POLYTECHNIQUE MONTRÉAL

UNIVERSITÉ D'INGÉNIERIE

Département de génie informatique et de génie logiciel

INF8970 – INF8980 – INF8985

Projet final en génie informatique

Rapport final

Développement d'un environnement OpenAi/Gym pour simulation de réseaux électriques intelligents et soutenables — Projet Mila

Équipe No 1

Ghazi Ben Achour

Alex Hua

Johnny Khoury

Alison Nemr

Victor Vergeau

Mohamed Zakaria

Jeudi 20 avril 2023

Table des matières

*Note : Les chiffres placés en exposant à la suite de certains mots font référence à leur définition dans le dictionnaire placé à la fin du rapport.

1. Changement dans le contexte du projet

1.1 Description des changements imposés par le client depuis la mi-session

Après la présentation de mi-session du projet, le client était satisfait du travail que nous avons effectué et de l'état d'avancement du projet. Le client a également bien apprécié la nouvelle interface que nous avons développée et les nouvelles fonctionnalités qui n'étaient pas dans l'ancienne interface. Cependant, après la présentation de mi-session, le client nous a rappelé l'importance de se concentrer à implémenter les fonctionnalités existantes sur la nouvelle version afin d'avoir un produit final utilisable. Ces fonctionnalités sont essentielles pour que l'ancienne version du produit puisse être laissée de côté.

Nous avons donc tenu une réunion avec le client pour qu'il nous donne ses rétroactions après la présentation de mi-session. Un des changements demandés concernait l'interface utilisateur[1]. Le client nous a demandé d'apporter quelques modifications quant à l'apparence du projet. Certains commentaires par rapport aux contrastes des éléments de l'interface utilisateur ont été soulevés et résolus. De plus, à certains endroits, la taille de l'écriture était trop petite et à d'autres places, il manquait certains détails. Nous avons donc eu la chance de faire les changements nécessaires pour atteindre nos objectifs et satisfaire le client.

Du côté du serveur et de la logique[2], le client était satisfait de notre travail. Il nous a toutefois rappelé les tâches prioritaires qui devaient être effectuées. Nous les présenterons dans la section 3.1.

Bien que le client n'ait pas exigé de changements majeurs lors de la rencontre de mi-session, nous avons considéré les requis et les demandes du client comme des changements que nous devions apportés, puisqu'ils sont des tâches qui devaient être effectuées. Nous parlerons de ces tâches dans la section 4.1.

1.2 Difficultés techniques nouvellement apparues

Puisque le projet a commencé à prendre plus d'ampleur depuis la mi-session, il est normal d'avoir rencontré des difficultés lors du développement du projet. Nous avons rencontré certaines difficultés tant au niveau de l'interface utilisateur que le serveur et la logique.

Tout d'abord, du côté interface utilisateur, nous avons rencontré des difficultés au niveau de la disposition de certains éléments. Par exemple, nous avions prévu, au départ, une zone sur la page pour y placer les graphiques. Nous nous sommes rendu compte qu'il fallait mieux présenter cette zone; nous ne pouvions pas seulement y placer des graphiques, car cela ne serait pas beau visuellement. Certaines couleurs et tailles d'écriture ont été choisies pour avoir une belle présentation visuellement.

Pour ce qui est des difficultés techniques, nous avons utilisé certaines technologies que nous ne connaissions pas avant. Nous avons également utilisé des technologies que nous connaissions déjà, mais que nous devions utiliser dans un nouveau contexte. Par exemple, pour afficher les graphiques sur l'interface utilisateur, nous avons utilisé la librairie « Chart.js ». Cette librairie permet d'afficher des données dans un graphique. Nous avons rencontré certaines difficultés avec cette fonctionnalité. En effet, un des défauts de cette librairie est qu'il n'y a pas beaucoup de documentation qui explique comment en faire usage. Toutefois, nous avons quand même choisi de l'utiliser, car elle est très couramment utilisée pour la création de graphes dans les fureteurs. Pour remédier à la situation qu'il y ait peu de documentation, nous avons expliqué comment ajouter d'autres graphes dans la documentation que nous allons offrir au client à la fin du projet. Nous avons toutefois passé beaucoup de temps à comprendre comment en faire usage. Les difficultés que nous avons rencontrées sont arrivées lorsque nous voulions afficher des données en temps réel sur le graphe. En effet, nous devions trouver une façon de faire afficher des données en temps réel et que les graphiques se mettent à jour automatiquement. De plus, l'utilisateur devait être en mesure de se déplacer à l'intérieur du graphe. Toutes ces contraintes ont été pour nous une des difficultés techniques que nous avons rencontrées. Par ailleurs, « Chart.js » est une librairie faite pour le langage JavaScript. Bien que nous utilisions « TypeScript » comme langage, « Chart.js » est fonctionnel avec ce langage, mais nous avons dû faire quelques modifications afin de le rendre compatible. Cela a pris un peu de temps, mais, finalement, nous avons un beau produit qui est facilement utilisable par le client maintenant que nous l'avons intégré.

Par la suite, nous avons également rencontré des difficultés au niveau de certaines technologies que le client voulait qu'on utilise. Par exemple, nous devions prendre en compte la simulation Monte-Carlo. Nous ne connaissions pas vraiment cette méthode d'interpolation auparavant. Nous avons donc décidé de tenir une rencontre avec le client pour avoir quelques explications sur cela. Après la réunion, le lien entre cette méthode et le reste de notre application était beaucoup plus clair. La simulation Monte-Carlo permet en fait de prédire les résultats potentiels que pourrait avoir notre système. En comprenant cela et quelques autres concepts, nous avons pu l'intégrer à notre application.

2. Description de l'architecture finale du système (Q5.2 – Q5.3 – Q5.4)

2.1 Résumé de ce qu'était le système à la mi-session et ce qu'il est maintenant (Q4.5 et Q4.6)

Ce qui a été fait à la mi-session :

- Nous avons implémenté une nouvelle architecture pour l'interface utilisateur;
- Nous avons développé une architecture de code robuste et résiliente à l'ajout de nouvelles dynamiques et fonctionnalités dans le côté interface client;
- Nous avons commencé la documentation interne (commentaire du code dans les fichiers) et externe (création d'un « wiki doc » contenant toutes les fonctions du code avec les commentaires) du projet;
- Nous avons commencé la refonte du code serveur et logique;
- Nous avons commencé la mise en place d'une structure pour le fichier de configuration;
- Nous avons été en mesure d'envoyer les vraies données du serveur à l'interface utilisateur en temps réel pour afficher les données globales de la simulation sur la grille d'agents pour les maisons et les détails de chaque maison dans les modales.

Ce qui a été fait comme pour la remise finale :

- Nous avons implémenté la fonctionnalité filtrage et triage qui fonctionne bien;
- Nous avons ajouté les graphiques des maisons individuelles et de la grille en général;
- Nous avons terminé la refonte du code;
- Nous avons fait l'intégration de « Wandb »;
- Nous avons été en mesure de faire un programme qui fonctionne avec plus d'un agent;

- Nous avons implémenté une fonctionnalité qui permet de naviguer sur différents pas de temps;
- Nous avons terminé le développement de l'architecture développée;
- Nous avons fait afficher les détails des maisons individuelles dans les modales;
- Nous avons réalisé des tests unitaires et d'intégration.

Les tâches qui ont été réalisées dans la seconde moitié de la session étaient planifiées dans notre échéancier. En effet, nous n'avons pas connu de surprises ou de problème majeur quant à l'échéancier.

La version du système développé à la mi-session était plutôt un prototype qui montrait les grandes lignes du produit final. Bien qu'elle ait servi à donner une bonne idée de ce que nous allions fournir au client à la fin du projet, plusieurs éléments n'étaient pas terminés. Bien que nous fussions dans les temps pour la mi-session, il nous restait beaucoup de travail à faire pour atteindre nos objectifs et ceux du client.

En résumé, le produit à la mi-session n'était bien évidemment pas complet. Entre la présentation de mi-session et la remise finale, nous avons continué à développer le produit. Nous avons terminé les fichiers de configuration et la refonte du code. De plus, les dernières fonctionnalités ont été implémentées en plus d'être testées. Les fonctionnalités incomplètes lors de la mi-session ont également pu être terminées. Par ailleurs, les architectures développées ont pu être testées et validées.

