

IUT Aix-Marseille - Site Aix-en-Provence
413 avenue Gaston Berger
13625 Aix-en-Provence cedex 1

Université Aix-Marseille - Site de Luminy
Inserm TAGC UMR_S1090
Parc Scientifique de Luminy, Case 928
13288 Marseille cedex 9, France

Finalisation de l'application mimicINTWeb

Rapport de stage

17/06/2022 - Version 1.9

Gestionnaire de version

Date	Version	Modification
15/04/2022	1.1	Mise en page, fiche technique
20/05/2022	1.2	Remerciements, 1er partie présentation de l'entreprise
25/05/2022	1.3	2ème et 3ème partie de la section présentation de l'entreprise, Matériels et outils logiciel
03/06/2022	1.4	1er et 2ème partie de la section travail réalisé
10/06/2022	1.5	2ème et 3ème partie de la section travail réalisé, Introduction
13/06/2022	1.6	Introduction et conclusion de la section travail réalisé
14/06/2022	1.7	Reformulation de matériels et outils utilisés, de la 2ème partie de la section travail réalisé.
15/06/2022	1.8	Bilan et perspective, bibliographie, table des illustrations
16/06/2022	1.9	Liste des abréviations, Introduction en anglais, Annexe

Remerciements

Tout d'abord, je tiens à remercier mes maîtres de stage, Andreas Zanzoni et Lionel Spinelli de m'avoir guidé durant mon stage et de m'avoir intégré dans l'équipe Network Biology au laboratoire TAGC de l'INSERM.

J'aimerais également remercier Sébastien Choteau et Mégane Boujeant de m'avoir partagé leur savoir et de m'avoir beaucoup aidé à la réalisation de ma mission et à résoudre certains problèmes rencontrés.

Je souhaite aussi remercier mon tuteur Yavar Kian, pour m'avoir conseillé pour la rédaction de mon rapport et de mon oral.

De plus, je tiens à remercier Mme Christine Brun pour m'avoir aussi bien accueilli et fait découvrir le domaine de la biologie en m'expliquant certains points techniques.

Enfin, je voudrais remercier l'ensemble des professeurs du département informatique de l'IUT Aix-Marseille pour m'avoir guidé durant ces deux années.

Fiche Technique

Étudiant : Kevin Maldonado

Année : 2022

Raison sociale de l'entreprise : Institut national de la Santé et de la Recherche Médicale

Maîtres de stage : Andreas Zanzoni, enseignant chercheur; Lionel Spinelli, ingénieur de recherche.

Tuteur : Yavar Kian

Mission : Finalisation de l'interface d'une application web pour la prédiction et l'analyse des interactions moléculaires entre microbes et cellules humaines. Les missions consistent à faire communiquer un outil bioinformatique avec une interface web.

Plateforme informatique et système :

Ubuntu, PostGreSQL, Docker

Outils et langages :

Python, framework Django, Shell

Web, Python, Bio-informatique

Sommaire

Gestionnaire de version	2
Remerciements	3
Fiche Technique	4
Sommaire	5
Introduction	6
Présentation de l'entreprise	7
l'INSERM : son histoire, son rôle et ses caractéristiques	8
L'unité de recherche TAGC	9
L'équipe Biologie des réseaux	9
Travail réalisé	11
Paragraphe introductif / mission	12
Matériels et outils logiciels	12
Django : de Python à un site web	12
Le workflow mimicINT	13
SLURM & Snakemake : Deux outils essentiels à l'exécution du workflow	13
Docker : La conteneurisation de logiciels	14
Sourcesup & Smartgit : Deux outils de versionnage de projet collaboratif	15
SCRUM et Trello : Une méthode pour organiser des travaux de manière agile	15
Grandes étapes du travail	16
Adaptation et apprentissage	16
Formulaire et arborescence du fichier	16
Exécution et vérification du workflow	18
Requête utilisateur pour la demande des résultats	21
Conclusion et perspective	22
Bilan du stage	24
Bilan de la formation	25
Conclusion	26
Liste des abréviations, acronymes et sigles	27
Bibliographie	28
Table des illustrations	29
Journaux de stage	30
Annexes	41

Introduction

De nombreux micro-organismes sont vecteurs de maladie pour l'Homme, causant des millions de morts à travers le monde. Afin de développer des traitements plus efficaces, il est nécessaire de comprendre et d'identifier les mécanismes moléculaires des interactions entre les microbes, tels que les bactéries et les virus, et leurs hôtes.

Les approches expérimentales peuvent être très coûteuses en ressources humaines et matérielles ainsi qu'en temps.

C'est pour cela que l'utilisation d'outils informatiques, notamment bioinformatiques, peut être un très bon moyen de prédire et modéliser ces interactions.

C'est dans ce contexte et dans une unité de recherche rattachée à l'*Institut National de la Santé et de la Recherche Médicale* (INSERM), qu'une approche bioinformatique a été développée pour inférer les interactions des protéines des microbes avec les protéines humaines.

La mission de mon stage, réalisé dans le laboratoire *Theories and Approaches of Genomic Complexity* (TAGC) de l'INSERM, consistait à finaliser une interface web permettant de faciliter l'utilisation d'un outil bioinformatique d'inférence d'interactions développé par l'équipe "*Biologie des réseaux*".

Many microorganisms are vectors of human diseases, causing millions of deaths worldwide. In order to develop more effective treatments, it's necessary to understand and identify the molecular mechanisms of the interactions between microbes, such as bacteria and viruses, and their hosts.

Experimental approaches can be very costly in terms of human and material resources, and time.

This is why the use of computational tools, especially bioinformatic tools, can be a very good way to predict and model these interactions.

It's in this context and in a research unit attached to the *Institut National de la Santé et de la Recherche Médicale* (INSERM), that a bioinformatic approach has been developed to infer the interactions between microbial and human proteins.

The mission of my internship, carried out in the laboratory *Theories and Approaches of Genomic Complexity* (TAGC) of the INSERM, was to finalize a web interface to facilitate the use of a bioinformatic tool for inferring protein interactions developed by the "*Biologie des réseaux*" team.

Présentation de l'entreprise

l'INSERM : son histoire, son rôle et ses caractéristiques

L'Institut national de la santé et de la recherche médicale (INSERM), fondé par Raymond Marcellin en 1964, est un établissement public consacré à la recherche biologique, médicale et à la santé humaine. L'INSERM a pour objectif d'améliorer la santé de tous en accumulant des connaissances sur le vivant ainsi que les maladies. Il a été l'acteur d'avancées médicales majeures depuis sa création et continue de jouer un rôle important, notamment sur la recherche des maladies infectieuses comme le COVID-19 par exemple.

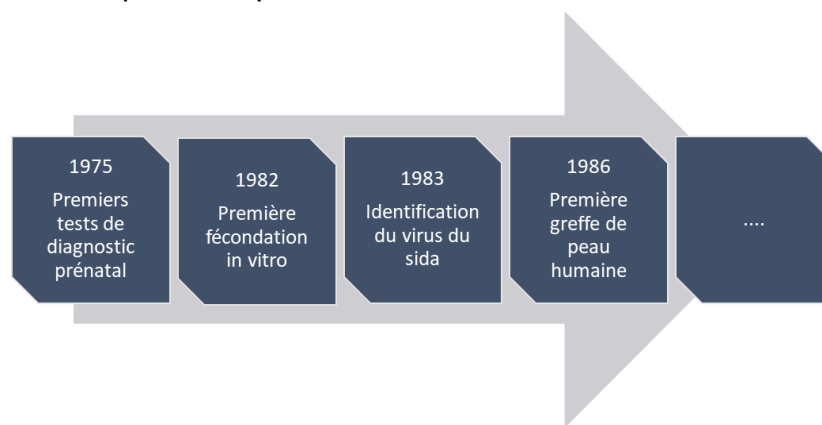


Figure 1 : Frise chronologique de quelques dates clés où l'INSERM a été un acteur majeur

Cet institut est constitué de nombreuses unités de recherche, avec 281 unités en 2017 réparties dans toute la France et compte près de 15 000 personnes travaillant au sein de ces structures cette même année. Chaque unité, souvent implantée dans des hôpitaux ou des campus universitaires, se consacre à la recherche d'une thématique liée à la santé ou à la biologie, permettant alors un plus grand champ d'exploration dans ces domaines vastes et complexes. À noter que l'INSERM travaille souvent sur des projets en partenariat avec d'autres infrastructures (hôpitaux, autres instituts de recherche étrangers).

