiques et géographiques sur la vipère péliade. Un arbre des espèces généré au préalable par les outils qui ont été développés pour ce projet est également présent dans cette section. Cet arbre permet de situer la vipère péliade par rapport à d'autres notamment considérées comme sentinelles du climat.



FIGURE 3.2 – Onglet contenant les informations générales

Le troisième onglet intitulé « Séquences » marque le début du *pipeline* de phylogénie qui a été mis

en place. Il est présenté dans la figure 3.3. Ainsi, il sera nécessaire de passer par cet onglet avant de poursuivre sur les autres.

Dans cet onglet, des gènes ont été sélectionnés pour leur relation avec le venin de la vipère péliade. D'autres gènes ont été sélectionnés provenant d'autres espèces de serpents afin de construire un arbre des gènes. Ce dernier peut renseigner sur l'évolution de ces gènes chez la vipère. Une liste de gènes est donc proposée avec la possibilité d'en sélectionner certains ou l'intégralité par un système de case à cocher. Une fois le choix réalisé, il est possible de télécharger les séquences sélectionnées et de passer automatiquement au prochain onglet du *pipeline*.

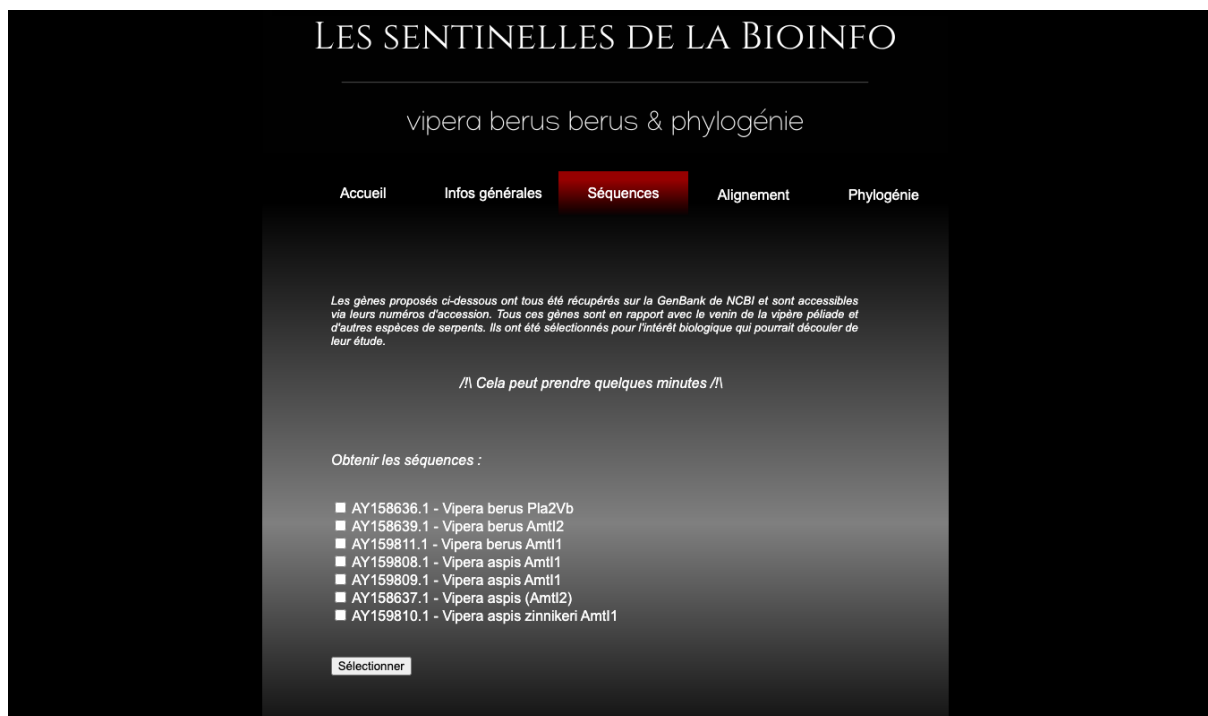


FIGURE 3.3 – Onglet de choix des séquences

Le quatrième onglet intitulé « Alignement » est chargé automatiquement une fois que les séquences sont récupérées. Il est présenté dans la figure 3.4.

