



**UNIVERSITÉ
DE GENÈVE**

FACULTY OF SCIENCE
Department of Informatics

Projet en nouvelles technologies de l'information et de la communication

Silvio Fossati, (Silvio.Fossati@etu.unige.ch)

Amine Hadjiat, (Ahmed-Amine.Hadjiat@etu.unige.ch)

Sebastien Chassot, (Sebastien.Chassot@etu.unige.ch)

Hemoglobin Variants (HbVar)

Cours Master en Sciences Informatiques

Semestre Printemps 2021

Table des matières

1	Introduction	3
2	Cahier des charges	4
2.1	Page administration des paramètres d'analyse	4
2.2	Section "MS3" pour le détail d'une expérience	4
2.3	Section "Variants" pour le détail d'une expérience	5
2.4	Organisation	5
3	Infrastructure du projet	6
3.1	Infrastructure du Backend	6
3.2	Plan du site Web	7
3.2.1	Expériences et résultats	7
3.2.2	Admin	8
3.3	Diagrammes des cas d'utilisation	8
4	Outils NTIC - Frontend	10
5	Description de la nouvelle interface utilisateur	11
5.1	Page admin	11
5.2	Section MS3	12
5.3	Section des variants	14
6	Validation de l'UI par les utilisateurs	16
6.1	Validation de la page Admin	16
6.2	Validation de la page MS3	16
6.3	Validation de la page Variants	17
7	Difficultés rencontrées	18
7.1	Débogage	18
7.2	Table utilisée pour les variants	18
7.3	Mise à jour de l'UI	19
8	Conclusion	20
9	Annexes	22
9.1	Listes des mockups	22

1 Introduction

Dans le cadre de sa thèse de doctorat, le Dr. Didia Coelho Graça[1] a développé une méthode de diagnostique des mutations de l'hémoglobine basé sur la spectroscopie de masse[2]. Son travail de recherche a été affiné et est depuis utilisé aux HUG par des chercheurs et des médecins. Le diagnostique est basé sur un examen manuel des résultats de l'analyse spectroscopique. Cet examen suit une procédure systématique et prend plusieurs jours.

Sur mandat des HUG, l'HEPIA a développé en 2019 une application en ligne de commande afin d'automatiser le processus d'analyse. L'application produit une grande quantité d'information sous forme de fichier JSON difficile à visualiser. Bien que répondant aux attentes, le format des résultats les rendent difficile à interpréter. Sur demande des utilisateurs un prototype de frontend a été développé afin de les aider dans leur tâche de diagnostique. Ce prototype est actuellement en production.

Une seconde version complètement réécrite est dans sa phase finale de développement. L'architecture a été complètement repensée pour tourner en containers. Il est constitué d'une librairie pour la partie analyse, d'un parser écrit en JAVA qui interroge une base de donnée publique de mutations connues[3], d'une API pour servir les résultats et d'une ébauche de frontend.

2 Cahier des charges

La première version du frontend a été réalisée en 2019 dans le cadre d'un travail de Bachelor par un étudiant de l'HEPIA. Développé en quelques semaines, certaines fonctionnalités ne répondaient pas complètement aux besoins des médecins et des chercheurs.

Il avait été demandé de pouvoir changer les paramètres du backend lors du lancement d'une nouvelle analyse. À l'utilisation, ce choix s'est révélé être de peu d'intérêt tout en risquant d'amener à des erreurs. Une première demande était de retirer ces options et de les migrer sur une page d'administration à accès restreints.

Concernant la partie la plus utilisée, la visualisation des résultats d'analyse, l'information utile n'était pas toujours mise en avant alors que certains détails l'étaient. Les points à améliorer étaient clairement identifiés par les utilisateurs mais nous avions une grande liberté quant aux solutions à apporter.

2.1 Page administration des paramètres d'analyse

La page "Admin" correspond à une nouvelle fonctionnalité demandé par rapport à l'ancien site. En effet, il s'agit d'une demande qui a été formulée pour pouvoir paramétrer les expériences.

- Réorganisation global de la page selon mockup (cf. Section 9).
- Ajouter deux boutons pour envoyer un fichier json, l'un pour la partie "ions" et le second pour les valeurs de temps de rétention des variants.
- Récupération des paramètres du serveur via une requête à l'API, modification par l'utilisateur et pouvoir les renvoyer au serveur.
- Nécessite des chemins "GET" et "PUT" de l'API.

2.2 Section "MS3" pour le détail d'une expérience

La section "MS3" était affichée sous forme de colonne avec beaucoup d'informations, sous forme de liste. Cela n'était pas pratique car l'utilisateur devait dérouler toute la page. Dans le cadre de ce projet, nous avons développé une nouvelle version plus compacte.

- Réorganisation global de la page selon mockup (cf. Section 9).

- Représentation des chaînes protéiques par une suite de lettres ¹, et le plot correspondant en dessous (via collapse).
- Les positions des charges : "c" sont au dessus de la lettre; "z" en dessous.
- La charge affiche une valeur et une couleur (trouvé ou pas), la bordure est épaisse pour nous informer s'il est "major".
- Passer la souris sur une charge révèle : position; m/z.

2.3 Section "Variants" pour le détail d'une expérience

La section "Variants" n'affichait que les noms des mutations. Désormais, nous affichons plus de détails : la masse du variant, la charge, le temps de rétention, etc.

- Une liste de variants par précurseurs ².
- Pour chaque variant afficher toutes les informations connues; nom de la mutation, état de charge, masse de la mutation (moyenne ou monoisotopique), description de la mutation.
- Un filtre global (dit HPLC) par temps de rétention ³.
- Un moyen de mettre en évidence l'état de charge d'un variant.
- Pouvoir sauvegarder la liste filtrée en .csv.