2.2 Diagrammes de classes ou de modules

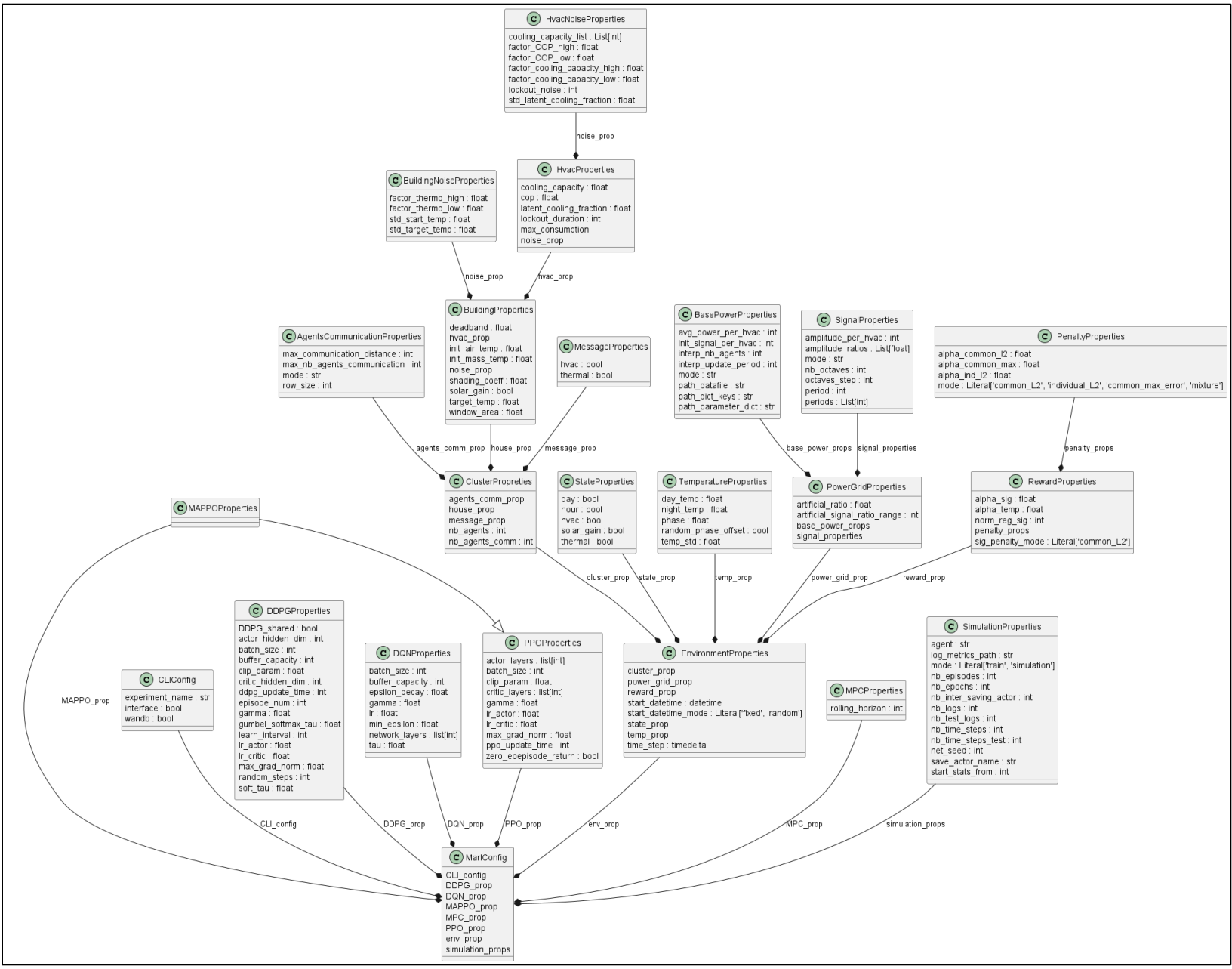

Figure 1: Diagramme de classe de la configuration

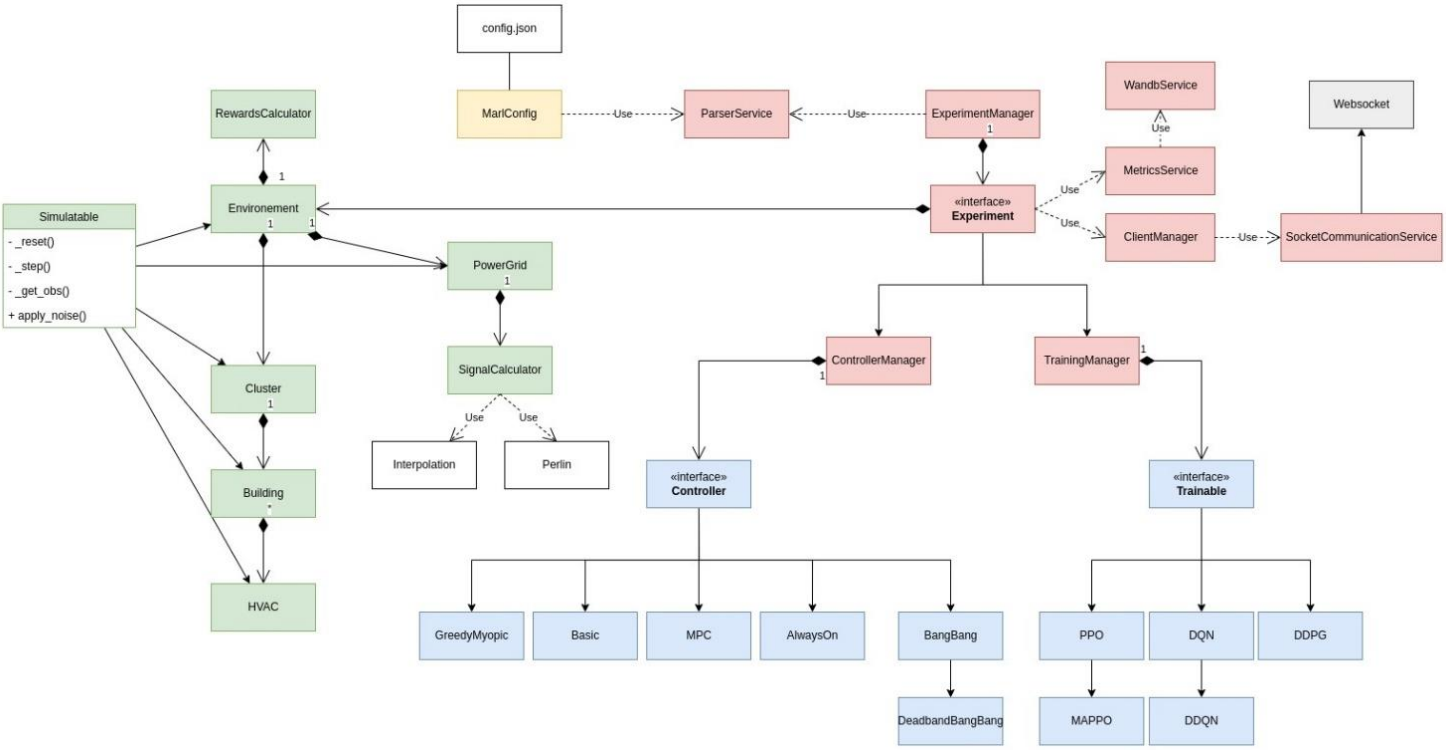

Figure 2: Diagramme de classe du côté serveur et logique

Pour ce qui est du diagramme de classe pour la configuration, il y a eu quelques changements depuis la mi-session. La figure 1 montre la nouvelle version du dictionnaire de configuration. C'est un dictionnaire qui contient toute la configuration du système. Le plus grand changement qu'il y a eu est le fait qu'avant, lorsqu'on voulait modifier la configuration, il fallait le faire par le dictionnaire de configuration, notamment en changeant les paramètres de ligne de commande. Dans le but d'avoir une architecture plus solide et mieux définie, nous avons fait en sorte que pour modifier les paramètres de ligne de commande, on doit effectuer la modification dans le fichier JSON. Ainsi, nous avons maintenant une configuration différente que nous avons appelée « marl config ». Il s'agit en fait d'un objet de type Pydantic, soit une librairie qui est utilisée avec notre projet (voir la section 2.5 pour plus de détails).

Pour ce qui est de la logique en lien avec le fonctionnement du serveur, la figure montre la logique que nous avons implémentée. Nous avons toujours, comme c'était le cas lors de la mi-session, deux composantes majeures au système. La première composante est celle de l'environnement et la seconde est celle de l'agent. L'environnement et l'agent ne communiquent jamais de façon directe; il faut passer par un « TrainingService ». L'agent

s'occupe de fournir des tâches à faire à l'environnement en fonction des observations que ce dernier lui donne par un dictionnaire d'observation. Cela se fait à chaque saut de temps, de façon à respecter la norme Gym qui était implémentée avec la version de départ du client.