Étant donné son grand nombre d'unités, mais aussi sa vision globale qui est animée par l'apprentissage et l'accumulation des connaissances, cette organisation met en place plusieurs moyens pour permettre aux scientifiques de toujours s'informer sur les nouvelles avancées. Outre les collaborations possibles entre différentes unités et les publications d'autres chercheurs, l'INSERM met en place un site web ainsi qu'un magazine mensuel pour toujours informer son effectif sur les travaux des autres employés.

L'unité de recherche TAGC

J'ai pu travailler dans l'une de ces unités, l'unité TAGC dont l'activité suit toujours le principe de la recherche en biologie. C'est un laboratoire, localisé dans le campus universitaire de Luminy, dont la thématique de recherche est centrée sur la biologie des systèmes et les maladies multifactorielles.

Les projets sont organisés autour de deux axes dans ce laboratoire :

- Le premier axe, nommé "Bioinformatique et génomique des réseaux moléculaires", qui vise à comprendre le réseau d'interactions moléculaires au sein d'une même cellule ou entre cellules via des outils bioinformatiques.
- Le deuxième axe, qui porte le nom "génétique et génomique des maladies multifactorielles", cherche à identifier le rôle et les mécanismes des gènes exprimés dans les cellules, conduisant à l'apparition de certaines maladies multifactorielles (malaria, sepsis ...).

Pour tenir informés les employés de l'avancement ou des découvertes des autres personnes travaillant au sein de l'unité, tous les mercredis se tient un séminaire scientifique au cours duquel un chercheur, un post-doctorant ou un doctorant présente son sujet d'étude.

Le laboratoire est composé de plusieurs équipes travaillant sur plusieurs projets (souvent en collaboration avec d'autres laboratoires) sur les deux axes de recherche évoqués précédemment.

L'équipe Biologie des réseaux

J'ai intégré l'équipe "Biologie des réseaux", qui est rattachée au premier axe de recherche du TAGC. L'objectif de cette équipe est de mieux comprendre les réseaux d'interactions moléculaires cellulaires, notamment les réseaux d'interactions protéine - protéine.

En effet, on ne connaît pas encore toutes les interactions possibles entre chaque protéine présente dans les cellules, car elles peuvent ne survenir que dans certaines conditions (type cellulaire, conditions de stress etc.), qui peuvent être totalement inconnues.

Tous les membres de l'équipe ne travaillaient pas forcément sur le même projet, c'est pour cela qu'il y avait tous les vendredis, une réunion avec l'ensemble des membres de l'équipe.

En premier lieu un tour de table était organisé, où chacun résumait ce qu'il avait fait dans la semaine.

Ensuite il suivait la présentation d'un projet par une personne, afin qu'elle puisse recueillir les avis et suggestions sur l'avancée scientifique du projet, mais aussi s'entraîner et s'améliorer dans sa présentation en vue d'un prochain séminaire ou d'une conférence scientifique.

Par ailleurs, l'un des projets sur lequel travaillait l'équipe et auquel j'ai participé, était un projet nommé *mimicINT*. Ce projet consistait à prédire le réseau d'interactions entre les protéines d'un microbe (bactérie ou virus) et les protéines d'une cellule hôte grâce à un outil bioinformatique.

Travail réalisé

Paragraphe introductif / mission

Au sein de l'équipe "Biologie des réseaux", un outil bioinformatique permettant de prédire le réseau d'interactions entre les protéines d'un microbe pathogène et les protéines de l'hôte a été développé.

Cet outil, nommé *mimicINT*, consiste en un enchaînement de tâches d'analyse, d'interprétation et de modélisation d'un réseau d'interactions entre protéines (appelé *workflow*).

Néanmoins, l'utilisation de *mimicINT* est assez difficile à prendre en main. Au sein de la communauté scientifique, notamment chez les biologistes, peu de personnes sont à l'aise en informatique. Ces personnes ne possèdent pas les aptitudes nécessaires pour pouvoir manipuler *mimicINT*, étant donné que son utilisation se faisait exclusivement en ligne de commande. Il fallait donc trouver un moyen de rendre plus accessible cet outil.

La solution adoptée pour répondre à cette problématique a été la réalisation d'un projet de site web nommé *mimicINTWeb*.

Au début de mon stage, l'aspect du site web était quasiment finalisé, mais plusieurs éléments fonctionnels permettant notamment de communiquer avec *mimicINT*, n'étaient pas encore développés.

Ma tâche consistait donc à :

- Finaliser l'aspect de certaines pages
- Assurer l'exécution du workflow de *mimicINT* à partir des données fournies par l'utilisateur dans l'interface et contrôler la bonne exécution du workflow
- Afficher les informations adéquates à la fin du workflow.
- Permettre à l'utilisateur d'accéder aux résultats ou à une information compréhensible si le workflow est en cours d'exécution ou a connu une erreur d'exécution

Matériels et outils logiciels

Django : de Python à un site web

Django est un framework Python, c'est-à-dire une bibliothèque d'outils informatiques permettant de faciliter le développement de sites web. Il permet entre autres de faciliter la séparation de l'interface web et des données pour un affichage dynamique.

Django permet aussi d'éviter l'utilisation du langage *SQL* et quelques balises *HTML* (comme les balises pour les formulaires par exemple), grâce à son fonctionnement dérivé du modèle *MVC* (Modèle, Vue, Contrôleur), le modèle *MVT* (Modèle, Vue, Template). Ces deux modèles servent de patron de conception, pour des interfaces graphiques par exemple.

Une base de données a aussi été utilisée pour le site web. *PostgreSQL* a été choisie comme système de gestion de base de données. Une interface graphique, nommée *Adminer*, a été déployée pour faciliter la manipulation de cette base de données.

Le workflow *mimicINT*

Un workflow est un enchaînement de processus d'analyse de telle sorte que le paramètre de sortie d'un processus sert de paramètre d'entrée au prochain. Il permet d'effectuer un ensemble de tâches d'analyse, aussi appelé règles, permettant de traiter une liste de protéines et quelques autres variables soumises par l'utilisateur.

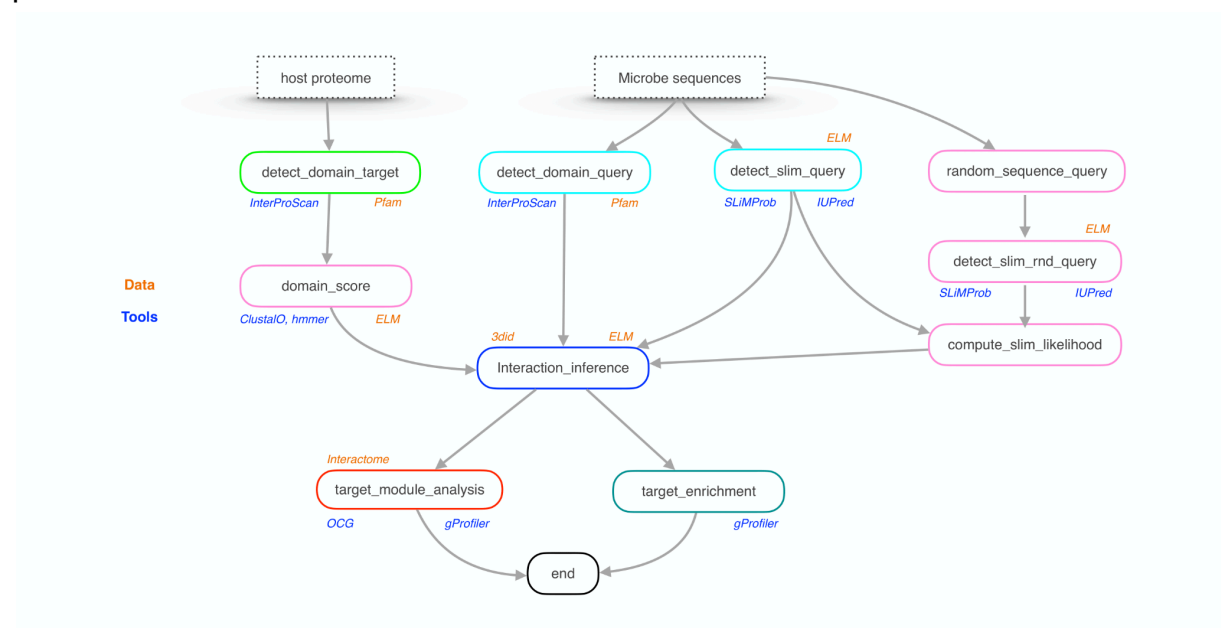


Figure 2 : Schéma des différentes tâches d'analyse du workflow *mimicINT*

SLURM & Snakemake : Deux outils essentiels à l'exécution du workflow

Le site web se basait sur un workflow pour analyser en arrière-plan les séquences de protéines soumises par l'utilisateur.