2.4 Organisation

Dans un premier temps, nous avons débuté en prenant le temps de comprendre les outils nécessaires, surtout par rapport à React, grâce au tutoriel disponible sur le site officiel [\[4\]](#).

Après cela, nous avons défini l'ordre des tâches pour commencer avec un objectif simple, puis nous avons mis en pratique ce que nous avons appris.

Nous voulions démarrer par du "pair programming" pour mettre notre savoir des outils en commun, et pour nous permettre de prendre en main le projet, mais très vite nous avons continué avec cette méthode, qui nous a permis d'être efficace et résoudre les divers problèmes rencontrés en groupe.

La construction du rapport - qui sera également la présentation - est fait en groupe d'après la mise en page trouvée sur le cours. Cette fois-ci, nous avons privilégié la division du groupe pour mieux rédiger, étant donné que nous avons travaillé préalablement tous ensemble.

1. Chaque lettre représente l'acide aminé correspondant dans la protéine.

2. Les précurseurs sont les différentes protéines fragmentées lors de la phase MS3.

3. Les variants apparaissent à des temps connus, compris entre 0 et 6 minutes lors de la phase MS3.

3 Infrastructure du projet

Dans cette section nous présentons l'infrastructure du projet, dans laquelle nous passons en revue les différentes pages du frontend.

3.1 Infrastructure du Backend

Dans la première version de l'application⁴, une partie du traitement était effectué par le frontend en appelant directement des scripts bash et python. L'architecture modèle-vue-contrôleur[5] n'était pas respectée et le code difficile à maintenir. Il a été décidé de rendre le backend autonome et de fournir une API qui pourrait être à l'avenir utilisée par plusieurs applications. Ainsi, le frontend sur lequel nous avons travaillé est indépendant et ne peut plus avoir d'effet de bord sur le backend comme c'était le cas auparavant.

Le backend est composé de trois containers qui peuvent être utilisés par plusieurs frontend ou GUI :

- compassX : Bruker[6] fournit un utilitaire pour convertir les fichiers yep dans le format ouvert mzML. Malheureusement seule une version pour Windows existe. Un container Ubuntu avec Wine⁵ et Flask⁶ permet d'envoyer un fichier et récupérer le fichier converti.
- xtract : Il s'agit d'une application développée par deux étudiantes du Bachelor en sciences informatiques qui récupère l'ensemble des mutations connues sur une base de données publique et retourne les mutations compatibles avec l'analyse en cours.
- le backend : Les résultats d'expériences sont sérialisés en JSON lors du processus d'analyse. L'API permet de lancer une analyse aussi bien que récupérer les résultats.

4. Écrit en python et en javascript.

5. <https://fr.wikipedia.org/wiki/Wine>

6. <https://flask.palletsprojects.com/en/2.0.x/>

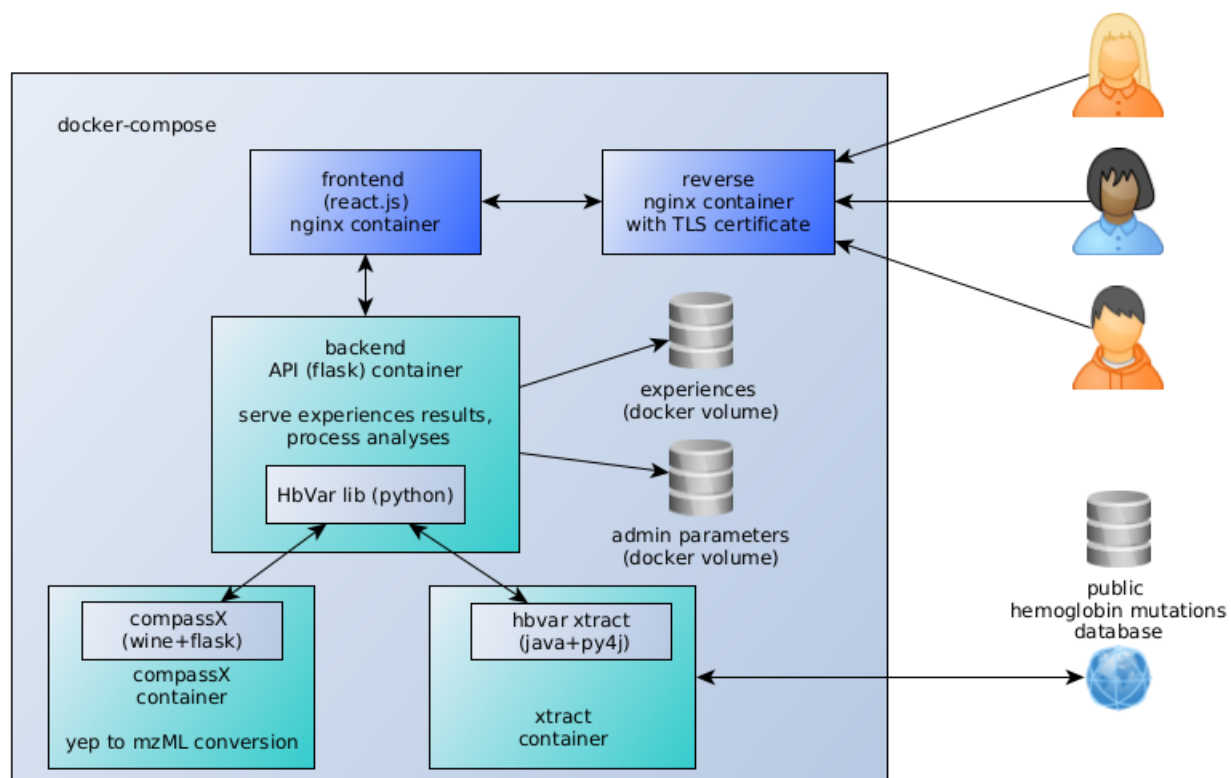


FIGURE 1 – infrastructure docker

3.2 Plan du site Web

3.2.1 Expériences et résultats

La section qui concerne les expériences comporte deux pages :