Cet onglet propose tout d'abord la possibilité de télécharger le fichier *multifasta* des séquences choisies précédemment, grâce à un bouton en haut de la page. Il permet également de réaliser l'alignement ainsi que la construction d'arbre. L'utilisateur a le choix entre deux outils pour réaliser l'alignement, *Clustal Omega* ou *Muscle*, qu'il peut choisir grâce à un menu déroulant.

Une fois ce choix effectué, il peut choisir la méthode de construction d'arbre, *Neighbor Joining* ou *Maximum Likelihood*. En appuyant sur le bouton « Faire l'arbre », ces étapes seront effectuées et l'utilisateur pourra passer automatiquement à l'onglet suivant.





FIGURE 3.4 – Onglet d'alignement et de choix d'outil pour la réalisation de l'arbre

Enfin le cinquième onglet « Phylogénie » est automatiquement chargé une fois que l'alignement et la construction de l'arbre sont effectués. Cet onglet, présenté dans la figure 3.5, propose tout d'abord à l'utilisateur de télécharger le fichier d'alignement généré à l'étape précédente au format **.fasta** grâce à un bouton en haut de la page. Ensuite, il est possible d'y trouver un affichage de l'arbre généré par l'outil choisi. Enfin, un bouton en bas de la page permet à l'utilisateur de télécharger la séquence Newick de l'arbre sous la forme d'un fichier **.txt**.