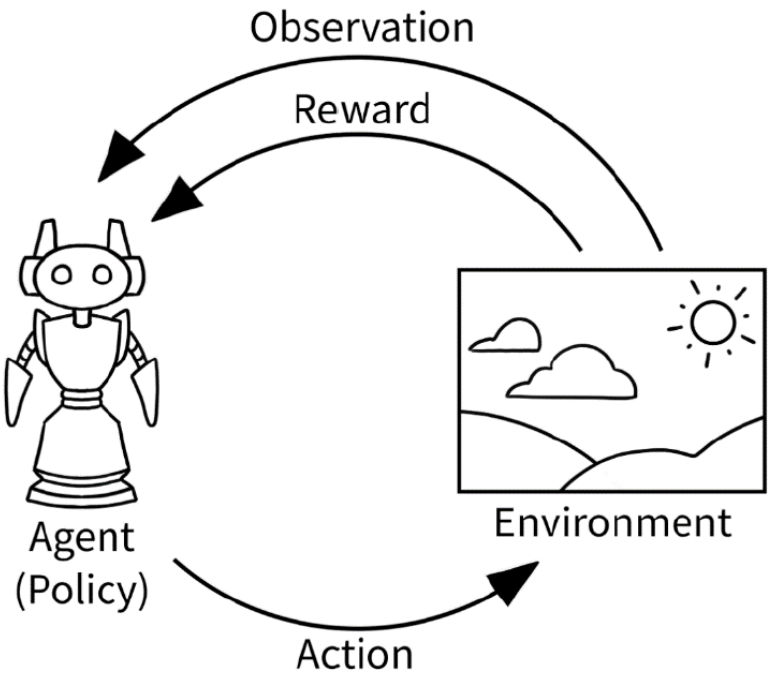

Figure 3: Norme Gym pour la communication entre l'environnement et l'agent [2]

Dans la figure 3 que nous avons également présentée dans le rapport de mi-session, cette figure montre le fonctionnement dont l'agent que nous implémentons et l'environnement communiquent ensemble. Nous avons jugé pertinent de montrer cette figure dans le rapport final, car cette norme a été utilisée tout au long du projet. De façon générale, l'agent envoie des actions à l'environnement. Ce dernier lui retourne des observations et des récompenses en fonction des actions que l'agent lui a dit de faire. Ensuite, ce cycle recommence tout au long de la simulation. C'est de cette façon que les agents que nous avons implémentés et notre environnement communiquent. Ce sera également le cas avec n'importe quel agent qui sera implémenté dans le système, puisqu'il s'agit de la norme que nous avons suivie dans notre application au niveau de la communication entre le serveur et logique entre l'environnement et l'agent.

2.3 Diagramme d'états et/ou d'interface usager

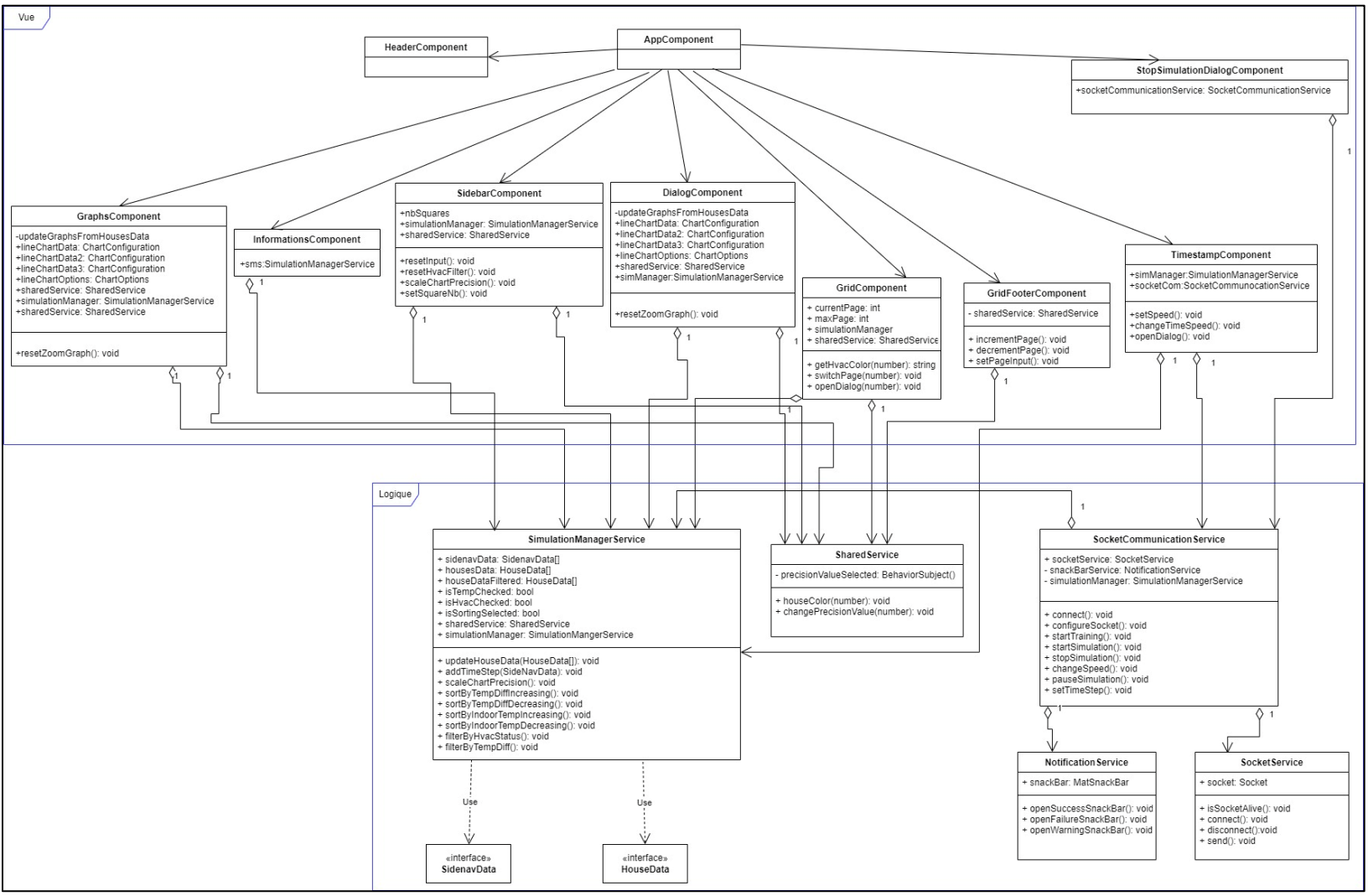

Figure 4: Diagramme de classe client interface usager

La figure 4 présente le diagramme de classe de notre interface usager. Nous pouvons voir qu'il y a 10 composantes pour l'interface utilisateur, soient « HeaderComponent », « GraphsComponent », « InformationsComponent », « SidebarComponent », « DialogComponent », « GridComponent », « GridFooterComponent », « TimestampComponent », « StopSimulationDialogComponent » et « AppComponent ». Nous avons continué de nommer en anglais les composantes afin d'assurer l'uniformité dans le code, puisque c'est cette langue utilisée par le client pour développer le projet. Ce choix a été fait après en avoir discuté avec le client en début de session. Puisque la

version de base que nous avons reçue du client était faite en anglais, nous avons continué avec la même logique. De plus, les utilisateurs finaux seront des chercheurs qui ont tous des bagages différents. En faisant le code en anglais, nous permettons à plus de chercheurs d'être en mesure d'utiliser l'application. Ces 10 composantes se regroupent toutes sous la composante « AppComponent ». Chacune d'entre elles s'occupent d'effectuer une tâche bien précise que nous décrirons.

Pour ce qui est du côté logique de l'interface, plusieurs services et composantes sont utilisés pour assurer le bon fonctionnement. Tout d'abord, le service « SimulationManagerService » est utilisé pour récupérer les informations du serveur et les distribuer aux différentes composantes en fonction de leur rôle. Ensuite, le service « SharedService » est utilisé pour faciliter l'échange d'éléments entre les différents composants, tandis que le service « SocketCommunicationService » est utilisé pour gérer la communication entre le serveur et le client.

L'interface « HouseData » est utilisée pour organiser les données des maisons et les afficher dans le composant « GridComponent ». Les différents services et composantes de l'interface utilisateur sont étroitement liés, avec certaines composantes qui implémentent des méthodes qui sont utilisées par d'autres.

Il est donc important de comprendre la nature des relations entre les différentes classes pour écrire du code optimisé et faciliter la maintenance du système.

Le « HeaderComponent » s'occupe de l'en-tête de la page. Comme nouveauté, nous avons rajouté la dynamisation du nombre d'agents qui peut être affiché par page. Cette fonctionnalité se retrouve dans la composante « SidebarComponent », puisque c'est là où la liste déroulante a été ajoutée, allant de 25 à 256 maisons pouvant être affichées sur la même page. La composante « GridComponent » permet de faire l'affichage des agents sur l'interface séparée par une grille. La composante « GridFooterComponent » permet la gestion de l'incrémentation et de la décrémentation du numéro de page pour l'affichage des agents.

Il y a aussi l'ajout des graphiques, qui étaient déjà présents dans la version originale de l'interface du client. Afin d'implémenter cette fonctionnalité, deux composantes ont été utilisées : « DialogComponent » et « GraphsComponent ». Les deux composantes contiennent, en grande partie, la même logique pour afficher les graphes. Nous faisons afficher les graphes dans deux composantes, car nous en affichons sur deux pages

différentes. Pour les graphes généraux qui contiennent les valeurs de toutes les maisons, elles se font afficher sur la page d'accueil et cela est placé dans le « GraphsComponent ». Pour ce qui est des graphes pour les maisons individuelles, la logique est placée dans le « DialogComponent ».

De plus, pour implémenter la fonctionnalité de la gestion de la simulation pour ce qui est de la vitesse de la simulation et de son état (arrêt de la simulation complète, pause de la simulation et la glissière pour naviguer dans le temps pour voir l'évolution de l'état des agents), la composante « TimestampComponent » et le service « SocketCommunicationService » ont été utilisés afin de communiquer avec le serveur, puisque c'est ce dernier qui s'occupe de l'envoi des données. « StopSimulationDialogComponent » est une composante utilisée pour afficher une fenêtre qui demande à l'utilisateur s'il est sûr de vouloir arrêter la simulation lorsqu'il pèse sur le bouton d'arrêt de la simulation.