- "Run an Analysis" : où l'on peut sélectionner un fichier au format yep ou mzML⁷ et l'importer en tant que nouvelle expérience.
- "Results" : où sont affichées toutes les expériences lancées par les utilisateurs. Cf. Figure 2 pour les différentes actions possibles que l'on peut effectuer depuis cette page.

Ainsi, un ordre logique d'utilisation consiste à se connecter avec un compte utilisateur, aller dans la section "Run an Analysis" pour importer une expérience, puis être redirigé dans la page "Results" pour visualiser les différentes expériences, notamment celle qui

7. Le format yep est le format propriétaire des spectroscope Bruker[6] et nécessite une conversion côté backend dans le format ouvert mzML

vient d'être importée.

Depuis la page "Results", il est possible de sélectionner une expérience pour consulter en détail ses paramètres de l'analyse et de Xtract, les pics détectés, MS1, MS3 et les variants. Toutes ces informations sont affichées depuis une nouvelle page qui se charge juste après avoir sélectionné l'expérience.

Notons que toute la communication se fait via des routes depuis la partie backend :

- une route *get* et *put* pour les paramètres par défaut avec laquelle les expériences sont lancées sur le serveur.
- une route *put* pour les renvoyer.
- route *ion MS3* qui renvoie la liste de tous les ions.
- une liste qui renvoie les variants.
- une route pour les plots.

3.2.2 Admin

Pour y accéder, il faut se connecter avec un compte administrateur, c'est-à-dire avec des identifiants différents de ceux qui permettent l'accès aux pages décrites dans la section 3.2.1.

Toutes les routes sont protégées par JWT⁸ (un token est renvoyé en même temps que la requête).

3.3 Diagrammes des cas d'utilisation

Afin de définir les différentes actions possibles pour un utilisateur, nous avons conçu un diagramme de cas d'utilisation pour la page "Results" (cf. Figure 2). Nous avons ressenti le besoin de définir un tel diagramme pour cette page car celle-ci exige une importante implication de la part de l'utilisateur. En effet, ce dernier doit avoir la possibilité de :

- effectuer une recherche : par des noms de job (expérience) et mots-clés (tags).
- supprimer une ou plusieurs expérience(s) après avoir coché au préalable celles qu'il souhaite supprimer.
- importer un fichier local après en avoir sélectionné un.
- exporter un fichier après avoir coché au préalable celles qu'il souhaite exporter.
- actualiser la page afin d'obtenir un rendu rapidement (suite à une modification par exemple).
- trier par nom et date de l'expérience.

8. <https://jwt.io/>

En s'appuyant sur le mockup (cf. Section 9) qui porte sur la même page, il était important de vérifier avec l'utilisatrice finale, le Dr. Didia Coelho Graça[1], que ce diagramme répondait aux besoins. Ce fut parfaitement le cas et par la suite, nous n'avons pas eu besoin d'apporter des modifications au diagramme.

Notons que dans le cadre du projet NTIC, nous n'avons pas eu le temps de développer cette page.

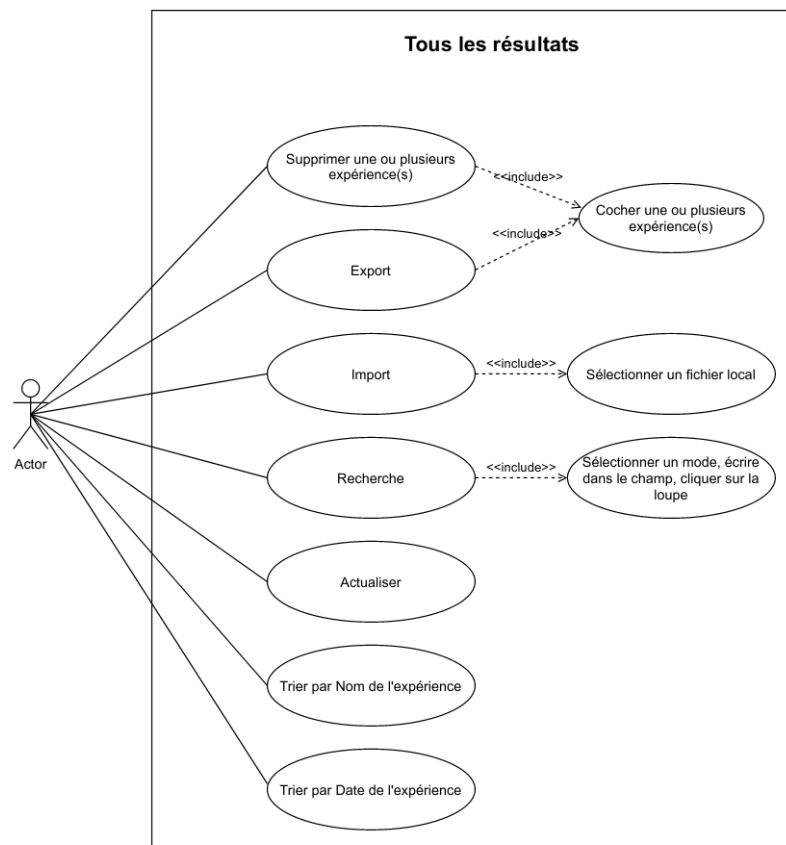


FIGURE 2 – Diagramme de cas d'utilisation de la page "Results".