Pour la gestion des tâches du workflow, Snakemake a été utilisé. Il permet de gérer et formaliser un enchaînement de processus. Il assure donc le bon déroulement du

workflow, en assurant l'ordre d'exécution des processus et la traçabilité des erreurs survenues.

SLURM est un gestionnaire de ressources, il décide pour chacun des processus qui lui est soumis, quand l'exécuter en fonction des ressources physiques disponibles (CPU, RAM, ...) et de celles requises par l'exécution du processus. Il est nécessaire et s'allie très bien avec Snakemake car de nombreuses instances du workflow peuvent être en cours ou soumises simultanément, ce qui rend la gestion de ces ressources indispensable dans le cadre de l'application web.

Docker : La conteneurisation de logiciels

Docker est un logiciel permettant d'emballer une application avec toutes ses dépendances dans un conteneur. Il est utilisé dans le but d'isoler chacune des applications vu précédemment dans des conteneurs afin d'éviter des conflits entre elles, mais aussi pour faciliter le déploiement de ces applications. On peut voir Docker comme une alternative aux machines virtuelles bien que l'utilisation des ressources du serveur soit très différente.

Dans le cas du projet *mimicINTweb*, il existait 4 conteneurs :

- Le conteneur serveur web, qui possédait les dépendances nécessaires pour utiliser Django.
- Le conteneur PostgreSQL, qui s'occupait de la base de données
- Le conteneur SLURM & Snakemake, qui permettait d'utiliser le gestionnaire de jobs SLURM et d'exécuter le workflow via Snakemake.
- Un conteneur Adminer, qui facilitait la gestion de la base de donnée avec une interface graphique (non essentiel à l'interface web de *mimicINT*)

Grâce à un fichier `docker-compose.yaml`, on pouvait définir des interdépendances entre chaque conteneurs. En effet, il était possible de faire communiquer les conteneurs entre eux, permettant ainsi l'échange de données.

Néanmoins le conteneur SLURM & Snakemake ne pouvaient pas communiquer avec les autres conteneurs pour des soucis de sécurité étant donné que ce conteneur avait un accès privilégié aux ressources de la machine..

Pour permettre tout de même l'échange de données, on inscrit toutes les données qui doivent transiter dans des fichiers stockés sur l'ordinateur pour que les autres conteneurs puissent les interpréter.

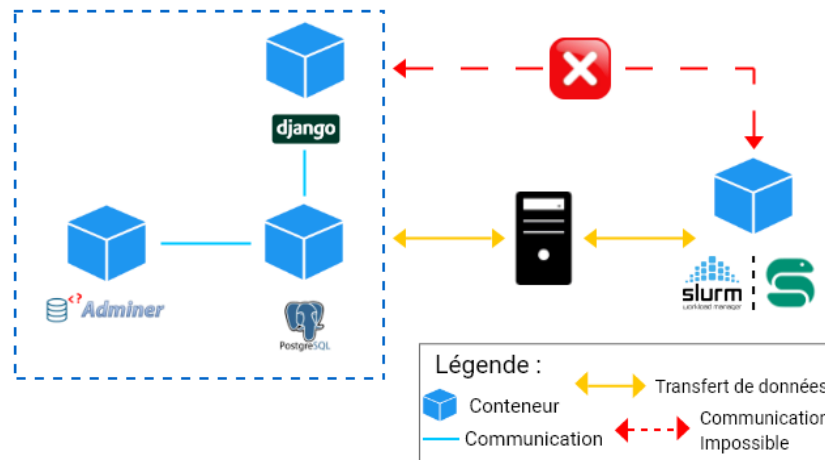


Figure 3 : Schéma des communications entre les conteneurs Docker

Sourcesup & Smartgit : Deux outils de versionnage de projet collaboratif

Sourcesup est une forge logicielle, c'est-à-dire un outil informatique permettant à plusieurs développeurs de participer ensemble au développement d'un projet commun. Cette forge est très similaire à GitHub dans son fonctionnement, néanmoins on utilise celle-ci car elle est destinée aux établissements d'enseignement supérieur et à la recherche française.

Smartgit est une interface logicielle permettant de faciliter la gestion du versionnage de code via Git. Il permet d'éviter l'utilisation de lignes de commande Git, qui peuvent être assez complexes dans certains cas, mais aussi d'éviter de potentielles erreurs de manipulation.

SCRUM et Trello : Une méthode pour organiser des travaux de manière agile

Tout au long de mon stage, le travail était organisé selon une méthode dite Agile appelée SCRUM.

Dans un premier temps, le *product owner* (ici mon maître de stage), avait défini un ensemble de tâches à réaliser pour finaliser le projet dans un carnet de produit. Il se pouvait que des tâches s'ajoutent au fil de l'avancement du projet.

Dans un deuxième temps, l'équipe se réunissait pour planifier pour la semaine à venir, les tâches à effectuer. Ces tâches étaient classées par ordre d'importance, et leur difficulté était évaluée par une durée, indiquant le temps nécessaire pour l'achever.

Ensuite, un sprint d'une semaine avait lieu, où je devais réaliser toutes les tâches définies lors de la réunion dans le temps imparti.

Enfin, à la fin du sprint, une autre réunion avait lieu pour passer en revue tous les travaux réalisés, et définir les tâches à réaliser pour le prochain sprint.

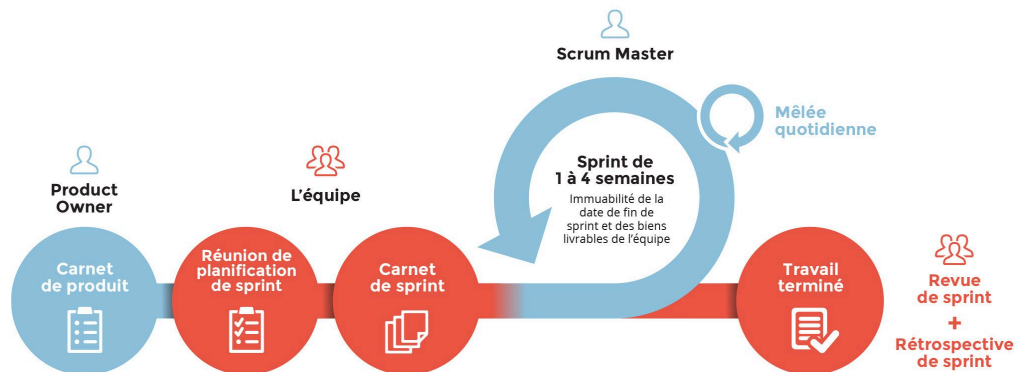


Figure 4 : Représentation de la méthode Agile SCRUM
(adapté à partir de <https://bubbleplan.net/blog/agile-scrum-gestion-projet/>)

Pour faciliter l'utilisation de cette méthode, une application web du nom de Trello a été utilisée. Elle permet de stocker toutes les tâches à réaliser afin de définir dans quel état elles sont considérées par l'ensemble des membres de l'équipe travaillant sur le projet *mimicINTweb*.