Pour ce qui est des filtres, la majorité de l'implémentation se trouve dans le service « SimulationManagerService » et la composante « SidebarComponent » est utilisée pour afficher les trois différents filtres : trier par ordre de différence de température et de température intérieure, filtrer par statut de CVC et filtrer par intervalle de différence de température.

Par la suite, la composante « InformationComponent » permet l'affichage des informations qui peuvent être utiles aux chercheurs qui étaient déjà présents dans la version originale de l'interface du client.

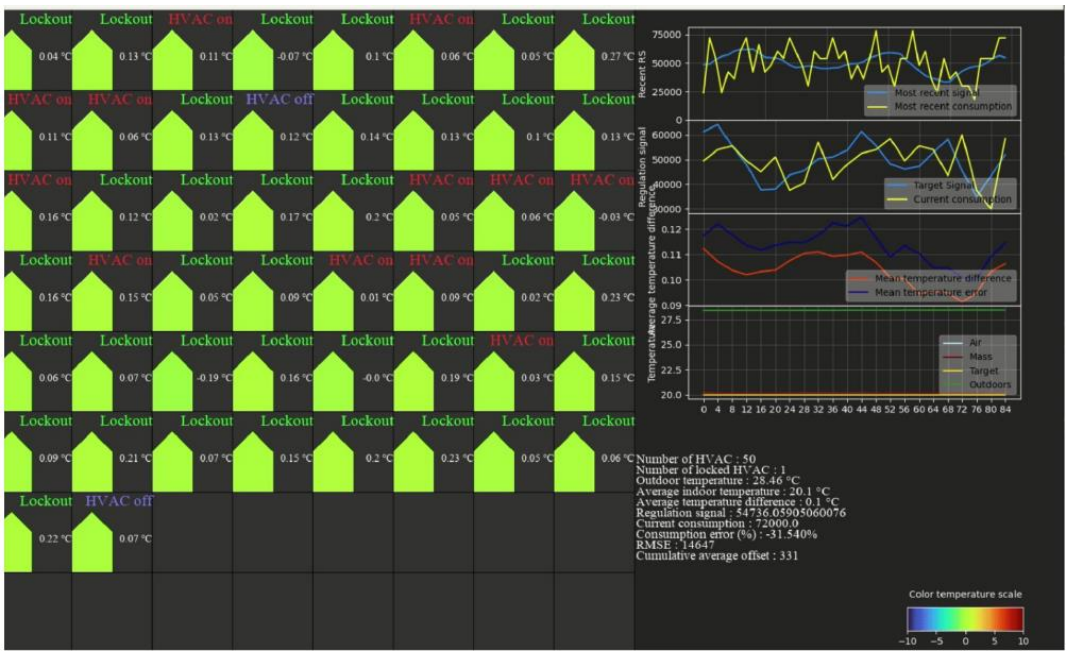

Figure 5: Ancienne interface usager (la version de base du client)

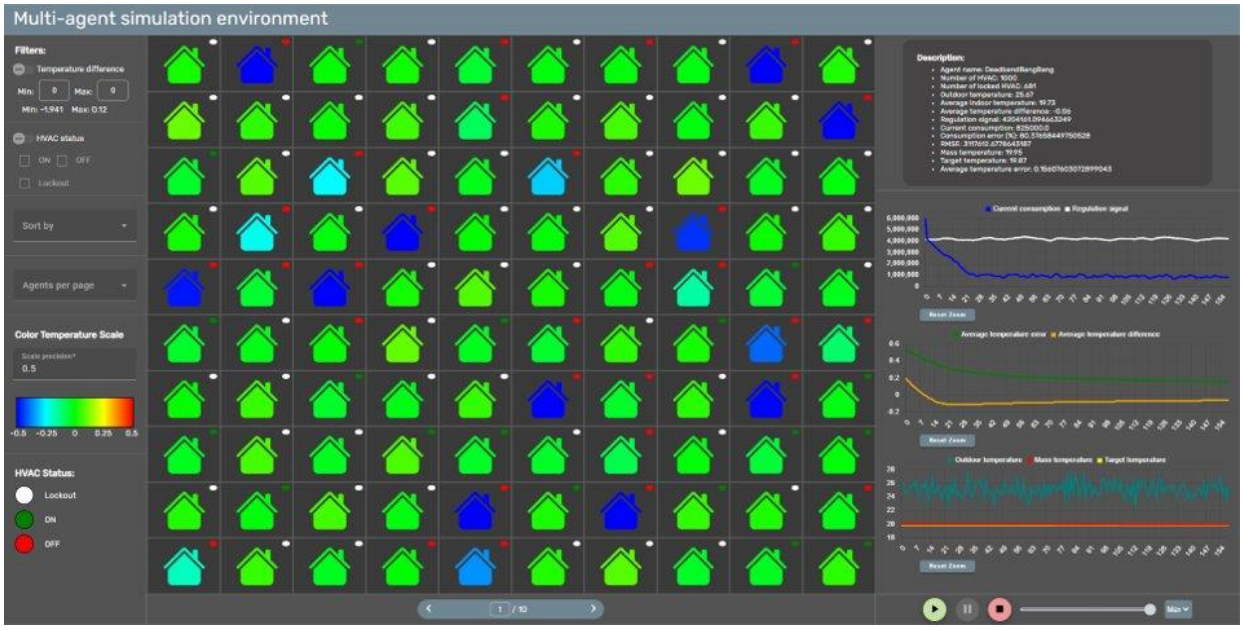

Figure 6: Nouvelle interface usager (celle que nous avons développée)

Les figures 5 et 6 présentent les interfaces utilisateurs de l'application. La figure 5 montre l'interface utilisateur de base, soit celle que le client avait dans sa version de base et la figure 6 montre la nouvelle interface que nous avons développée tout au long de la session. Nous pouvons voir qu'elle est beaucoup plus moderne, élégante et facile d'utilisation. Plusieurs nouvelles fonctionnalités sont également disponibles.

2.4 Interfaces logicielles et/ou matérielles importantes

Tout comme c'était le cas lors de la remise de mi-session, notre projet n'a pas nécessité d'interfaces matérielles pour la remise de la fin de session. En effet, aucun capteur ou autre objet n'est nécessaire au bon fonctionnement du système que nous avons développé. Les sections 2.2 et 2.3 qui présentent nos diagrammes de classes sont suffisantes pour expliquer le fonctionnement de l'application.

2.5 Description d'outils, librairies ou cadre de travail (*framework*) utilisés

Du côté des outils et des technologies que nous avons utilisées, nous avons sensiblement gardé les mêmes que ceux que nous avions choisis lors de la mi-session. Outre le matériel nécessaire pour le travail et la communication qui inclut, entre autres, des ordinateurs, des microphones et une connexion à internet, nous avons également fait usage des différentes connaissances acquises durant tout le parcours à Polytechnique Montréal.

Pour ce qui est de l'environnement technologique, tous les membres de l'équipe ont utilisé l'éditeur de code « Visual Studio Code » comme c'était le cas pour le développement de la mi-session. Il n'est pas nécessaire que tous les membres de l'équipe utilisent le même environnement technologique, mais cela rend plus facile l'entraide lorsqu'il y a un problème. De plus, nous sommes tous à l'aise à utiliser cet environnement technologique.

Par ailleurs, nous avons utilisé différents cadriciels[3] lors du développement. Comme c'était le cas pour la mi-session, nous avons continué à utiliser Angular Material pour l'interface utilisateur. Ce cadre de travail nous a permis de développer une interface utilisateur moderne et d'y intégrer des composantes dont le code est fourni et testé.

Par ailleurs, toujours pour l'interface utilisateur, nous avons utilisé la librairie « Chart.js » pour être en mesure de tracer les graphiques. « Chart.js » permet de facilement intégrer des graphiques et de les optimiser de la façon voulue. Nous avons connu quelques difficultés avec cette librairie au départ pour différentes raisons telles qu'expliquées dans la section 1.2 du présent rapport.

Pour ce qui est des langages que nous avons utilisés du côté de l'interface utilisateur, nous avons utilisé le « TypeScript », « HTML » et « CSS » comme convenu avec le client en début de session. Ces choix de langages n'ont pas changé depuis et ils nous ont permis d'accomplir toutes les tâches qui étaient requises.

Du côté du serveur et de la logique, les choix de librairies et de langages n'ont également pas changé. Pour ce qui est du langage, nous avons continué d'utiliser le « Python ». Ce langage a été utilisé par le client dans la version qu'il nous a fournie, car les chercheurs connaissent déjà, en majorité, le « Python ». Nous avons donc continué à utiliser ce langage pour la suite du projet, même après avoir fait la refonte du code.

Pour ce qui est des librairies utilisées, tout comme c'était le cas pour la remise de mi-session, nous avons continué à utiliser la bibliothèque « Socket.io ». Cette dernière permet d'assurer la communication bidirectionnelle entre le client et le serveur. Dans notre cas, elle nous était utile, car notre système demande de recevoir des données sur les maisons de la part du serveur afin de les afficher sur l'interface. En assurant une bonne communication entre le client et le serveur, nous pouvons afficher les données, afficher les détails des maisons et aussi afficher des graphiques. Pour fonctionner, « Socket.io » se base sur des protocoles « Websocket ». Cette bibliothèque s'intègre parfaitement à notre projet, car elle est utilisable avec le langage « Python », soit le langage que nous utilisons pour le serveur et la logique.