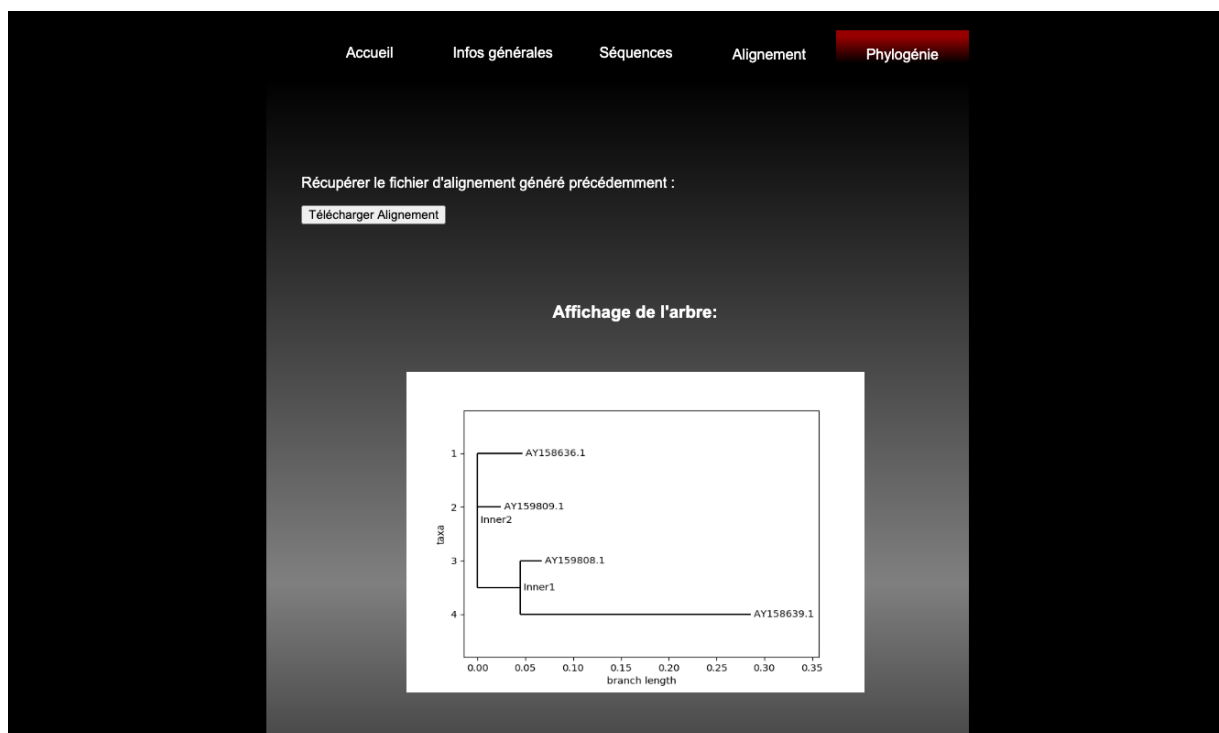


FIGURE 3.5 – Onglet de phylogénie

### 3.4 Limites et perspectives

La réalisation de la plateforme "Sentinelles de la Bioinfo" permet l'étude des gènes impliqué dans la production de venin chez la vipère péliade et seulement cela. De plus, le *pipeline* doit être utilisé dans son intégralité pour fonctionner et n'accepte pas de recevoir des fichiers fournis par l'utilisateur.

Il pourrait être intéressant, à l'avenir, d'ajouter des fonctionnalités donnant la possibilité à l'utilisateur de charger ses propres fichiers et ainsi lui permettre d'utiliser les outils quels que soient les gènes étudiés. Cela permettrait de généraliser l'utilisation de cette plateforme à l'étude d'autres gènes et d'autres espèces. De plus, l'utilisateur pourrait ainsi accéder à chaque étape du *pipeline* indépendamment.

D'autre part, il pourrait également être intéressant d'ajouter des annotations ou une légende lors de la création des arbres afin d'en faciliter la lecture et l'interprétation.

Enfin, il serait intéressant de faire tester cette plateforme à différents laboratoires ou chercheurs afin d'en vérifier l'ergonomie et la performance. Ces tests pourraient alors aboutir sur une nouvelle phase de développement et peut-être sur une phase de déploiement différente de *PythonAnywhere*.

# Conclusion

Le développement de cette plateforme utilisateur permettant l'utilisation centralisée de divers outils bioinformatiques a été réalisée et déployée dans le cadre de projet. Elle confère à l'utilisateur la possibilité de réaliser une étude phylogénétique de gènes d'une espèce sentinelle du climat (la vipère péliade) de bout en bout de manière transparente.

Initialement, ce travail devait permettre à un client d'utiliser des outils de NGS et de phylogénie issus de différents *web server* sur une même interface en toute transparence. Une des améliorations potentielles attendue était la possibilité de télécharger les fichiers à chaque étape puis de rendre cette interface accessible en ligne. Ces différentes améliorations ont été réalisées. En effet, la plateforme "Les sentinelles de la bioinfo" est accessible en ligne et permet à l'utilisateur de réaliser le *pipeline* précédemment cité mais aussi de récupérer, à chaque étape, des fichiers standardisés qu'il pourrait utiliser avec d'autres outils.

En conclusion, les objectifs fixés ainsi que les améliorations potentielles préalablement identifiées ont été réalisés. De plus, le travail de recherche et d'analyse effectué en amont a permis une identification juste et claire des besoins et des outils nécessaires. Enfin, le programme est fonctionnel et répond aux attentes fixées mais pourrait être amélioré par l'ajout des diverses fonctionnalités comme la possibilité de charger des fichiers de l'utilisateur ou d'utiliser indépendamment les outils du *pipeline*. En outre, cela permettrait un déploiement plus vaste de ce programme chargé de faciliter le travail des chercheurs par la centralisation de divers outils de NGS et de phylogénie.