Grandes étapes du travail

Adaptation et apprentissage

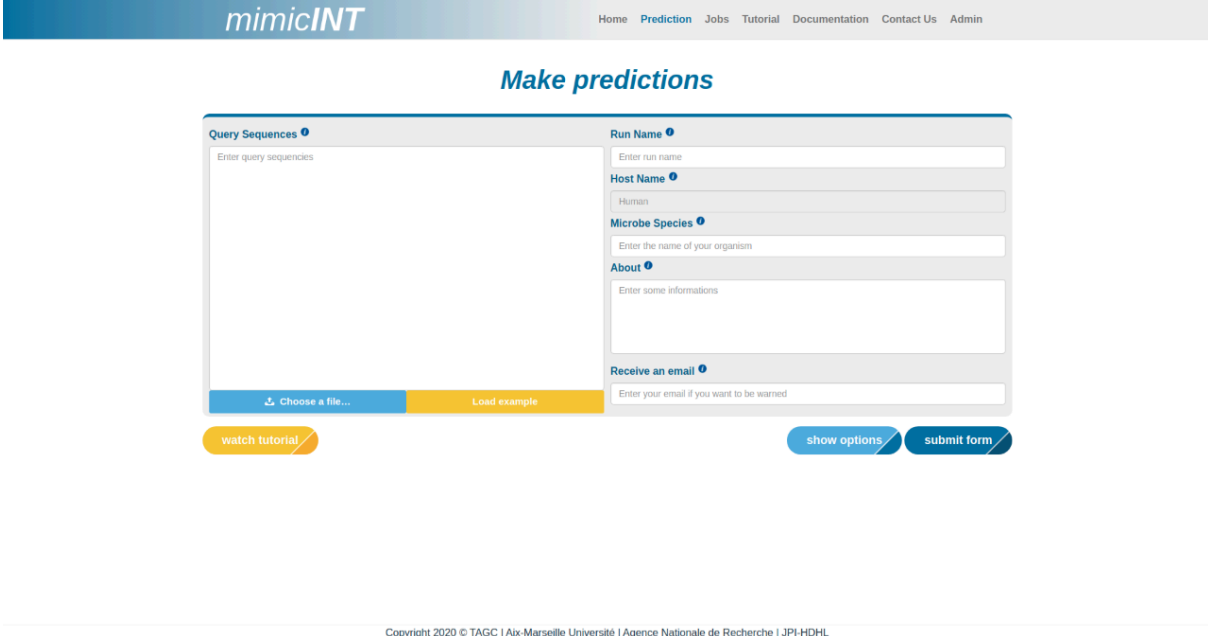
Les deux premières semaines étaient surtout des semaines d'adaptation où j'ai dû assimiler beaucoup de choses nouvelles pour moi en très peu de temps.

Tout d'abord, il m'a fallu me familiariser avec le code déjà existant en me renseignant avec les documentations écrites par les stagiaires qui avaient travaillé sur le projet avant. Ensuite j'ai dû assimiler les nouveaux outils utilisés dans le cadre du projet, comme Django et Docker qui m'ont demandé beaucoup d'exercice pratique.

Formulaire et arborescence du fichier

Mon premier travail assigné était de finaliser la page du formulaire dans laquelle les utilisateurs entrent leur données et les paramètres pour lancer *mimicINT*. L'aspect de la page était déjà réalisé, il s'agissait maintenant de peaufiner la page, mais surtout de communiquer les informations soumises par l'utilisateur au workflow pour qu'il

puisse générer un résultat à partir des données fournies et en créer une représentation graphique.



The screenshot shows the 'mimicINT' web application interface. At the top, there is a navigation bar with links: Home, Prediction, Jobs, Tutorial, Documentation, Contact Us, and Admin. Below the navigation bar, the main heading is 'Make predictions'. The form is divided into two main sections: 'Query Sequences' on the left and 'Run Name' on the right. The 'Query Sequences' section has a large text area for entering query sequences, a 'Choose a file...' button, and a 'Load example' button. The 'Run Name' section contains several input fields: 'Enter run name', 'Host Name' (with a dropdown menu showing 'Human'), 'Microbe Species' (with a dropdown menu), 'Enter the name of your organism', 'Enter some informations' (under the 'About' section), and 'Enter your email if you want to be warned' (under the 'Receive an email' section). At the bottom of the form, there are three buttons: 'watch tutorial', 'show options', and 'submit form'. The footer of the page contains the copyright information: 'Copyright 2020 © TAGC | Aix-Marseille Université | Agence Nationale de Recherche | JPI-HDHL'.

Figure 5 : Capture d'écran de la page du formulaire permettant la soumission de séquence de protéine

Néanmoins, comme expliqué auparavant, le conteneur permettant l'exécution du workflow (SLURM & Snakemake) ne pouvait pas communiquer avec les autres conteneurs pour des soucis de sécurité, il était complètement isolé des autres conteneurs.

Il fallait donc trouver un moyen pour transmettre les informations du formulaire du conteneur web au conteneur SLURM-Snakemake.

La solution a été d'enregistrer les informations du formulaire dans des fichiers, organisés dans une arborescence structurée afin que *mimicINT* puisse lire les informations utiles, les interpréter et fournir un résultat.

J'ai dû faire en sorte que lors de la soumission du formulaire, un dossier `run` (ensemble des fichiers liés à l'exécution du workflow) soit créé avec un identifiant généré aléatoirement (exemple : "EAzCXAZz9UU74fBYAfg") dans un dossier commun nommé `jobs`, où toutes les exécutions (répertoire de formulaire soumis) étaient stockées.

Ce répertoire devait contenir les informations fournies par l'utilisateur mais aussi des données complémentaires nécessaires à la bonne exécution du workflow d'analyse.

Ces dernières étaient rassemblées dans le dossier `common_files`.

Ces fichiers étant les mêmes pour tous les runs, ils étaient copiés vers les dossiers runs plutôt que de les recréer à chaque formulaire soumis.

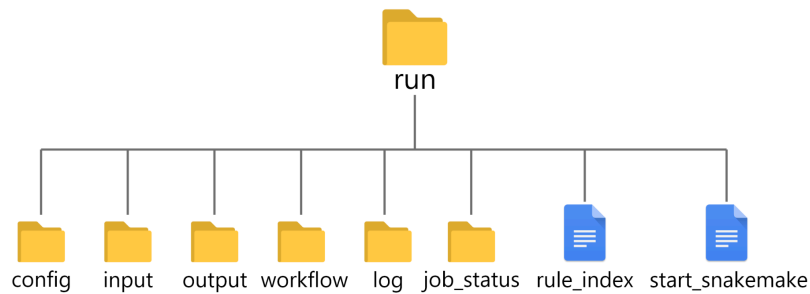


Figure 6 : Architecture simplifiée d'un répertoire `run`

Après avoir copié l'architecture et ces fichiers fixes, il fallait que je fasse en sorte que chacune des valeurs inscrites dans le formulaire aillent dans le bon fichier, et cela toujours lors de la soumission du formulaire.

Pour certaines valeurs il fallait créer un fichier spécifique, comme la séquence de protéines qui est stockée dans le dossier `input` au format FASTA (format de fichier permettant de stocker des séquences biologiques). Pour d'autres valeurs, il fallait modifier des fichiers copiés, comme le `config.yaml` dans lequel j'ai dû rajouter à la suite du fichier les options choisies par l'utilisateur.

Ensuite, après avoir défini l'architecture du répertoire, copié et créé les fichiers nécessaires pour le workflow, j'ai finalisé le répertoire en créant un fichier vide nommé `start_snakemake`. La présence de ce fichier déclenche l'exécution du workflow d'analyse grâce à un script lancé automatiquement par crontab toutes les minutes.

Exécution et vérification du workflow

Ma deuxième grande tâche a été de comprendre comment réagissait le workflow lors de son exécution afin de réaliser un script python permettant de vérifier l'état de chaque répertoire `run` et la bonne exécution des analyses.

Après la soumission du formulaire et la création de tous les fichiers dans le dossier, *mimicINT* était censé se lancer et donc créer encore d'autres fichiers, notamment dans les répertoires `input` et `output`.

Ces dossiers, comme leur nom l'indique, stockaient les fichiers d'entrées et de sorties de chaque processus du workflow.

Néanmoins, lorsque que le workflow était en cours, il n'y avait aucun moyen de savoir si celui-ci s'était bien déroulé, s'il était toujours en cours ou même juste savoir s'il s'était achevé. Il fallait donc créer un programme se lançant de manière périodique pour vérifier la bonne exécution du workflow pour chaque run et informer l'utilisateur en cas de problème.