Ensuite, toujours du côté du serveur et de la logique, nous avons continué à utiliser la librairie « FastApi ». Elle nous sert lorsque nous utilisons des API en Python. Elle est basée sur la librairie « Pydantic », soit une librairie qui était utilisée par le client dans la version de base qu'il nous a donnée. Nous utilisons toujours la librairie « Pydantic » pour faire la validation de données sur le fichier de configuration. Nous pouvons ainsi structurer les données envoyées à l'interface.

Pour poursuivre, pour la documentation de « Python », nous avons utilisé « Sphynx », comme lors de la remise de mi-session. Ce générateur de documentation de « Python » permet de documenter les classes et les méthodes avec les commentaires dans le code.

Finalement, pour être en mesure d'effectuer les tests, nous avons utilisé des bancs d'essais[4]. La librairie « Jasmine » a aussi été utilisée. « Jasmine » nous permet d'utiliser les « spy » qui s'assure de regarder la valeur de retour des fonctions que nous testons. Nous pouvons ainsi comparer la valeur de retour de la fonction avec la valeur attendue.

On peut donc tester les résultats des fonctions et s'assurer qu'elles retournent bien le résultat attendu.

2.6 Quelques méthodes de tests appliquées pour valider la solution

Dans le but d'être en mesure de vérifier le bon fonctionnement du système et la qualité de celui-ci, nous avons effectué différents tests afin de confirmer ou d'infirmer cela. Dans le cas où le bon fonctionnement du système n'était pas confirmé à l'aide des tests, nous devions arranger la fonctionnalité en question et la retester jusqu'à ce que son bon fonctionnement soit confirmé.

Nous avons continué à appliquer la méthode de fonctionnement que nous avons décrite dans le rapport de mi-session. Nous allons l'expliquer et montrer comment elle s'applique pour la remise finale.

Tout d'abord, le processus de révision que nous avons établi demande à ce que nous révisions le code écrit et la documentation écrite. Tous les membres de l'équipe doivent participer à cette étape, car elle est d'une grande importance. Sans la confirmation du bon fonctionnement du système, nous ne pouvons assurer de la satisfaction du client. Ainsi, la révision du code et de la documentation par tous les membres nous donnera plus de chance de détecter les erreurs possibles.

Pour ce qui est de la révision de code, comme mentionné précédemment, il faudra passer le code à travers différents tests. L'analyse pour le code se fait de façon continue, comme nous l'avons mentionné dans le rapport de mi-session. Cette méthode s'applique bien dans notre cas, il s'agit d'un gros projet qui s'est échelonné sur plusieurs mois. Alors, le fait de faire l'analyse en continu nous a permis de détecter les erreurs au fur et à mesure que nous développions l'application, plutôt que de laisser l'analyse à la fin. Bien que l'écriture des tests n'était pas très avancée lors de la mi-session, nous nous sommes rattrapés pour la remise de fin de session et nous pouvons dire avec confiance que le système fonctionne comme attendu. Pour être en mesure de dire que le code fonctionne comme attendu, nous avons écrit des tests unitaires pour les fonctions, surtout au niveau serveur et logique. Il n'était pas nécessaire d'en faire pour l'interface, car elle ne fait qu'afficher les valeurs qu'elle reçoit du serveur et de la logique. Toutefois, à défaut d'écrire des tests unitaires pour l'interface utilisateur, les fonctionnalités ont dû être testées manuellement. Pour ce qui est des tests unitaires pour le serveur et la logique, les tests ont été écrits de façon incrémentale. Lorsque le test retourne le bon résultat, on

peut alors dire que la fonction marche bien. Toutefois, si elle ne retourne pas le résultat attendu, il est de la responsabilité de la personne qui a écrit la fonction de l'arranger. Pour ce qui est des librairies utilisées pour effectuer les tests, cela a été décrit dans la section 2.5. Par ailleurs, pour les tests effectués sur l'interface utilisateur, comme mentionnée précédemment, ils ont été faits manuellement. Les fonctionnalités présentes sur l'interface, par exemple les graphiques ou le filtrage des maisons, ont toutes été analysées par différents membres de l'équipe pour s'assurer que ça fonctionne bien. Par ailleurs, nous avons également remis une version beta[5] au client le 5 avril 2023, soit un peu plus de 2 semaines avant la remise afin d'avoir son avis sur le système. De cette façon, avec ses rétroactions, nous avons pu arranger quelques petits éléments pour la remise finale.

Pour ce qui est de la documentation qui devait être écrite, un plan de révision a également été développé. En effet, il est important de s'assurer de la qualité des livrables, car il s'agit de traces écrites de l'état d'avancement du projet. La documentation du projet a également été lu par les membres de l'équipe pour s'assurer qu'elle soit claire. La présence de commentaires dans le code permet de faciliter la compréhension du code pour les futurs développeurs. Lorsqu'une fonctionnalité est prête à être ajoutée au système, nous devons faire un « pull-request » sur Gitlab et les membres de l'équipe ont la responsabilité de relire le code pour accepter la demande d'ajout sur la branche principale. En faisant cela, on peut recevoir les rétroactions des autres membres de l'équipe. Ainsi, tout le monde participe activement tant au développement du système qu'au processus de révision.

3. Gestion du projet

3.1 Identification des tâches principales du projet

Les principales tâches que nous devions réaliser ont bien été identifiées au début du projet. Bien que les objectifs n'aient pas changé, les tâches ont évolué.

Lors du début de session, nous avons fait une rencontre avec le client afin d'identifier les principales tâches du projet. Nous avons ensuite également discuté des attentes concernant le projet et des tâches qui sont plus prioritaires aux autres. Nous avons ensuite décomposé les tâches en sous-tâches plus facilement réalisables. De cette façon, nous avons pu présenter de nouveaux éléments lors de nos rencontres

hebdomadaires avec le client qui avaient lieu, en grande majorité, les vendredis matin à 10h.

Tout d'abord, la principale tâche du projet reste la refonte du code. Autant du côté serveur et logique que du côté de l'interface utilisateur, la principale tâche consistait à revoir le code au complet et à l'améliorer. Le code reçu en début de session était écrit de façon incrémentale. Cela veut dire qu'il n'y avait pas vraiment de logique qui ne supportait le code, ni même une architecture. Cela nous a compliqué la tâche, car nous devions comprendre du code qui ne respectait pas les normes de codage et d'assurance qualité. Bien que le code reçu au départ fût compilable, la façon dont il a été écrit le rend difficilement maintenable. Notre rôle principal était d'en faire la refonte pour le rendre modulable. De plus, il fallait également refaire l'interface utilisateur en le rendant plus moderne et plus clair. L'interface que le client avait développée était très simple. Elle contenait les éléments nécessaires pour le sujet de recherche sur lequel il travaille, mais elle n'était pas attirante et elle n'incitait pas les gens à l'utiliser. Nous l'avons donc utilisé comme inspiration pour déterminer quels éléments nous allions présenter sur la nouvelle interface et quels éléments nous voulions ajouter. De plus, la disposition des différentes composantes a changé comparativement à la version de base du client. Puisque le client a pour but que le logiciel que nous développons en fonction de son sujet de recherche devient la référence pour ce type de programme, il est primordial d'avoir une interface qui saura plaire à tous les chercheurs pour les inciter à l'utiliser. Les critères que nous nous sommes fixés pour évaluer la nouvelle interface sont: la simplicité, la facilité d'utilisation, l'allure générale, la clarté et la présentation des informations, comme nous en avons discuté lors de la remise de mi-session.

Outre l'interface utilisateur, un autre élément qu'il a fallu améliorer afin d'encourager les chercheurs à utiliser l'application est l'architecture développée. Comme mentionné précédemment, le code de départ n'avait pas de logique ni d'architecture. Le code écrit était fait de façon incrémentale. Cette façon de faire qui consiste à ajouter des fonctions une après l'autre sans avoir d'architecture n'est pas idéale pour le maintien du code ni pour encourager d'autres personnes à utiliser le code, car il est très difficile de s'y retrouver et de comprendre ce qui a été fait. Nous avons également noté la présence de grosses fonctions qui faisaient plusieurs dizaines de lignes. Pour rectifier cela, nous sommes passés à travers tout le code et à travers chacune des fonctions pour comprendre le comportement de chacune d'elles. L'aide du client pour cette partie a bien été appréciée, car nous avons tenu quelques réunions avec lui pour qu'il puisse nous expliquer les parties qui étaient plus difficiles à comprendre sans ses explications. En faisant la refonte du code sur le côté logique et serveur, il devient beaucoup plus facile pour les futurs chercheurs d'y ajouter leurs propres agents et fonctionnalités.