# Bibliographie

- [Al-Shekhadat *et al.*, 2019] AL-SHEKHADAT, R., LOPUSHANSKAYA, K., SEGURA, , GUTIÉRREZ, J., CALVETE, J. et PLA, D. (2019). Vipera berus berus venom from russia : Venomics, bioactivities and preclinical assessment of microgen antivenom. 11.
- [Altschul *et al.*, 1990] ALTSCHUL, S., GISH, W., MILLER, W., MYERS, E. et LIPMAN, D. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410.
- [Bocian *et al.*, 2016] BOCIAN, A., URBANIK, M., HUS, K., ŁYSKOWSKI, A., PETRILLA, V., ANDREJČÁKOVÁ, Z., PETRILLOVÁ, M. et LEGATH, J. (2016). Proteome and peptidome of vipera berus berus venom. *Molecules*, 21(10):1398.
- [CREUX, 2019] CREUX, T. (2019). Les vipères filent vers l’extinction, dans le silence le plus complet.
- [Czajka *et al.*, 2013] CZAJKA, U., WIATRZYK, A. et LUTYŃSKA, A. (2013). Mechanism of vipera berus venom activity and the principles of antivenom administration in treatment. 67:641–646, 729–733.
- [Edgar, 2004] EDGAR, R. (2004). Muscle : a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, 5:113.
- [Guillemin *et al.*, 2003] GUILLEMIN, BOUCHIER, GARRIGUES, WISNER et CHOUMET (2003). Sequences and structural organization of phospholipase a2 genes from vipera aspis aspis, v. aspis zinnikeri and vipera berus berus venom. 270:2697–2706.
- [Hamilton *et al.*, 2019] HAMILTON, J., KAUSE, J. et LAMB, T. (2019). Severe systemic envenomation following vipera berus bite managed with viperatab antivenom. 30:56–58.
- [Hermansen *et al.*, 2019] HERMANSEN, M., KRUG, A., TJØNNFJORD, E. et BRABRAND, M. (2019). Envenomation by the common european adder (vipera berus) : a case series of 219 patients. 26:362–365.
- [Losange, 2008] LOSANGE (2008). *Amphibiens et reptiles*. Editions Artemis.
- [Malina *et al.*, 2013] MALINA, T., BABOCSAY, G., KRECSÁK, L. et ERDÉSZ, C. (2013). Further clinical evidence for the existence of neurotoxicity in a population of the european adder (vipera berus berus) in eastern hungary : second authenticated case. 24:378–383.
- [Malina *et al.*, 2017] MALINA, T., KRECSÁK, L., WESTERSTRÖM, A., SZEMÁN-NAGY, G., GYÉMÁNT, G., M-HAMVAS, M., ROWAN, E., HARVEY, A., WARRELL, D., PÁL, B., RUSZNÁK, Z. et VASAS, G. (2017). Individual variability of venom from the european adder (vipera berus berus) from one locality in eastern hungary. 135:59–70.
- [Netgen, 2011] NETGEN (2011). Une morsure de serpent venimeux : une mort sûre ?
- [Olsen *et al.*, 1999] OLSEN, R., HWA, T. et LÄSSIG, M. (1999). Optimizing smith-waterman alignments. *Pacific Symposium on Biocomputing*, page 302–313.
- [Phillips *et al.*, 2010] PHILLIPS, D., SWENSON, S., FRANCIS, S., MARKLAND, J. et MACKESSY, S. (2010). Thrombin-like snake venom serine proteinases. *Handbook of Venoms and Toxins of Reptiles*, 139:154.
- [Sievers et Higgins, 2014] SIEVERS, F. et HIGGINS, D. G. (2014). Clustal omega, accurate alignment of very large numbers of sequences. *Methods in molecular biology*, 1079:105–116.

# Appendices

# Annexe A

## Guide utilisateur

### A.1 Installation, configuration et démarrage du serveur Flask

1. Installer Python 3.

`https://www.python.org/downloads/`

2. Cloner le *repository* GitHub ou télécharger le .zip.

`https://github.com/les-sentinelles-de-la-bioinfo/ProjetPhylogenie`

3. Créer l'environnement virtuel dans le répertoire contenant flask\_app.py.

`python3 -m venv env`

4. Activer l'environnement virtuel.

`source env/bin/activate`

5. Installation de Flask et des modules/*packages*.

`pip3 install Flask`

`pip3 install matplotlib`

`pip3 install biopython`

6. Démarrage de Flask.

`export FLASK_APP=flask_app.py`

`export FLASK_ENV=env`

`flask run`

L'interface utilisateur est maintenant disponible à l'adresse suivante : `http://127.0.0.1:5000`

### A.2 Utilisation du site en ligne

Le site en ligne (`http://sentinellesbioinfo.pythonanywhere.com`) est composé de plusieurs pages dont chacune d'entre-elles a une fonction particulière.