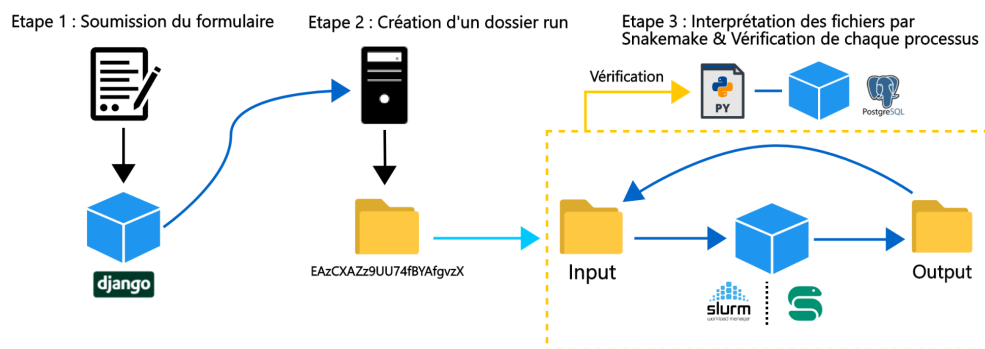


Figure 7 : Schéma représentant les trois étapes d'un run.

Pour réaliser cette tâche, j'ai dû tout d'abord concevoir un diagramme UML de l'algorithme pour avoir une représentation facile à lire, mais aussi afin de le faire valider rapidement aux membres de l'équipe avant de passer à son implémentation. Je devais pour mon diagramme, tenir compte de toutes les situations possibles lors de l'exécution du workflow. Il fallait donc analyser les erreurs possibles, savoir si ces erreurs étaient fatales ou non (si je devais arrêter l'exécution de celui-ci), et trouver un moyen de déterminer à quelle étape de l'enchaînement de processus était l'exécution du workflow.

Après avoir réalisé le diagramme UML (cf Annexe 1), j'ai développé ce programme de surveillance en me basant toujours sur cet algorithme.

Dans un premier temps, mon programme devait vérifier l'état de chaque dossier `run`. Grâce à une fonction permettant de vérifier l'existence ou non de certains fichiers, il était possible de déterminer si le dossier était "finalisé", "en cours" ou "corrompu".

- Un `run` était considéré comme finalisé lorsque le workflow avait effectué toutes ses tâches d'analyse et créer toutes les sorties requises ainsi qu'un fichier final indiquant la fin correcte de celui-ci.
- Un `run` était corrompu lorsqu'une erreur était survenue lors de l'exécution du workflow. Un fichier d'erreur permettait de déterminer l'erreur survenue. La

mise en valeur des erreurs dans un fichier était due à la deuxième partie du programme qui sera expliquée dans le prochain paragraphe.

- Un `run` était considéré en cours lorsqu'aucun des deux fichiers précédents (final ou erreur) n'existait.

Pour des soucis de performances, il fallait faire en sorte que la deuxième partie du programme ne s'exécute que si le `run` était considéré en cours, en dehors de ce cas, il vérifiait l'état du prochain dossier `run`.

Dans un deuxième temps, lorsque le `run` était considéré "en cours", une deuxième fonction se lançait. Les objectifs de cette fonction étaient de :

- déterminer à quel niveau d'exécution du workflow en est le dossier.
- vérifier si une erreur n'est pas survenue depuis la précédente exécution du programme de vérification (on rappelle que celui-ci se lance de manière régulière).

Pour répondre au premier objectif, la fonction devait parcourir les tâches d'analyse du workflow une à une pour vérifier si elles étaient finalisées.

Pour savoir quelles tâches il fallait vérifier, la fonction utilisait la base de données, dans laquelle se trouve une table des règles (tâches de d'analyse) du workflow. Dans cette table, il y a le nom de la règle, son ordre et sa durée d'exécution.

Lorsque la fonction savait quelles règles vérifier et dans quel ordre, elle pouvait déterminer grâce à 2 autres moyens, si la tâche d'analyse était finalisée ou non :

- Le premier moyen, étant l'existence ou non de fichier de début et de fin de règle dans le dossier `job_status`. Comme le nom de ces fichiers l'indiquent, ils servent à déterminer à quel moment a débuté et a fini la tâche d'analyse.
- Le deuxième moyen, est l'utilisation du fichier `slurm_squeue.txt`. En effet, ce document mis à jour automatiquement toutes les minutes, contient le résultat d'une requête effectuée à SLURM. Dans ce fichier, il y avait donc le nom du `run` suivi du nom de la règle et de son état (finalisé, en cours, en attente).

En combinant ces deux moyens dans cette fonction, l'algorithme pouvait déterminer via une double vérification à quel stade en est le workflow, mais aussi faire remonter les erreurs. Pour le deuxième objectif de la fonction (la vérification d'erreur), il suffisait de rajouter les erreurs qui peuvent survenir lors de l'exécution du workflow dans mon programme. Par exemple, il fallait prendre en compte, la durée d'exécution de chaque tâche d'analyse, qui ne doit pas excéder la durée définie. Cette durée était connue grâce à la table des règles.

Requête utilisateur pour la demande des résultats

Enfin, ma dernière tâche a été de gérer les requêtes utilisateurs sur le site web. Lors de la soumission du formulaire, l'utilisateur est informé de l'identifiant de son run, et du fait qu'il peut afficher l'état de son run *via* la soumission d'une requête sur une autre page.

J'ai dû faire en sorte que lorsque l'utilisateur demande les résultats de son run, une page soit affichée selon l'état du run.

Il existe trois états possibles pour le `run` :

- **Fini:** le répertoire possède un fichier indiquant la fin de l'exécution de *mimicINT*. Il contient aussi tous les fichiers nécessaires pour afficher une représentation des interactions protéines-protéines selon la séquence et variables fournies auparavant dans le formulaire. Dans cette situation, la page de résultats est affichée à l'utilisateur
- **Corrompu:** une erreur est survenue lors du déroulement du workflow, le dossier possède donc un fichier d'erreur. Dans cette situation, une page indiquant le type d'erreur et à quelle règle *mimicINT* a cessé de fonctionner est affichée à l'utilisateur. Les erreurs indiquées sont différentes de celles inscrites dans les fichiers `log`, car elles doivent être compréhensibles pour tout le monde.
- **En cours:** le `run` possède ni fichier de fin, ni fichier d'erreur. La page affiche une liste de description de tâches d'analyse. A côté de ces tâches, apparaissent des checkbox, indiquant si oui ou non la tâche a été finalisée.

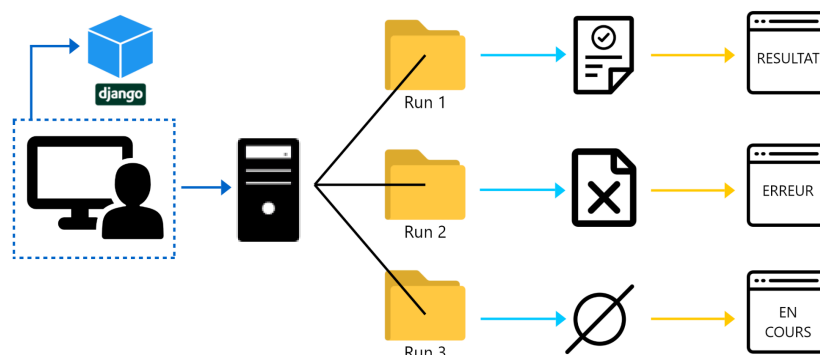


Figure 8 : Schéma représentant les différents types possibles de page de résultat

Conclusion et perspective

Grâce aux missions réalisées, j'ai pu répondre à la problématique liée au manque de communication entre l'interface web et le workflow.

Maintenant, il est désormais possible d'utiliser *mimicINT* directement depuis le site web, grâce à la page de formulaire qui envoie bien les informations nécessaires au workflow et grâce à la page de requête, qui affiche soit le résultat du workflow, soit l'erreur survenue ou alors à quel stade le workflow en est.

En perspective, il faudrait peaufiner l'aspect de certaines pages, pouvant être encore améliorées. L'équipe prévoit de mettre l'outil en production rapidement en le plaçant sur un serveur à accès public. Il est aussi prévu de publier un article dans un journal scientifique pour informer la communauté de l'existence de *mimicINTWeb*. Je serai cité parmi les auteurs de cet article.

Bilan et perspectives

Bilan du stage

Ce stage m'a permis de découvrir le monde professionnel dans le domaine de l'informatique. Ce fut pour moi ma première expérience dans ce domaine.