Ensuite, une autre tâche qui était de notre responsabilité était la rédaction de la documentation. La version de base du client n'était pas documentée. Sans documentation, il est difficile de comprendre le code et le fonctionnement du programme développé. C'est pour cela que nous avons tenu des réunions avec le client à quelques reprises. La présence de la documentation va permettre aux futurs chercheurs d'ajouter facilement leurs agents et leurs fonctionnalités. La documentation écrite contient tous les éléments nécessaires pour la compréhension du code, comment lancer le programme, comment ajouter des agents, la description des fonctions, etc. Bref, tous les éléments nécessaires pour la compréhension du code. Cela est d'autant plus important dans notre situation, car le client souhaite que le code soit de source libre[6] La documentation est bien faite, car elle contient la description du code de haut niveau et de bas niveau. Il s'agit donc d'un guide pour les futurs utilisateurs.

Nous étions également responsables de l'intégration de « Wandb ». « Wandb » est un mécanisme qui permet de garder les traces des expériences effectuées. Il permet également de comparer des résultats ou de reproduire certaines expériences par exemple. C'est surtout utile dans le monde de l'intelligence artificielle. C'était une tâche très importante aux yeux du client, alors nous nous sommes assurés que cela soit bien fonctionnel.

Finalement, une autre tâche générale sur laquelle nous avons travaillé est l'implémentation et l'amélioration des fonctionnalités du programme. Certaines des fonctionnalités sur lesquelles nous avons travaillé étaient déjà présentées dans la version de base du client. Parmi celles-ci, il y a les graphes. Les graphes sont une fonctionnalité très importante pour la visualisation des données. Nous avons amélioré l'allure des graphes sur la page principale. Nous avons également ajouté des fonctionnalités sur les graphes. Il est maintenant possible de se déplacer à travers le graphe et de les agrandir pour mieux visualiser certaines données. Un autre ajout concernant cette fonctionnalité est la présence des graphes pour chaque maison individuellement. Ainsi, en cliquant sur une maison, il est possible de voir la courbe de différentes données pour la maison elle-même. Ainsi, bien que la fonctionnalité des graphes était présente sur la version de base du client, nous l'avons grandement amélioré. Outre cette fonctionnalité, il y a également la fonctionnalité du filtrage de maison que nous avons ajouté. Cette fonctionnalité permet de sélectionner les maisons que nous voulons voir en fonction de certains paramètres qui sont disponibles sur la barre de navigation. Nous avons également ajouté un bouton d'arrêt temporaire de la simulation et un gradateur[7], comme nous avons mentionné précédemment. Ceux-ci permettent aux chercheurs de se déplacer à différents moments de la simulation.

Bien que la plupart des tâches ont déjà été décrites dans le rapport de mi-session, nous avons continué à travailler sur ces mêmes tâches pour la remise de fin de session. Nous estimons que le travail que nous avons réalisé en début de session pour identifier les tâches à compléter a été très bien fait et cela nous a facilité la compréhension générale du projet, car nous avions les lignes directrices et nous connaissions déjà vers quoi on se dirigeait. Ainsi, même si la plupart des tâches étaient décrites dans le rapport de mi-session, elles n'étaient pas toutes complètes lors de la présentation de mi-session. Nous en avions parlé, car nous savions que cela faisait partie du projet. Maintenant arrivés à la fin du projet, nous pouvons dire avec confiance que nous avons réalisé ce qui nous était demandé.

3.2 Répartition des tâches et responsabilités dans l'équipe

Pour réaliser ce projet, nous avons été affectés à une équipe de 6 personnes. Nous estimons que la taille de l'équipe est parfaite pour le niveau de difficulté et de tâches à réaliser pour ce projet. En effet, ce projet contient plusieurs éléments complexes et le travail à réaliser est assez gros. La complexité du travail vient surtout du fait qu'il n'y avait pas d'architecture ou de logique qui supportait le code de départ. La revue complète du code, tant au niveau serveur/logique que de l'interface utilisateur a pris beaucoup de temps. Il fallait commencer par comprendre ce qui a été fait pour savoir vers quoi l'on allait se diriger. Ensuite, il a fallu réécrire la grande majorité du code. Pour ce qui est de la partie intelligence artificielle du projet, nous avons pu réutiliser directement ce que le client avait déjà écrit comme code. Ainsi, pour réaliser ce grand projet, nous avons séparé notre équipe en sous-groupes.

Notre projet était divisé en deux grandes parties. Du début de la session jusqu'à la mi-session, cela représentait la première étape du projet. La deuxième étape du projet commençait de la mi-session jusqu'à la remise de la fin de session. Afin d'être en mesure de se séparer les tâches entre coéquipiers, il a fallu passer à travers tout le code. Nous avons fait cela en équipe, pour la majorité du code. Nous avons tenté de comprendre les grandes parties de façon à identifier les tâches et que chaque personne pourrait faire en fonction des forces et intérêts. Pour ce qui est de la première étape du projet, notre équipe était divisée en deux, soient une partie pour l'interface utilisateur et une autre partie pour la logique et le serveur. Nous étions trois membres pour chaque partie. Ainsi, Victor, Ghazi et Alex ont commencé par travailler sur la partie logique et serveur alors que Mohamed, Alison et Johnny se sont occupés de l'interface utilisateur.

Pour la partie logique et serveur, la principale tâche était de développer une architecture solide et modulable pour le code. Par ailleurs, la refonte du code faisait également partie des tâches à effectuer. Par refonte du code, on parle de réécrire les fonctions, les classes et à revoir tous les fichiers de façon à respecter les normes de codage et l'assurance qualité. Le but de cette partie était surtout de réduire la taille de plusieurs fonctions. On voulait aussi s'assurer que chaque fonction s'occupe d'effectuer une tâche en particulier, plutôt que de faire de grosses fonctions qui effectuent plusieurs tâches. De plus, il fallait aussi faire de la documentation de code et cela entrait dans la partie du serveur et de la logique. Nous voulions nous assurer que chaque fonction soit décrite et qu'il y ait une explication claire pour que n'importe qui avec des connaissances de base en programmation puisse ajouter des agents et des fonctionnalités ou apporter toute autre sorte de modifications. La partie serveur et logique s'occupe aussi d'envoyer les données à l'interface utilisateur. Il fallait donc s'assurer d'y établir une bonne communication pour pouvoir afficher les données de chaque maison et d'afficher les graphiques pour toutes les maisons et pour les maisons individuelles. Ensuite, la partie serveur et logique comporte également des éléments d'intelligence artificielle. Le client nous a spécifié en début de session qu'on pouvait réutiliser ces parties directement et que nos tâches ne concernaient pas cette partie. Nous avons donc répondu aux attentes du client. Pour ce qui est du langage utilisé, nous avons gardé le « Python », puisque c'était le langage utilisé par le client lorsqu'il nous a remis sa version du programme en début de session.

Pour poursuivre, la seconde partie du projet consistait à refaire l'interface utilisateur. Pour cette partie, lors du début de session, nous avons tenu une réunion avec le client afin de lui expliquer comment nous allions nous y prendre. Au départ, le client préférait que nous utilisions « Python » et la librairie « Pydantic » notamment pour développer l'interface utilisateur. Cette façon de faire allait nous demander beaucoup de temps et elle allait nous limiter au niveau du potentiel de l'application. Nous avons donc suggéré au client d'utiliser des façons de faire beaucoup plus modernes et adapter à ses besoins. Nous avons donc opté, avec son approbation, pour les langages « HTML », « CSS » et « TypeScript ». Nous avons aussi utilisé le cadriciel « Angular material », comme mentionné dans la section 2.5 de ce rapport. Cette partie a été développée par Alison, Mohamed et Johnny. Nous nous sommes inspirés de la version de base du client afin de déterminer les éléments nécessaires à mettre sur la nouvelle interface. Une fois que cela a été décidé, nous avons monté une maquette que nous avons présentée très tôt dans la session pour avoir l'approbation du client le plus rapidement possible et, ainsi, commencé à travailler sans perdre de temps. Tout le côté visuel de l'application était contenu dans cette partie. Le choix des couleurs, la disposition des éléments, l'écriture, la réception des données du serveur pour les faire afficher sur l'interface ne sont que quelques exemples des responsabilités de l'interface utilisateur.

Lors de la deuxième moitié de la session, donc pour la deuxième étape du projet, la séparation des tâches a légèrement changé. Puisque l'interface utilisateur a bien avancé, il n'était plus nécessaire d'avoir 3 membres pour continuer les tâches en lien avec cette partie. Deux membres qui s'occupaient initialement de l'interface utilisateur sont allés dans le sous-groupe du serveur et logique afin d'aider à avancer le travail. Nous avons donc su nous adapter en fonction des situations auxquelles nous faisions face durant tout le projet. Nous ne voulions pas perdre de temps ou qu'un membre de l'équipe ne travaille pas. Nous tentions toujours de trouver des tâches à effectuer afin d'être en mesure d'atteindre nos objectifs et ceux du client. Lorsqu'un membre avait terminé ses parties, il était de sa responsabilité d'en informer l'équipe pour trouver d'autres tâches à faire. Les tâches qui se sont ajoutées à celles de mi-session étaient de faire afficher les graphiques sur l'interface utilisateur avec les données reçues du serveur, terminer le filtrage des maisons, l'intégration de « Wandb » et terminer la documentation, entre autres. Pour plus de détails sur les tâches, se référer à la section 2.1 du rapport.