4 Outils NTIC - Frontend

La partie frontend correspond à l'ensemble des interfaces interactives proposées par l'application. Dans le cadre de ce projet, React est utilisé pour créer de telles interfaces. Il s'agit d'un framework JavaScript qui produit des pages HTML. Le fonctionnement de React se base sur la création et l'association de composants réutilisables. Ainsi, chaque composant React peut être créé et utilisé indépendamment des autres composants.

Pour créer un composant, nous utilisons la syntaxe appelée JSX. Cette syntaxe permet d'injecter des balises dans du JavaScript. Ainsi, un composant pourra contenir des balises, pour le rendu de l'UI, et une logique (p. ex. les instructions conditionnelles) afin de définir comment le rendu doit se faire selon des conditions spécifiques.

Les données passées à chaque composant correspondent à des objets JSON. Il s'agit d'ensembles de couples nom/valeur, ordonnés ou non ordonnés, construits depuis le backend. Dans le cas où une donnée change, React s'assure de propager cette nouvelle modification à tous les composants enfants. En cas d'action de l'utilisateur seules les parties impactées sont mises à jours.

En ce qui concerne la mise en forme des pages, nous utilisons différents composants de la librairie Material-UI. Par exemple, pour la section "MS3" dans la page "Results", nous utilisons le composant Avatar pour l'affichage des ions et les éléments d'une chaîne. Comme avec le CSS pour les balises HTML, il est possible de personnaliser le style de chaque composant de la librairie Material-UI avec un langage très similaire au CSS (p. ex. pour modifier la taille d'un texte, nous modifions l'attribut `fontSize` alors qu'en CSS, nous modifions l'attribut `font-size`).

5 Description de la nouvelle interface utilisateur

Dans cette section, nous présentons en détail la description du travail que nous avons réalisé dans le cadre du projet NTIC, plus précisément, le développement des mockups qui portent sur l'UI, en discussion avec les utilisateurs.

5.1 Page admin

La page "Admin", comme discuté dans la section 3.2.1, permet le paramétrage des expériences.

L'administrateur peut modifier deux types de paramètres : analyse et variants. Les expériences futures seront lancées avec ces valeurs.

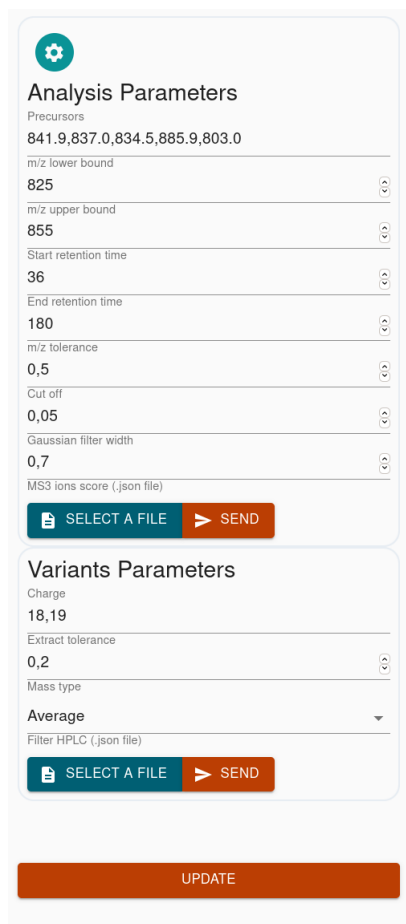
Dans la partie *Analysis Parameters* (cf. partie haut de la Figure 3), on y trouve les différents champs qui permettent la modification des paramètres d'analyse. Le champ *Precursors* prend plusieurs valeurs qui doivent chacune être séparées par une virgule. Quant aux autres champs, ils ne contiennent qu'une seule valeur.

Une limite est fixée pour chaque valeur d'un champ. Par exemple, il est impossible d'insérer une valeur supérieure à 830 pour le champ *m/z lower bound*.

En ce qui concerne le champ *MS3 ions score*, l'utilisateur a la possibilité de sélectionner un fichier au format *.json* à l'aide du bouton *SELECT A FILE* et le transmettre à l'aide du bouton *SEND*.

La partie *Variants Parameters* (cf. partie bas de la Figure 3) est similaire à la partie *Analysis Parameters*. A l'exception du champ *Mass type* qui correspond à une liste déroulante, à travers laquelle l'utilisateur peut choisir entre deux valeurs distinctes.

Une fois que le paramétrage de l'analyse et des variants terminé, l'utilisateur peut mettre à jour les modifications à l'aide du bouton *UPDATE*.



The image shows a web interface for configuring analysis parameters. It consists of two main sections: 'Analysis Parameters' and 'Variants Parameters', each with a 'SELECT A FILE' button and a 'SEND' button. Below these is a large 'UPDATE' button.

Analysis Parameters

- Precursors: 841.9,837.0,834.5,885.9,803.0
- m/z lower bound: 825
- m/z upper bound: 855
- Start retention time: 36
- End retention time: 180
- m/z tolerance: 0,5
- Cut off: 0,05
- Gaussian filter width: 0,7
- MS3 ions score (.json file)

Variants Parameters

- Charge: 18,19
- Extract tolerance: 0,2
- Mass type: Average
- Filter HPLC (.json file)

Buttons: SELECT A FILE, SEND, UPDATE

FIGURE 3 – Page "Admin"

5.2 Section MS3

L'utilisateur arrive sur cette partie pour voir les détails d'une expérience, spécifiquement pour la section dite *MS3*.