Concernant les missions que j'ai pu effectuer, elles étaient à la hauteur de mes compétences et de ce que j'étais capable de réaliser. Néanmoins, il m'est arrivé tout de même de rencontrer certaines difficultés, notamment sur les nouveaux outils qui ont nécessité un temps d'apprentissage.

Cependant grâce à ce temps d'adaptation, j'ai pu en apprendre davantage sur ces outils, mais j'ai aussi réussi à pouvoir les manipuler dans le cadre de mon stage.

Je suis assez fier de ce que j'ai pu réaliser durant ce stage, car je pense que ce que j'ai entrepris répond complètement aux exigences.

Au début de mon stage, je ne pensais pas devoir réaliser ce type de tâche. Je m'attendais vraiment à faire beaucoup de développement web alors que j'ai fait essentiellement de la programmation en python. Cela m'a beaucoup plu étant donné que j'étais plus intéressé par cet aspect de l'informatique.

De plus, avec du recul je trouve que les missions proposées suivent un niveau de difficulté croissant. En effet, je ne me suis jamais retrouvé devant une tâche insurmontable ou trop simple, j'ai constamment appris et évolué dans un cadre d'apprentissage et professionnel.

Cette période de stage m'a aussi beaucoup ouvert l'esprit sur la vie professionnelle. Tout d'abord, j'ai pu vivre pendant ces dix semaines, la culture du laboratoire, qui prônait la communication et la recherche constante de connaissance.

De plus, j'ai aussi pu travailler au sein d'une équipe expérimentée, qui a pu constamment me conseiller sur les différents aspects de l'informatique, notamment la différence entre ce que j'ai appris en cours et ce que je peux apprendre professionnellement.

Enfin, le fait d'avoir travaillé dans ce laboratoire, et avoir côtoyé le domaine de la biologie en même temps m'a permis de prendre plus de recul à la suite de mon parcours scolaire. J'ai eu une grande chance d'effectuer ce stage car j'ai eu l'opportunité de découvrir l'informatique appliquée à la biologie. Mais aussi car je m'intéressais avant ce stage, à effectuer un parcours bioinformatique plutôt orienté vers la neuroscience. J'ai donc pu être énormément conseillé par le personnel du laboratoire pour déterminer vers quel parcours m'orienter pour atteindre cet objectif.

Bilan de la formation

La formation proposée par l'IUT informatique m'a permis d'approfondir les bases que je possédais en informatique grâce à mon parcours précédent et mon intérêt pour ce domaine.

J'ai pu apprendre avec une grande efficacité grâce aux conditions de travail de l'IUT. En effet, les cours magistraux, expliquant l'aspect théorique des thèmes abordés, ainsi que les TD et TP, appliquant quant à eux la théorie expliquée via différents exercices, m'ont permis d'en apprendre davantage sur des facettes de l'informatique que je ne connaissais pas encore.

Les projets réalisés durant cette formation sont pour moi un bon moyen d'utiliser toutes les connaissances apprises lors des différents cours.

Mais aussi, je trouve qu'ils sont assez représentatifs de ce que j'ai pu voir du monde professionnel car ils permettent d'apprendre aux étudiants l'importance de la communication dans une équipe, et aussi envers le client pour satisfaire ses attentes.

Beaucoup de cours m'ont servi pour ce stage, que ce soit des cours d'informatique ou non. Je pense notamment aux TP de mathématiques qui nous ont initié aux langages python, mais aussi aux cours de gestion de projets qui nous ont appris l'existence et l'utilisation des méthodes agiles pour le développement d'un projet informatique. Bien sûr, j'ai énormément mobilisé les connaissances que j'ai acquises durant les cours de programmation et développement web.

Néanmoins, même si je n'ai pas pu exploiter l'ensemble des connaissances que j'ai pu apprendre durant cette formation, j'estime qu'ils sont nécessaires à celle-ci pour avoir une bonne vision du monde professionnel en informatique.

J'estime que cette formation propose de nombreuses voies à suivre à sa suite. En effet, grâce à celle-ci plusieurs parcours professionnels me sont maintenant accessibles, comme continuer les études en licence ou en école d'ingénieur, mais aussi d'entrer directement dans le monde du travail.

Conclusion

Dans le cadre du quatrième semestre de ma formation en IUT informatique, j'ai dû réaliser un stage de dix semaines.

L'unité de recherche TAGC, rattachée à L'INSERM, m'a accueilli pour faire ce stage afin de finaliser le développement de l'application web *mimicINTWeb*.

Cette application web a pour but de délivrer à la communauté scientifique un outil permettant de prédire les interactions entre les protéines d'un microbe et les protéines d'une cellule hôte et qui soit simple d'utilisation.

Mes missions consistaient dans une grande majorité à faire communiquer le workflow *mimicINT* et l'interface web *mimicINTWeb*.

Pour cela, j'ai principalement utilisé le langage python, que ce soit lors de l'élaboration de programme, ou du développement web grâce au framework Django.

Les tâches étaient organisées de manière méthodique grâce à une approche Agile et des réunions régulières permettant un meilleur suivi du développement du projet.

Ce stage finalise mes deux années en IUT informatique. Je souhaite à l'avenir effectuer une licence 3 en informatique pour éventuellement poursuivre en master CMB (Computational and mathematical biology) orienté en bioinformatique, ou en master MaSCo (Master in cognitive science) centré en neurologie et informatique.

Liste des abréviations, acronymes et sigles

TAGC : Theories and Approaches of Genomic Complexity

INSERM : Institut National de la Santé de la Recherche médicale

SQL : Structured Query Language

HTML : HyperText Markup Language

MVC : Modèle Vue Contrôleur

MVT : Modèle Vue Template

CPU : Central Processing Unit

RAM : Random Access Memory

UML : Unified Modeling Language

CMB : Computational and Mathematical Biology

MaSCo : Master in Cognitive Science

Bibliographie

- Documentation de Snakemake
<https://snakemake.readthedocs.io/en/stable/>
- Documentation de Docker
<https://docs.docker.com/>
- Documentation de Bootstrap
<https://getbootstrap.com/docs/5.2/getting-started/introduction/>
- Documentation de SLURM
<https://slurm.schedmd.com/documentation.html>
- Documentation de Python
<https://docs.python.org/fr/3/>
- Documentation de Django
<https://docs.djangoproject.com/fr/4.0/>
- Site de la forge sourcesup
<https://sourcesup.renater.fr/>
- Site OpenClassroom
<https://openclassrooms.com/fr/>
- Site de l'INSERM
<https://www.inserm.fr/>
- Source de la Figure 4
<https://bubbleplan.net/blog/agile-scrum-gestion-projet/>
- DRETS Lilian, *finalisation de l'application mimicINTweb*, 2021 (rapport de stage, IUT - Département informatique - Aix-Marseille Université)
- CRISTIANINI Marceau, *développement d'une application web dédiée à l'inférence d'interactions hôte-microbiote*, 2020 (rapport de stage, Master en Bioinformatique - Développement de Logiciels et Analyse de Données, Aix-Marseille Université)

Table des illustrations

Figure 1 : Frise chronologique de quelques dates clés où l'INSERM a été un acteur majeur	8
Figure 2 : Schéma des différentes tâches d'analyse du workflow <i>mimicINT</i>	13
Figure 3 : Schéma des communications entre les conteneurs Docker	15
Figure 4 : Représentation de la méthode agile SCRUM	16
Figure 5 : Capture d'écran de la page du formulaire permettant la soumission de séquence de protéine	17
Figure 6 : Architecture simplifiée d'un répertoire run	18
Figure 7 : Schéma représentant les étapes d'un run	19
Figure 8 : Schéma représentant les différents types possibles de page de résultats	22

Journaux de stage

Journal de stage - Semaine 1

Adaptation à l'environnement professionnel

Chapeau / Commentaire

Entreprise : L'INSERM (Institut National de la Santé et de la Recherche Médicale) est un établissement public spécialisé dans la recherche biologique et médicale créé en 1964. Je travaille dans le laboratoire TAGC (*Theories and Approaches of Genomic Complexity*) situé à Marseille (Luminy), dans l'équipe *Network Biology* axée en bio-informatique.

Projet : *MimicINTWeb* est une application web destinée à la communauté scientifique permettant de prédire les interactions d'une protéine pathogène avec les protéines de l'hôte. Le projet n'est pas encore finalisé.