La liste ci-dessous présente les principaux rôles de chaque membre de l'équipe pour la première moitié de la session qui s'est terminée avec la remise de mi-session, soit le 24 février 2023.

Ghazi Ben Achour

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Tests.

Alex Hua

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Tests.

Johnny Khoury

- Développeur l'interface utilisateur;
- Participation dans le développement du serveur et de la logique;
- Assurance qualité;
- Secrétaire lors des rencontres;
- Tests.

Alison Nemr

- Développeur l'interface utilisateur;
- Participation dans le développement du serveur et de la logique;
- Assurance qualité;
- Tests.

Victor Vergeau

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Coordonnateur;
- Tests.

Mohamed Zakaria

- Développeur l'interface utilisateur;
- Participation dans le développement du serveur et de la logique;
- Assurance qualité;
- Tests.

La liste ci-dessus présente les principaux rôles des membres de l'équipe. Il est possible de voir que tous les membres de l'équipe touchent à toutes les parties du projet. En effet, que ce soit du côté interface utilisateur ou du côté du serveur et de la logique, tout le monde a une connaissance de ce qu'il se passe et participe au développement en donnant son avis par exemple.

Pour ce qui est de la seconde moitié de la session, donc du 24 février 2023 jusqu'à la remise du projet, soit le 20 avril 2023, les rôles ont légèrement changé, comme

mentionnés précédemment. La liste ci-dessous présente les rôles pour la deuxième étape du projet.

Ghazi Ben Achour

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Tests.

Alex Hua

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Tests.

Johnny Khoury

- Développeur du serveur et de la logique;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Secrétaire lors des rencontres;
- Tests.

Alison Nemr

- Développeur du serveur et de la logique;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Tests.

Victor Vergeau

- Développeur logique et serveur;
- Participation dans le développement de l'interface utilisateur;
- Assurance qualité;
- Coordonnateur;
- Tests.

Mohamed Zakaria

- Développeur l'interface utilisateur;
- Participation dans le développement du serveur et de la logique;
- Assurance qualité;
- Tests.

Comme mentionné précédemment, notre équipe a dû s'adapter à l'évolution des tâches et au niveau de difficulté de celles-ci qui augmentait plus on avançait dans le travail. Avec ces changements effectués, donc la migration de deux membres de l'équipe vers le côté serveur et logique, nous avons pu terminer les tâches requises et remettre un travail qui répondait aux attentes du client, sans compromettre l'interface utilisateur. Bien que nous ayons pris de l'avance sur l'interface utilisateur pour la première moitié de la session, il restait beaucoup de tâches à faire. Nous avons évalué qu'une personne aurait assez de responsabilités pour s'en occuper seule pendant que les autres membres de l'équipe travaillent sur le serveur et la logique. Cette façon de faire a demandé beaucoup de communication et de transparence dans l'équipe afin de ne pas causer de confusion dans l'équipe. Nous avons su relever ce défi avec succès. La communication dans l'équipe a toujours été bonne, donc cela n'était pas un obstacle durant la session.

Par ailleurs, une autre responsabilité que chaque membre de l'équipe devait assumer était les participations aux rencontres SCRUM que nous tenions entre les membres de l'équipe hebdomadairement, parfois même 2 ou 3 fois par semaine. Lorsque des rencontres de ce type avaient lieu, tous les membres devaient participer activement à la rencontre en donnant l'état d'avancement de ses tâches et parler de ses difficultés rencontrées, s'il y a lieu. De plus, les rencontres avec le client que nous avions tous les vendredis matin, pour la majorité de celles-ci à 10h, la présence de tout le monde était également requise, tout comme pour les rencontres hebdomadaires en présentiel avec l'enseignant les mardis à 15h15. Durant ces rencontres, tout le monde devait prendre la parole et parler de ses tâches et donner un résumé de ce qui a été présenté au client la semaine précédente et ce que nous voulions présenter au client pour la semaine courante. Ainsi, les présences et la participation active à toutes les rencontres faisaient partie des responsabilités des tous.

3.3 Principales difficultés de gestion rencontrées durant la deuxième partie du projet

Comme mentionné dans la section 3.2, le projet sur lequel nous avons travaillé était un projet de grande envergure. Plusieurs tâches devaient être effectuées et le remaniement des sous-groupes dans l'équipe lors de la deuxième moitié de la session a apporté son lot de difficulté. Nous allons en parler dans cette section.

Tout d'abord, lors d'un gros projet comme celui que nous avons réalisé, il faut s'assurer de certaines normes dans l'équipe. En début de session, nous avons planifié certaines responsabilités et normes dans l'équipe qu'il faut que chaque membre respecte. Ceux-ci sont présentés dans le tableau 1 placé ci-dessous.

Tableau 1: Normes et responsabilités des membres de l'équipe

Transparence
Communication
Respect
Confiance

Le tableau 1 présente certaines normes sur lesquelles nous nous sommes mis d'accord, en équipe, pour le déroulement du projet. Pour ce qui est de la transparence, cela concerne les parties individuelles de chaque membre. Lorsqu'une personne a fini de travailler sur une tâche, il doit en informer les autres membres de l'équipe afin de les garder à jour sur l'état d'avancement du travail. De plus, le membre qui a fini de travailler sur une tâche doit en commencer une nouvelle. Nous nous sommes mis d'accord, en équipe, que nous ne voulions pas qu'un membre ralentisse l'avancement du travail ou qu'il fasse semblant de travailler. C'est pour cela que la transparence est importante dans l'équipe. La transparence concerne aussi le cas où un membre de l'équipe a besoin d'aide pour sa partie. Il est de sa responsabilité d'en aviser les autres. Il est normal d'être bloqué sur du code, surtout pour un travail de cette taille. Toutefois, après avoir essayé à déboguer, si le membre n'est pas capable de trouver l'erreur après une certaine période, il doit en aviser les autres membres pour ne pas ralentir le développement du travail. En restant bloqué trop longtemps, cela n'apporte rien de positif au projet. Pour poursuivre, la communication est importante dans l'équipe et rejoint la norme de la transparence sur laquelle nous nous sommes mis d'accord en équipe. Tant au niveau professionnel que relationnel, si un membre sent qu'il doit dire quelque chose ou qu'il veut donner son avis,

il doit le faire. Toutefois, il doit le faire dans le respect pour éviter toute forme de conflit dans l'équipe. Finalement, la norme de confiance concerne les parties individuelles de chaque membre de l'équipe. Lorsqu'un membre donne une échéance pour la remise de sa partie ou qu'il donne son avis sur le travail, il doit le faire pour le bien de l'équipe et non pour son bien personnel. La relation de confiance dans l'équipe est importante pour assurer l'atteinte des objectifs fixés en début de session. Ces normes sur lesquelles nous nous sommes mis d'accord concernent les relations entre les membres de l'équipe, mais également la relation avec le client. Il est important d'assurer la transparence, le respect, la confiance et la communication avec le client. Nous voulons que notre client ait une expérience agréable avec notre équipe. Nous pensons qu'en appliquant ces normes, il sera satisfait de son expérience.

Nous nous sommes rappelé ces normes tout au long de la session, notamment lors de la rencontre HPR que nous avons eue le 21 février 2023.

Pour ce qui est des difficultés au niveau de la gestion, nous avons connu des défis, mais pas vraiment de difficultés. Tout au long du projet, la communication était bonne et les normes que nous nous sommes fixées ont été respectées par tous les membres. Les défis que nous avons dû relever ont eu lieu lors de la transition des membres de l'équipe du côté interface utilisateur vers le côté serveur et logique. Il a fallu faire preuve de très bonne communication pour s'assurer de la compréhension des différentes parties. Il n'y a pas eu de moment de tension dans l'équipe. Tout le monde a fait preuve de patience et de compréhension et cela a contribué au maintien de la bonne ambiance de travail dans l'équipe. À un certain moment dans la session, soit la semaine après la semaine de relâche, nous avons eu beaucoup de travaux à remettre dans nos autres cours. Cela a contribué au niveau de stress dans l'équipe, mais nous étions tous dans la même situation et nous avons tous fait preuve d'empathie. L'ambiance est donc restée bonne dans l'équipe, mais lors de cette semaine, nous n'avons pas beaucoup avancé le projet. Toutefois, nous avons su nous reprendre lors des semaines suivantes.

Pour ce qui est de la relation que nous avons eue avec le client, elle est restée très bonne et très professionnelle tout au long du projet, tant pour la première partie que pour la deuxième partie. Nous avons su assurer une bonne communication avec lui. Par exemple, lors de la semaine du 2 avril 2023, nous devions remettre une version bêta du projet au client pour qu'il puisse voir le travail et qu'il nous donne des rétroactions et son avis sur le travail réalisé. Toutefois, en raison de la tempête de verglas et des pannes d'électricité qu'il y a eu, nous n'avons pas pu lui remettre le travail à la date que nous lui avions dite. Nous avons donc fait preuve de transparence et nous avons communiqué

avec lui dans le respect pour lui expliquer notre situation. Le client a bien réagi et il a compris la situation dans laquelle nous étions et il a accepté que nous lui donnions la version bêta lorsque l'électricité serait revenue. La communication dans ce genre de situation est importante pour assurer une bonne expérience avec le client et pour garder une relation professionnelle agréable.