Cette affichage a fait l'objet d'une grande réflexion, nous avons gardé la représentation habituelle des utilisateurs pour une facilitation d'usage. De plus, la structure permet d'avoir un grand nombre de données sous différentes dimensions, comme via l'ajout de couleurs.

Nous pouvons la décomposer en sous-parties, pour chaque chaîne nous retrouvons : un titre ; les détail de la chaîne (avec un collapse) ; son graphique (avec un collapse). Pour une chaîne donnée :

- Nous affichons un titre composé d'un score en pourcentage, et si nous l'avons trouvé (*found*) ou lorsque nous ne l'avons pas, ce qui influence la couleur du texte de vert à rouge.
- Nous pouvons observer le détail de la chaîne avec les différents ions - via les charges - par lettre, avec toutes les informations nécessaires. Nous y retrouvons les positions, avec une échelle avec un pas de cinq, comme repère visuel. La couleur nous dit si nous l'avons trouvé, principalement du côté vert ou rouge, puis une interrogation quand la valeur n'est pas correcte. La donnée *major* est définie avec une bordure plus épaisse.
- Dans la chaîne, en passant la souris sur une charge, nous pouvons apercevoir la positions (*c* ou *z*) et la valeur (*m/z*) de la charge en question.
- Nous pouvons y afficher son plot, situé sous la chaîne et interagir avec, comme ajouter un zoom.

Tous les aspects sont affichés ci-dessous par captures d'écrans :

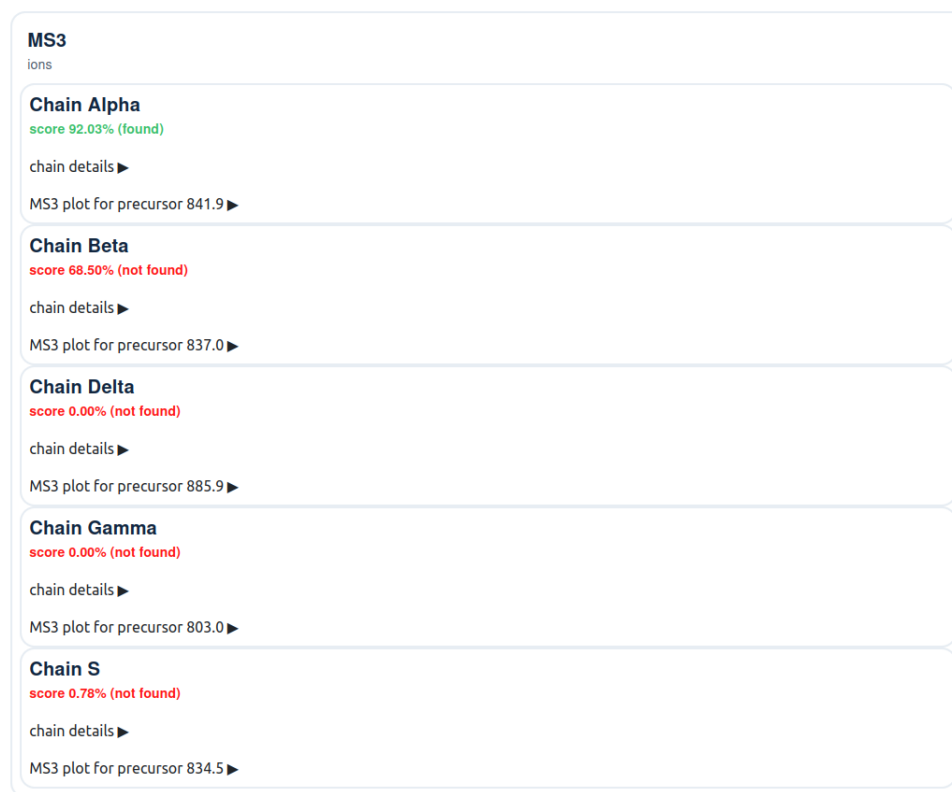


FIGURE 4 – Section MS3

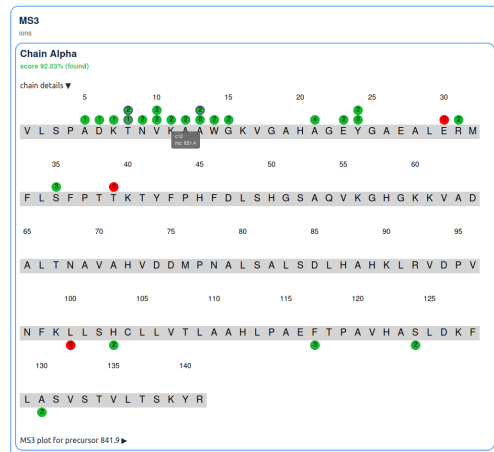


FIGURE 5 – Section MS3 pour la chaîne alpha détaillée

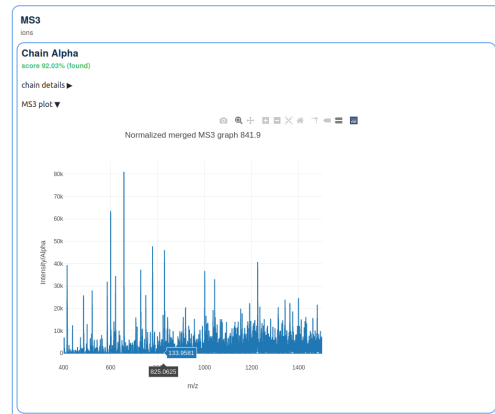


FIGURE 6 – Section MS3 pour la chaîne alpha plot

5.3 Section des variants

Dans le premier prototype, seuls les noms des mutations étaient affichés et une seconde page permettait de générer une nouvelle liste filtrée. Les variants sont maintenant affichés dans une table dont les colonnes peuvent être masquées et les données exportées en .csv.