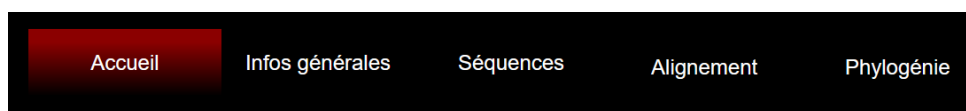


FIGURE A.2.0.1 – Composition du site

#### A.2.1 Accueil

Sur cette page, on retrouve les informations sur le thème du site dans un paragraphe "A propos" ainsi qu'un résumé des outils mis à disposition pour l'utilisateur.

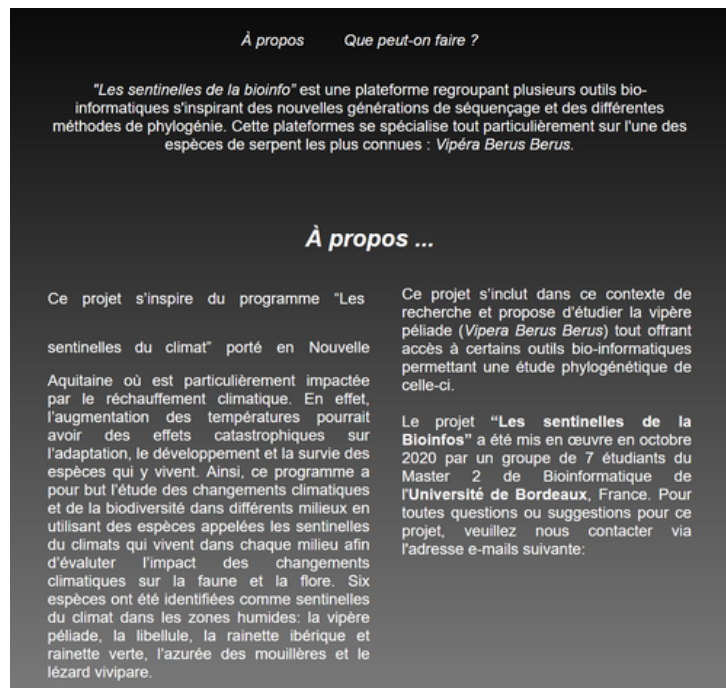


FIGURE A.2.1.1 – Accueil

## A.2.2 Infos générales

Cette page donne des renseignements sur l'espèce présentée sur le site : la vipère péliade. L'utilisateur peut y trouver une rapide présentation de l'espèce avec :

- Ses caractères morphologiques et biologiques.
- Des indications sur son comportement et son alimentation.
- Sa phylogénie avec sa taxonomie et l'arbre phylogénétique de son espèce.



FIGURE A.2.2.1 – Infos générales

## A.2.3 Séquences

1. L'utilisateur doit choisir les séquences qu'il veut étudier en cochant les cases correspondantes. Il faut sélectionner plus de deux séquences.