Réunion :

- **13/04/2022 à 10h30 :** visite de prévention sur les risques et les gestes à avoir dans le laboratoire.
- **13/04/2022 à 14h :** réunion avec l'équipe Network Biology travaillant sur *MimicINTWeb* pour définir les prochaines réunions hebdomadaires de revue de projet, et à la suite de ça des explications détaillées du projet.
- **15/04/2022 à 10h :** réunion avec toute l'équipe avec un tour de table (chacun explique ce qu'il a fait dans la semaine), et la présentation d'un projet biologique par un des membres.

Tâches effectuées

- Découverte du laboratoire et de l'équipe.
- Formation sur le framework Django grâce à OpenClassroom.
- Documentation sur Django, Docker et les anciens rapports de stage.
- Compréhension du code fourni par les anciens stagiaires.

Difficultés rencontrées

- Manque de connaissance concernant les logiciels mis en place.

Objectifs prévisionnels

- Manipuler le code déjà existant.

Bilan

- Une semaine de découverte et d'apprentissage sur les bases déjà établies.

Journal de stage - Semaine 2

Initiation à la programmation du projet

Chapeau / Commentaire

Workflow : Un workflow est un flux de processus, ayant des paramètres d'entrées et sorties, où un processus peut prendre en entrée les paramètres de sortie d'un autre processus. Il en existe déjà un pour interpréter les interactions entre protéines.

SLURM : SLURM est un gestionnaire de jobs, il est utilisé pour gérer les différentes étapes du workflow (savoir où en est le processus selon les jobs effectués).

Docker : Docker est un logiciel permettant d'empaqueter une application avec toutes ses dépendances dans un conteneur. Nous avons pour le projet 3 conteneurs principaux :

- Django, correspondant au site web.
- PostgreSQL, correspondant à la base de données.
- SLURM.

Pour des soucis de sécurité, le conteneur SLURM ne communique avec aucun des autres conteneurs (contrairement à Django et PostgreSQL qui peuvent communiquer entre eux), la solution pour résoudre ce problème est d'enregistrer les informations fournies par le formulaire du site web dans des fichiers, pour qu'ensuite SLURM et le workflow les interprète.

Réunion :

- **22/04/2022 à 10h** : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- **22/04/2022 à 14h** : réunion hebdomadaire sprint planning (méthode agile SCRUM) pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Rédaction d'un document expliquant l'arborescence du projet.
- Ajout dans le formulaire des options modifiables par l'utilisateur.
- Amélioration du formulaire (vérification des inputs, aspect de la page, ...).
- Récupération des valeurs du formulaire pour les enregistrer dans un fichier.

Difficultés rencontrées

- Manque de connaissance du framework bootstrap.

Objectifs prévisionnels

- Réaliser les tâches présentées lors de la réunion sprint planning.

Bilan

- Une initiation à la programmation du projet à rythme plutôt lent qui permet une bonne assimilation des connaissances apprises la semaine dernière.

Journal de stage - Semaine 3

Programmation à un rythme plus soutenue

Chapeau / Commentaire

Choix méthodologique : Depuis la semaine 2, la gestion et l'encadrement du projet sont organisés de manière agile. En effet, la méthode SCRUM est utilisée pour le développement de MimicINTWeb, cela permet entre autres de pouvoir attribuer grâce au sprint planning, des tâches pour le prochain sprint (période de développement entre les réunions), mais aussi de passer en revue les tâches accomplies et revoir certains problèmes.

listener : Un listener est un script qui vérifie chaque minute le fichier run pour savoir s'il y a un fichier start_snakemake, s'il existe le workflow est exécuté et le fichier est supprimé pour éviter que workflow se relance.

lien symbolique : Un lien symbolique est un fichier qui pointe vers un autre fichier (au sens de Linux). Au lieu de copier le fichier, on va plutôt créer une sorte de raccourci qui redirige vers le fichier original. C'est utile lorsque le fichier est un dossier trop lourd par exemple.

Réunion :

- 28/04/2022 à 14h : réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.
- 29/04/2022 à 10h : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.

Tâches effectuées

- Ajout des fichiers et dossiers nécessaires au lancement du run.
- Création d'un lien symbolique afin de permettre l'exécution des jobs.
- Familiarisation avec SLURM et les listeners.
- Correction du formulaire et de l'enregistrement des variables.
- Correction du document concernant l'arborescence du projet.
- Création d'un diagramme d'un algorithme vérifiant les différents états d'un processus (en attente, en cours, finis) et s'il ne dépasse pas la durée définie.

Difficultés rencontrées

- Problème au niveau de Docker, manque de certaines dépendances pour lancer le workflow.

Objectifs prévisionnels

- Continuer le diagramme et faire le script python de l'algorithme.

Bilan

- Une semaine à rythme un peu plus soutenue et avec un peu plus de difficulté.

Journal de stage - Semaine 4

Création de mon premier script

Chapeau / Commentaire

Description détaillée des tâches effectuées : La semaine entière a été consacrée à la création de ce script. Contrairement aux autres tâches effectuées auparavant, je ne me base pas sur du code existant et je dois créer de zéro le script. L'objectif de celui-ci se découpe en plusieurs étapes :

- Il doit tout d'abord vérifier l'état de chaque dossier run (en cours, achevé ou s'il y a eu une erreur).
- Ensuite, si le dossier run est en cours, il vérifie l'état de chaque processus du workflow. Il vérifie que chaque processus respecte une certaine durée, mais aussi qu'il n'y a pas d'erreur, s'il y a bien une erreur, un fichier d'erreur est créé. Si le workflow s'est déroulé sans problème, un fichier de fin est créé.
- À noter que le script ne fait pas toutes les étapes de vérification de processus si le dossier run est considéré comme fini ou s'il y a eu une erreur.

Réunion :

- **06/05/2022 à 11h :** réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- **06/05/2022 à 14h :** réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Finalisation du diagramme de l'algorithme vérifiant les différents états des processus et dossier run.
- Développement du script python se basant sur le diagramme.

Difficultés rencontrées

- Quelques difficultés de connexion en essayant de se connecter à la Base de données (mais très vite résolu).
- Problème de droit concernant la création et la lecture de fichier.

Objectifs prévisionnels

- Continuer et optimiser le script python.

Bilan

- Première semaine où j'ai entrepris la création d'un script sans base derrière. C'était une semaine plutôt chargée avec beaucoup de réflexion.

Journal de stage - Semaine 5

Continuité de la réalisation du script

Chapeau / Commentaire

Description détaillée des tâches effectuées : mon script fonctionnait plutôt bien, il me manquait juste à l'optimiser et à surtout gérer les exceptions. En effet, les exceptions permettent d'avoir une trace des erreurs possibles et aussi d'exécuter certaines fonctions par exemple selon l'erreur.

Au début, mon script ne gérait pas les exceptions, ce qui peut poser énormément problème, car mon code dépendait de beaucoup d'éléments extérieurs à celui-ci (l'état des conteneurs docker, l'existence de fichier pour lire les informations à l'intérieur, exécution de commande shell ...).

Il m'a fallu donc tout d'abord repérer dans mon algorithme les endroits dépendants d'éléments extérieurs, ensuite trouver le type d'erreur associé à l'élément.

Exemple : une erreur `FileNotFoundException` car le conteneur Docker SLURM a crash et a pu créer un certain dossier.

Enfin, il s'agissait de stocker le log de l'erreur dans un fichier et d'éventuellement essayer de résoudre l'erreur directement dans l'algorithme.

Réunion :

- **12/05/2022 à 11h :** réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- **12/05/2022 à 14h :** réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Optimisation du script et prise en charge des exceptions possibles.

Difficultés rencontrées

- Manque de connaissance sur la gestion des exceptions.

Objectifs prévisionnels

- Commencer la rédaction du rapport de stage.
- Création d'une page web permettant d'afficher les erreurs du workflow à l'utilisateur.

Bilan

- Une semaine dans la continuité du travail effectué la semaine précédente.

Journal de stage - Semaine 6

Commencement de la rédaction du rapport

Chapeau / Commentaire

Description détaillée des tâches effectuées :

Cette semaine, je me suis beaucoup consacré à la rédaction du rapport. J'ai préféré commencé par la partie présentation de l'entreprise, j'ai donc alors fais beaucoup de recherches sur l'histoire de l'INSERM, ses caractéristiques, etc...