Name	Protein	Mass Arg	Mass Mono	Mutation(s)
Hb Codon 14 (TGG>T)	alpha2	15,053,302	15,043,894	14A(A2)Tep>Lend
Hb Basel	alpha1	15,053,302	15,043,894	14A(A2)Tep>Lend
Hb Erenoum	alpha1	15,050,258	15,040,858	42(C7)Tyr>Ser
Hb Deer Lodge	beta	15,888,261	15,878,296	20(A2)His>Arg
Hb Karlsruhe	beta	15,889,268	15,879,285	21(B3)Asp>His
Hb Maryland	beta	15,889,268	15,879,285	47(CD6)Asp>His
Hb Summer Hill	beta	15,889,268	15,879,285	52(CD6)Asp>His
Hb Zürich	beta	15,888,261	15,878,296	63(E7)His>Arg
Hb Tigray	beta	15,889,268	15,879,285	79(F2)Asp>His
Hb Michalek	beta	15,888,261	15,878,296	82(F6)His>Arg
Hb Barcelona	beta	15,889,268	15,879,285	84(F4)Asp>His
Hb Yakima	beta	15,889,268	15,879,285	89(G1)Asp>His
Hb Sfax	beta	15,888,261	15,878,296	118(G18)His>Arg
Hb P-Galveston	beta	15,888,261	15,878,296	117(G18)His>Arg
Hb Abruzzo	beta	15,888,261	15,878,296	142(H2)His>Arg
Hb Cochin-Port Royal	beta	15,888,261	15,878,296	146(H2)His>Arg
Hb Costa Rica	beta	15,888,261	15,878,296	221(EF1)His>Arg

FIGURE 7 – filtre HPLC activé

Name	Protein	Mass Arg	Mass Mono	Mutation(s)
Hb Codon 14 (TGG>T)	alpha2	15,053,302	15,043,894	14A(A2)Tep>Lend
Hb Basel	alpha1	15,053,302	15,043,894	14A(A2)Tep>Lend
Hb Erenoum	alpha1	15,050,258	15,040,858	42(C7)Tyr>Ser
Hb Deer Lodge	beta	15,888,261	15,878,296	20(A2)His>Arg
Hb Karlsruhe	beta	15,889,268	15,879,285	21(B3)Asp>His
Hb Maryland	beta	15,889,268	15,879,285	47(CD6)Asp>His
Hb Summer Hill	beta	15,889,268	15,879,285	52(CD6)Asp>His
Hb Zürich	beta	15,888,261	15,878,296	63(E7)His>Arg
Hb Tigray	beta	15,889,268	15,879,285	79(F2)Asp>His
Hb Michalek	beta	15,888,261	15,878,296	82(F6)His>Arg
Hb Barcelona	beta	15,889,268	15,879,285	84(F4)Asp>His
Hb Yakima	beta	15,889,268	15,879,285	89(G1)Asp>His
Hb Sfax	beta	15,888,261	15,878,296	118(G18)His>Arg
Hb P-Galveston	beta	15,888,261	15,878,296	117(G18)His>Arg
Hb Abruzzo	beta	15,888,261	15,878,296	142(H2)His>Arg
Hb Cochin-Port Royal	beta	15,888,261	15,878,296	146(H2)His>Arg
Hb Costa Rica	beta	15,888,261	15,878,296	221(EF1)His>Arg

FIGURE 8 – charges 19 masquées

Comme cela a été demandé, deux champs (min, max) permettent de définir une fenêtre temporelle. Si le filtre est activé, les variants sont grisés et barrés étant donné qu'ils ne

remplissent pas cette condition. L'état du filtre est sauvegardé dans l'expérience⁹ et c'est cet état qui est affiché au chargement de la page. Le filtre peut être activé/désactivé à volonté et ses valeurs changées. Chaque changement provoque un rafraîchissement de la table.

Nous avons proposé d'utiliser deux couleurs selon la charge des variants (charge 18 ou 19) et d'ajouter deux boutons pour les afficher/masquer. Cette options remplissant son rôle, la colonne *charge* a donc été retirée de la table.

9. Sur le serveur via une requête *put*.

6 Validation de l'UI par les utilisateurs

Pendant le développement du projet, il fut nécessaire de valider notre avancement avec le Dr. Didia Coelho Graça[1] afin de s'assurer que nos travaux répondent réellement aux besoins des utilisateurs finaux.

Après avoir conçu les mockups des différentes pages, nous avons rencontré le Dr. Didia Coelho Graça[1] lors d'une réunion afin de valider avec elle l'ergonomie des IHM associée aux futures pages que nous allions développer. Nous avons eu de très bons retours de sa part et cela nous a permis de commencer le développement de l'UI avec React.

Cette section comporte trois sous-sections correspondant chacune aux pages développées dans le cadre de ce projet : Admin, MS3 et Variants. Ces sous-sections établissent une comparaison du travail annoncé par rapport au travail rendu.