2. Récupération des séquences choisies dans la base de données en ligne (*NCBI*).

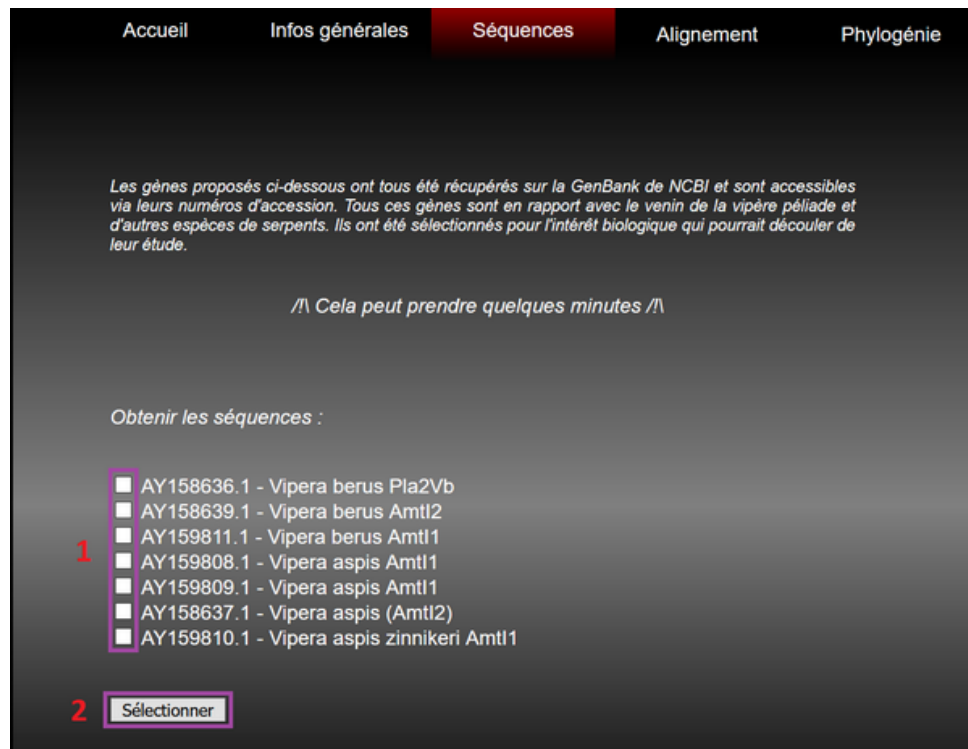


FIGURE A.2.3.1 – Séquences

#### A.2.4 Alignement

Sur cette page, l'utilisateur peut choisir :

1. La méthode d'alignement : *Clustal* ou *Muscle*.
2. La méthode pour construire l'arbre phylogénétique : *Neighbor Joining* ou *Maximum Likelihood*.