J'ai aussi essayé de montrer au plus possible la culture de l'entreprise derrière.

De plus, j'avais une autre tâche cette semaine qui était de faire une page d'erreur pour tous les runs auxquels une erreur était survenue selon mon script élaboré les semaines précédentes.

Réunion :

- 20/05/2022 à 10h : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- 20/05/2022 à 11h30 : réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Rédaction du rapport de stage.
- Finalisation du script.
- Ajout d'une page d'erreur pour informer l'utilisateur que le workflow a échoué.

Difficultés rencontrées

- Quelques difficultés avec l'assimilation du framework bootstrap.

Objectifs prévisionnels

- Faire une page pour informer l'utilisateur d'où en est la progression du workflow s'il n'est pas considéré comme fini ou s'il n'y a pas d'erreur.
- Continuer de rédiger le rapport.

Bilan

- Une semaine avec beaucoup de recherche pour le rapport et la page d'erreur.

Journal de stage - Semaine 7

Continuité de la rédaction du rapport

Chapeau / Commentaire

Réunion : Semaine de 3 jours, les réunions hebdomadaires n'ont donc pas eu lieu cette semaine.

Tâches effectuées

- Rédaction du rapport de stage.
- Réalisation d'une page web récapitulant où en est le workflow.

Difficultés rencontrées

- Beaucoup de difficultés rencontrées lors de la réalisation de la page (connexion à la base de données, aspect de la page ...).

Objectifs prévisionnels

- Continuer la rédaction du rapport.
- Finaliser la page web.
- Faire des tests pour vérifier le bon fonctionnement du script réalisé les semaines précédentes avec la version finale du workflow.

Bilan

- Une semaine beaucoup orientée sur la rédaction du dossier.

Journal de stage - Semaine 8

Continuité de la rédaction du rapport

Chapeau / Commentaire

Réunion :

- 03/06/2022 à 10h : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- 03/06/2022 à 11h30 : réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Rédaction du rapport de stage.
- Finalisation de la page web récapitulatif où en est le workflow.
- Réalisation de tests concernant le bon fonctionnement du script réalisé les semaines précédentes.

Difficultés rencontrées

Objectifs prévisionnels

- Continuer la rédaction du rapport.
- Continuer les tests concernant le bon fonctionnement du script.

Bilan

- Une semaine toujours centrée autour de la rédaction du dossier.

Journal de stage - Semaine 9

Continuité de la rédaction du rapport

Chapeau / Commentaire

Réunion :

- 10/06/2022 à 10h : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.
- 10/06/2022 à 14h : réunion hebdomadaire sprint planning pour mettre au clair les tâches à effectuer et l'avancement du projet.

Tâches effectuées

- Rédaction du rapport de stage.

Difficultés rencontrées

Objectifs prévisionnels

- Finaliser le rapport de stage.
- Préparer l'oral du stage.

Bilan

- Une semaine encore une fois centré autour de la rédaction du rapport de stage.

Journal de stage - Semaine 10

Semaine finale du stage professionnel

Chapeau / Commentaire

Entraînement pour l'oral du stage :

- **13/06/2022** : premier entraînement avec quelques membres de l'équipe. L'objectif était de bien établir l'architecture du support visuel.
- **17/06/2022** : deuxième entraînement pour vérifier le contenu préparé durant cette semaine.

Réunion :

- **17/06/2022 à 10h** : réunion hebdomadaire en anglais avec toute l'équipe avec un tour de table et la présentation d'un projet biologique par un des membres.

Tâches effectuées

- Finalisation du rapport de stage.
- Préparation de l'oral.

Difficultés rencontrées

Objectifs prévisionnels

Bilan

Cette semaine était encore axée sur la finalisation du rapport, mais aussi sur l'entretien oral qui aura lieu par la suite. Ce journal clôt la série de journaux de stage. Cette expérience a été extrêmement bénéfique pour moi, que ce soit au plan personnel ou professionnel.

Je tiens encore à remercier l'ensemble des membres de l'équipe "biologie des réseaux" pour m'avoir guidé tout le long de ces 10 semaines.

Finalisation de l'application mimicINTWeb Rapport de stage Annexes

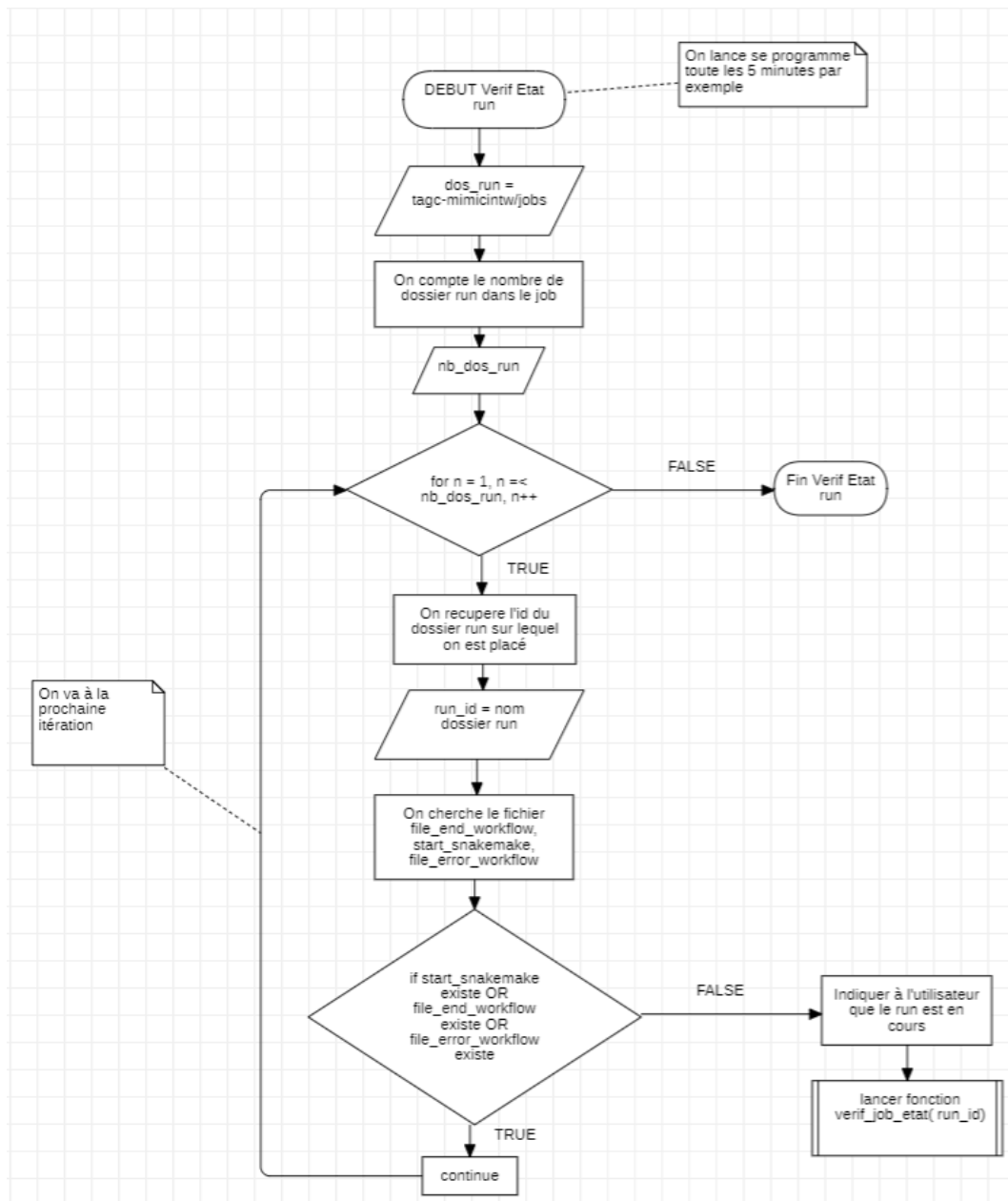
Sommaire

Programme de surveillance	44
Diagramme UML	44
Première parti du diagramme : verif état run	44
Deuxième parti du diagramme : verif_job_etat	45

Programme de surveillance

Diagramme UML

Première parti du diagramme : verif état run



Deuxième parti du diagramme : verif_job_etat