6.1 Validation de la page Admin

Le Dr. Didia Coelho Graça[1] est satisfaite de la page. Cela étant, on utilise le format de fichier .json pour uploader des fichiers alors que le Dr. Didia Coelho Graça[1] a l'habitude d'utiliser des fichiers Excel. Cette page rencontre donc une limite concernant le format de fichier mais qui n'est pour l'instant, pas prioritaire.

6.2 Validation de la page MS3

Après le développement de cette partie, nous l'avons rendu comme aperçu à l'utilisateur pour valider si la pratique correspondait à la théorie (nos mockups).

C'est important de noter que des détails ne se retrouvent pas forcément dans les exemples validés la première fois, et qu'il est primordiale de passer par cette étape pour modifier selon les besoins réels finaux, nous le retrouvons notamment au travers de l'exemple suivant.

Donc, le point qui a été retenu, c'est lorsque nous affichons les charges, une valeur de zéro n'existe pas, nous avons pris en compte ce cas particulier pour le traiter et modifier son affichage par un point d'interrogation.

En parallèle, pour assurer un code correct, nous avons également effectué nos propres tests pour réduire le facteur d'erreurs.

Nous avons commencé par rentrer des valeurs fictives et avons mis en place la structure de base, et des changements de taille d'écran pour le côté "responsive".

Lorsque tout a été mis en place, comme test final, nous prenons les données réelles du fichier et les comparons manuellement avec celles affichées. Autrement dit, nous vérifions également que les ions MS3 soient tous présents, d'après une expérience existante.

6.3 Validation de la page Variants

Comme cela a été discuté dans la Section 5.3, l'affichage des variants sous forme de table offre beaucoup plus d'informations qu'auparavant. Le Dr. Didia Coelho Graça[1] a été très satisfaite de la nouvelle version.

7 Difficultés rencontrées

Dans le cadre de ce projet, nous n'avons pas rencontré beaucoup de difficultés. Cela étant, étant donné que ne nous n'avons jamais eu d'expérience avec React et qu'il s'agisse d'un outil complexe à prendre en main, nous avons mis du temps à démarrer le développement. De plus, nous avons constaté qu'avec React, certaines fonctionnalités que l'on peut qualifier de difficiles à mettre en place, étaient au final très faciles et rapides tandis que pour certaines, qui correspondent à de légers détails, pouvaient prendre des heures.

Dans cette section, nous présentons les éléments pour lesquels nous avons rencontré certains obstacles durant la phase de développement.

7.1 Débogage

Le débogage sous React est difficile. En effet, pour comprendre le comportement de notre code, nous avons souvent eu besoin d'afficher la valeur de nos variables depuis notre navigateur web, comme suit :

```
console.log(maVariable)
```

Cela étant, nous n'avons aucun contrôle sur ce type de commandes qui s'exécutent en permanence, brouillent l'affichage des valeurs des variables et ralentissent l'affichage de la page. Au final, il est difficile d'aller de l'avant car nous avons du mal à se situer et à repérer les erreurs dans notre code.

7.2 Table utilisée pour les variants

Les variants sont affichés dans une table venant d'une librairie développée par material-ui¹⁰. Ce composant React est en cours de développement et la documentation en ligne décrit des fonctionnalités non encore implémentées. En faisant des recherches nous nous sommes aperçu que beaucoup d'autres développeurs rencontraient des difficultés avec ce composant. Comme il répond parfaitement aux besoins nous l'avons quand même utilisé, mais pour l'instant, il est par exemple impossible d'afficher une cellule sur plusieurs lignes ou de choisir l'alignement des colonnes. Selon la documentation, ces fonctionnalités devraient être disponibles lors des futures mises à jour.

10. <https://material-ui.com/components/data-grid/>

7.3 Mise à jour de l'UI

Cette aspect de React - qui lui est propre - est un point clé et très positif, mais également contraignant à gérer. Tout changement d'une partie est mis à jour automatiquement. Lié à cela, nous avons rencontré des problèmes dans le code en lui-même, c'est un nouveau concept qu'il faut prendre en main et penser à chaque étape. En plus, lors du débogage, un rafraîchissement constant devient vite illisible lors de chaque modification (cf. [Section 7.1](#)).

8 Conclusion

La base du projet repose sur une demande bien particulière des utilisateurs pour l'automatisation d'un processus d'analyse sur les mutations de l'hémoglobine. Il existe une première version, qui a nécessité d'être renouvelée à cause de certains défauts de conception et pour repenser son ergonomie.

Nous avons établi une liste de tâches selon les besoins prioritaires, puis avons conçu des mockups pour les représenter. Trois sections en sont ressorties : l'une orientée pour la modification du traitement ; les deux autres pour l'affichage des résultats.

Il existe une infrastructure importante sous forme de containers, nous décrivons également le plan général du site avec ses diverses pages et leurs accès.

Nous avons utilisé React comme outil principal, mais celui-ci est un framework se basant sur d'autres propriétés (HTML, CSS, etc.). Il a aussi une syntaxe qui lui est propre.

La grande partie du travail demandé dans le cadre du projet NTIC, est de concevoir des parties d'affichages, les trois sections. Après leurs développements, nous les présentons et avons décrit leurs fonctionnements.

Tout une partie est liée à la validation de l'utilisateur, c'est important de savoir durant le parcours si cela correspond toujours aux attentes entre le début et la fin du projet.

Durant ces mois, nous avons rencontré des problèmes variés que nous avons décrit, mais principalement lié au code. Mais finalement, les utilisateurs et nous sommes très satisfaits du rendu final.

Le développement du projet devrait continuer sa poursuite au sein de l'HEPIA ces prochaines semaines et être déployé en production.