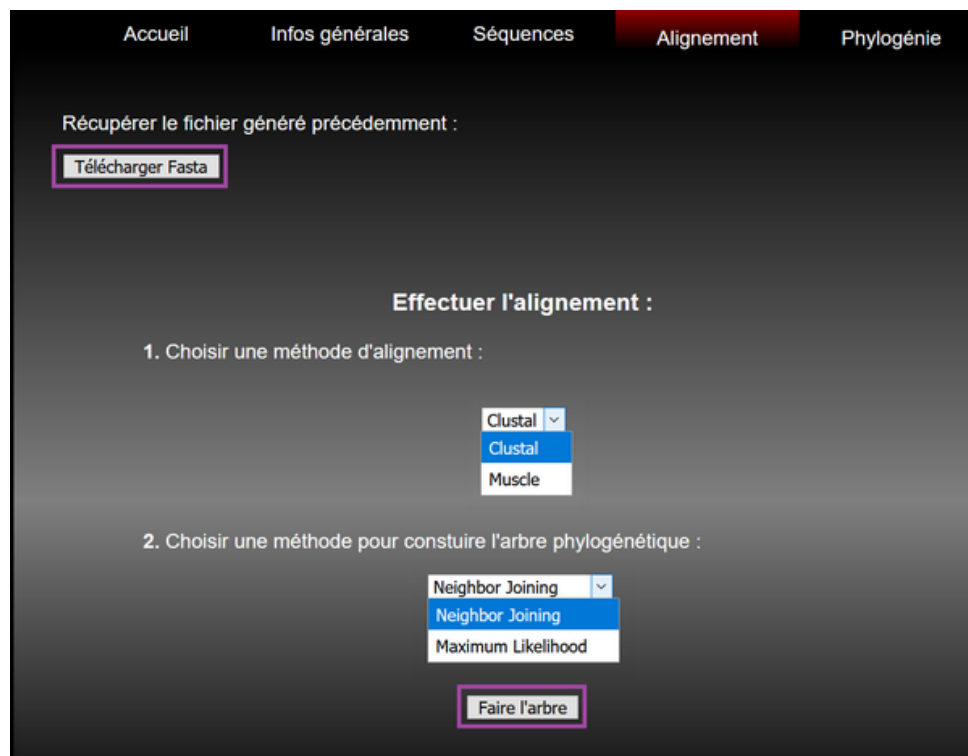


FIGURE A.2.4.1 – Alignements

3. Pour générer l'arbre, l'utilisateur doit cliquer sur "Faire l'arbre".



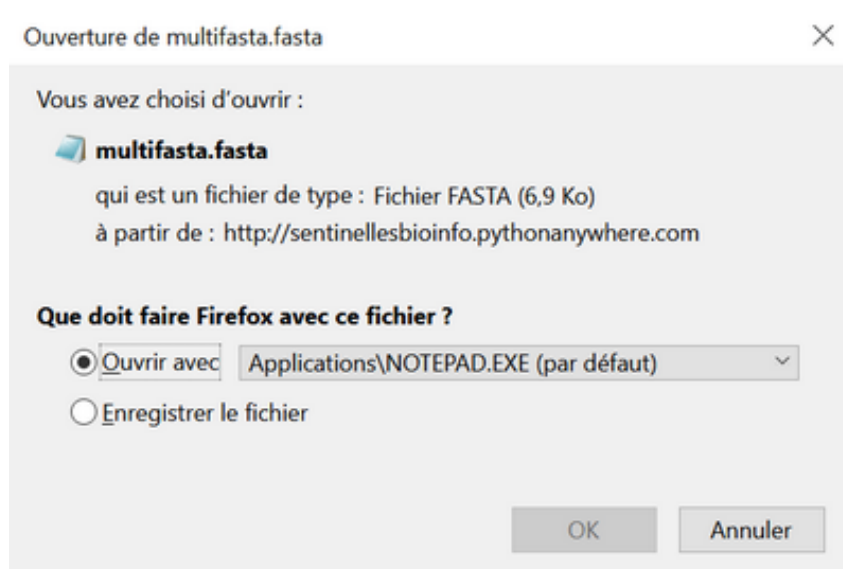


FIGURE A.2.4.2 – Télécharger un fichier *Fasta* des séquences sélectionnées

Mais il peut aussi télécharger un fichier `.fasta` qui contient les séquences choisies précédemment.

### A.2.5 Phylogénie

Cette page donne le résultat de l'alignement sous forme d'arbre.

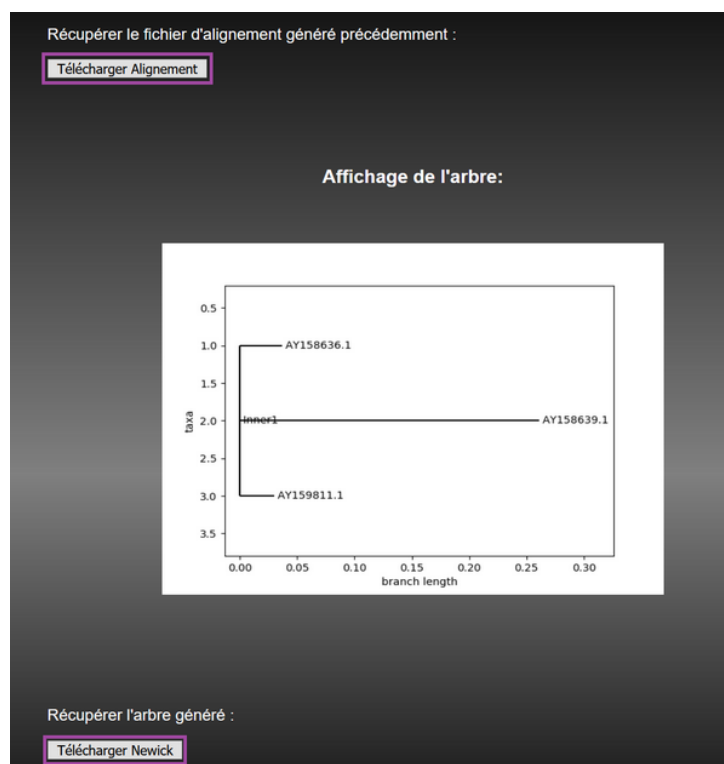


FIGURE A.2.5.1 – Affichage de l'arbre phylogénétique

L'utilisateur peut :

- Télécharger le fichier d'alignement précédemment créé.

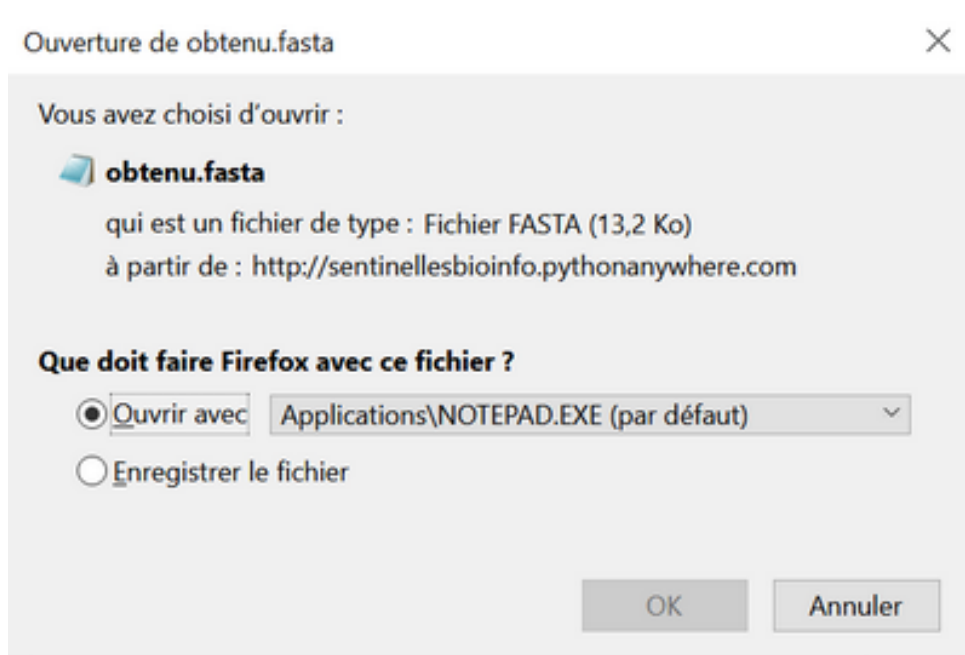


FIGURE A.2.5.2 – Télécharger le fichier d'alignement

- Enregistrer l'image de l'arbre en faisant un clic-droit sur l'image.

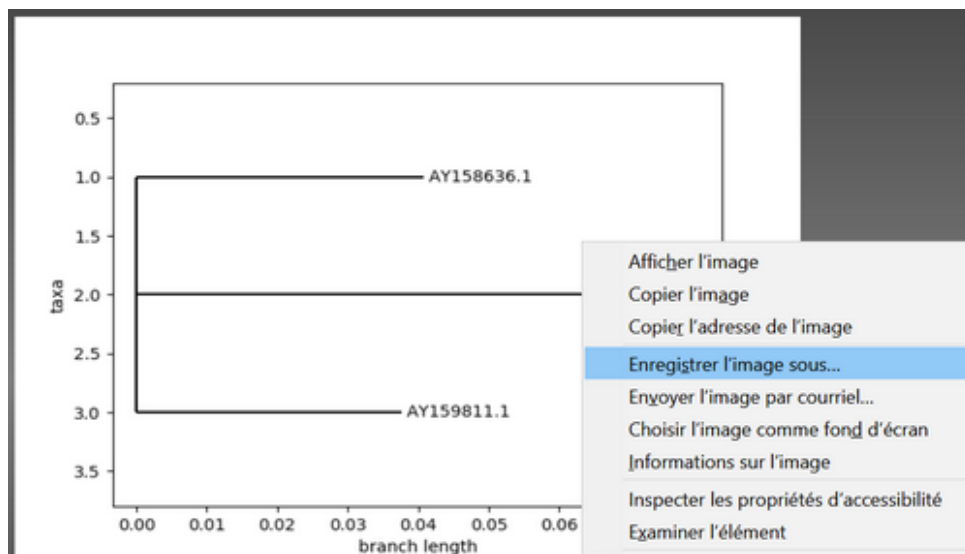


FIGURE A.2.5.3 – Enregistrer l'image de l'arbre phylogénétique

- Récupérer l'arbre généré en format *Newick*.

```
(AY159808.1:0.00162,AY159810.1:0.00324,(AY158637.1:0.01013,AY158639.1:0.21709)Inner1:0.05954)Inner2:0.00000;
```

FIGURE A.2.5.4 – Télécharger l'arbre phylogénétique en format Newick