Références

- [1] D. Coelho Graca, A. Scherl, K. Samii, D. Hochstrasser, and P. Lescuyer, “Diagnostic des hémoglobinopathies par spectrométrie de masse,” *Spectra Analyse*, vol. 314, p. 39, 2017. [Online]. Available : <https://archive-ouverte.unige.ch/unige:93265>
- [2] “Spectrométrie de masse,” Jun. 2020, page Version ID : 171766365. [Online]. Available : https://fr.wikipedia.org/w/index.php?title=Spectrom%C3%A9trie_de_masse&oldid=171766365
- [3] “HbVar query results.” [Online]. Available : https://globin.bx.psu.edu/cgi-bin/hbvar/query_vars3
- [4] “Tutoriel : intro à react.” [Online]. Available : <https://fr.reactjs.org/docs/hello-world.html>
- [5] “Modèle-vue-contrôleur,” Feb. 2021, page Version ID : 180227976. [Online]. Available : <https://fr.wikipedia.org/w/index.php?title=Mod%C3%A8le-vue-contr%C3%B4leur&oldid=180227976>
- [6] “Mass Spectrometry.” [Online]. Available : <https://www.bruker.com/en/products-and-solutions/mass-spectrometry.html>

9 Annexes

9.1 Listes des mockups

[Logout](#)

Admin parameters

Analysis Parameters

Precursors

m/z lower bound

m/z upper bound

Start rt

End rt

m/z tolerance

Cut off

Gaussian width

MS3 ions score

No file chosen : .json

Variants Parameters

Charge

Extract tolerance

Mass ☒ avg ☐ mono

Filter HPLC

No file chosen : .json

FIGURE 9 – Mockup Admin

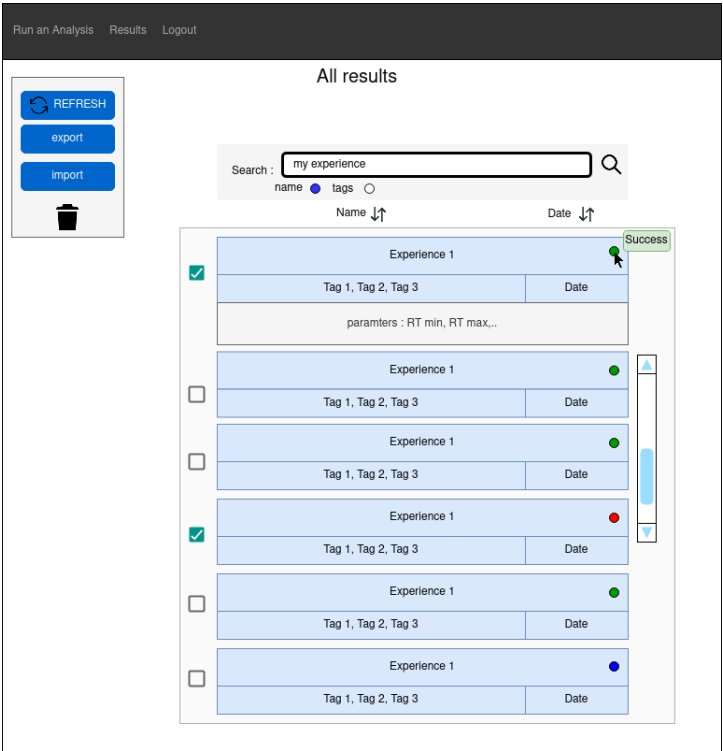


FIGURE 10 – Mockup Results



FIGURE 11 – Mockup MS3

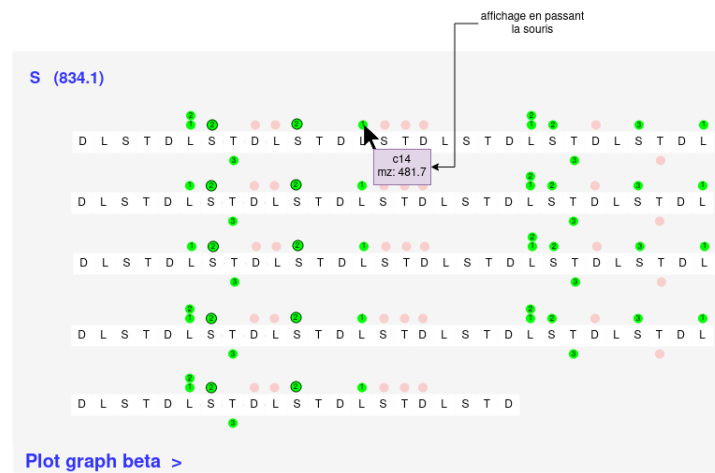


FIGURE 12 – Mockup MS3 details chain

filtre non sélectionné

Variants

HPLC filter min max

836.1 (6 variants) ch 18 ch 19

Name	charge	mass
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	

839.12 (22 variants)

841.1 (5 variants) ch 18 ch 19

Name	charge	mass
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	

FIGURE 13 – Mockup MS3 variant

filtre sélectionné

Variants

HPLC filter 0.15 2.5

836.1 (6 variants) ch 18 ch 19

Name	charge	mass
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	

839.12 (22 variants)

841.1 (5 variants) ch 18 ch 19

Name	charge	mass
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 18 m. avg : 15861.01 m. mono : 15856.2	
Hb F-Cobb II (Ggamma) 37(C3)(Trp)-(Gly)	charge 19 m. avg : 15861.01 m. mono : 15856.2	

FIGURE 14 – Mockup variant with filter