À part ces quelques situations, le projet s'est très bien déroulé et la gestion de l'équipe a été excellente. Tous les membres de l'équipe sont satisfaits du déroulement du travail.

4. Conclusion

4.1 Retour sommaire sur le travail complété et ce qui n'a pu être réalisé (Q3.5)

Certaines tâches qui ont été discutées en début de session avec le client ont été identifiées comme étant optionnelles. Certaines de celles-ci ont été réalisées par l'équipe, alors que d'autres n'ont pas été complétées. Nous allons analyser ces tâches et discuter des raisons pour lesquelles elles n'ont pas été faites.

Tout d'abord, commençons par dresser un sommaire du travail réalisé. Comme mentionné précédemment, nous avons identifié les tâches importantes à effectuer en début de session. Ces tâches étaient majoritairement liées à la refonte du code et au développement d'une architecture claire et bien définie, en plus d'être modulable. Le fait de développer une architecture claire et bien définie veut dire que nous ne voulons pas utiliser le code que le client nous a fourni en début de projet. Le code de départ était écrit de façon incrémentale et il n'y avait pas de logique. C'était donc très difficile d'y ajouter des fonctionnalités ou des agents. C'est pour cela que le fait de développer une application modulable est important. Part modulable, on veut dire que les chercheurs puissent ajouter différentes fonctionnalités ou agent qu'ils souhaitent avoir. Ce travail a été fait et nous l'avons accompli avec succès.

À la mi-session, les graphiques pour la grille de maisons et les graphiques pour chaque maison spécifique n'étaient pas encore commencés. Ceux-ci étaient présents dans la version originale fournie par le client. Ils représentent une partie importante du projet, car ils permettent de voir les détails des maisons de façon claire et d'en évaluer la tendance. C'était dans les plans que cette fonctionnalité ne soit pas intégrée à notre version pour la

mi-session. Le client a toutefois voulu spécifier que ce requis était nécessaire et qu'il fallait s'y attarder le plus rapidement possible dû à sa complexité. Alors, nous avons commencé à travailler sur cette tâche la journée même de la présentation de mi-session. Avec la fonctionnalité des graphiques, il est possible d'agrandir les graphes pour mieux voir la courbe à un instant donné et il est également possible de se déplacer le long de la courbe. Cela permet de sélectionner une plage de temps pour laquelle on veut visualiser les données.

La fonctionnalité de filtrage et de triage était déjà presque terminée lors de la présentation de mi-session, mais le client était un peu inquiet que cette tâche allait prendre du temps sur d'autres fonctionnalités. Il ne voulait pas que nous passions trop de temps à implémenter cette fonctionnalité, car bien qu'elle soit intéressante, elle n'est pas primordiale au projet. Il nous a donc spécifié qu'il préférerait qu'on se concentre sur les tâches prioritaires. Toutefois, nous avons tout de même pu réussir à terminer cette fonctionnalité quelques jours plus tard afin de ne pas accumuler du retard.

Le client nous a spécifié trois tâches prioritaires à la suite de la présentation de mi-session. Parmi celles-ci, il y avait l'intégration de « Wandb ». Du côté serveur et logique, cette tâche nous a pris assez de temps à implémenter. En effet, sans nécessairement avoir sous-estimé la tâche, nous ne nous attendions pas à ce qu'elle prenne autant d'ampleur.

Ensuite, nous avons également ajouté d'autres fonctionnalités nécessaires, telles que le gradateur. Cette fonctionnalité permet à l'utilisateur de choisir le temps de la simulation. De plus, un bouton d'arrêt complet[8] et de lecture[9] a été ajouté. Nous avons également développé un bouton d'attente[10] qui permet d'interrompre temporairement la simulation. La différence entre le bouton d'arrêt complet et le bouton d'attente est que le bouton d'arrêt complet efface toutes les données accumulées depuis le début de la simulation; il permet de tout réinitialiser. Le bouton d'attente n'efface pas les données accumulées; il ne fait qu'arrêter la simulation pour être reprise à un moment choisi par l'utilisateur.

Pour ce qui est des tâches que nous n'avons pas pu compléter, il s'agit de tâches qui ont été jugées optionnelles en début de session. Le client nous avait spécifié qu'il aimerait que ces tâches soient implémentées dans l'application, mais que si nous n'avions pas le temps de les faire, cela ne lui causerait pas de problème, car c'est simplement des fonctionnalités pour aller plus loin.

Parmi ces tâches, il y avait notamment l'implémentation des fils électriques dans l'application. De cette façon, il serait possible de voir comment les maisons sont branchées sur le réseau électrique et comment elles sont reliées les unes aux autres. Il

s'agit d'une fonctionnalité intéressante pour la suite de la recherche dans le futur, mais pour le moment, cette fonctionnalité ne limite pas le potentiel de l'application développée. C'est pour cela que nous nous sommes concentrés sur les tâches prioritaires aux yeux du client. Pour faire cela, il aurait également fallu implémenter un algorithme qui permet de placer les maisons ou les bâtiments en réduisant au maximum la distance entre elles. Il aurait fallu, par exemple, que les maisons qui sont voisines dans un quartier le soient également dans notre application.

Par la suite, une autre fonctionnalité que nous aurions voulu implémenter est l'ajout d'un agent. En effet, nous aurions voulu implémenter l'agent développé dans le cadre de la recherche du client, nommé « Tarmac », mais le client n'a pas pu nous l'envoyé à temps pour que nous puissions l'ajouter. Cela aurait été un test idéal pour s'assurer que notre application soit vraiment modulable et que notre travail remplit bien les critères, mais puisque le client ne nous a pas envoyé son agent, nous n'avons pas pu l'implémenter dans notre application.

Tout cela a été discuté avec le client et il est conscient des tâches effectuées et celles que nous n'avons pas faites. Nous avons été transparents avec lui tout le long du projet pour ne pas qu'il ait de mauvaises surprises. Nous sommes donc satisfaits du travail effectué par notre équipe.

4.2 Causes des succès et difficultés rencontrées (Q3.6)

Plusieurs éléments sont entrés en jeu durant le projet. Il y a bien évidemment eu des hauts et des bas. Dans cette section, nous allons donner une vue d'ensemble sur le déroulement du projet.

Nous évaluons le projet comme étant un succès pour notre équipe. Il y a plusieurs éléments que nous considérons comme des succès. Le tableau 2 ci-dessous montre ces éléments.

Tableau 2: Présentation des succès tout au long du projet

Éléments considérés comme des succès
Gestion de l'équipe
Gestion de l'assignation des tâches
Communication entre les membres de l'équipe
Communication avec le client
Communication avec l'enseignant
Gestion du temps
Séparation des tâches
Bâtir une relation de confiance avec le client
Réalisation des tâches demandées

Comme le montre le tableau 2, différents éléments ont été des succès dans notre équipe. Pour ce qui est de la gestion de l'équipe, notre approche qui est de la forme décentralisée nous a grandement été utile, comme le montre la figure 1.

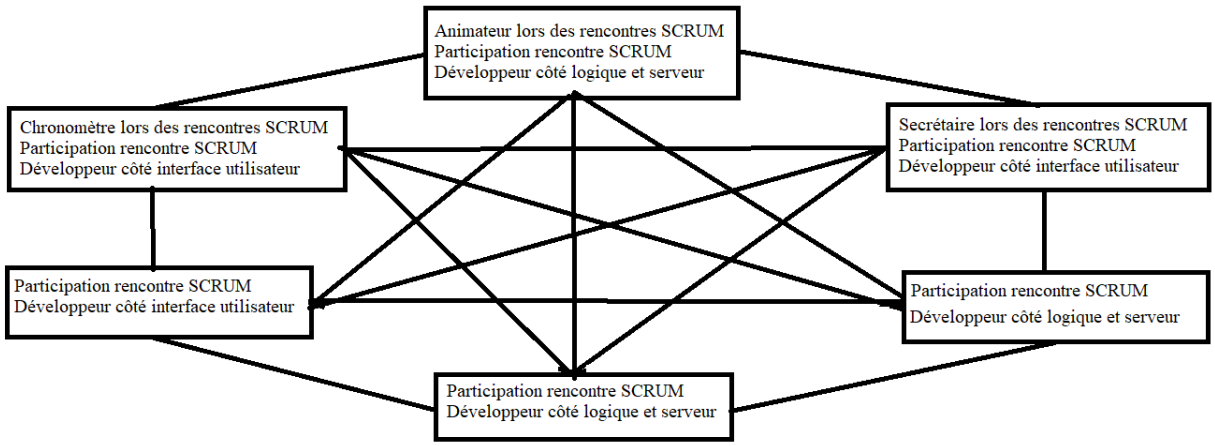

Figure 7: Présentation des rôles de chaque membre et de la méthode décentralisée de l'équipe

La figure 7 a déjà été présentée dans le rapport de mi-session, mais nous avons jugé pertinent de la présenter dans le rapport final également. En effet, en raison de la complexité du travail et de la taille de celui-ci, la méthode décentralisée s'avérait le meilleur choix pour nous pour ce qui est du mode de fonctionnement. Nous en avons fait

le choix en début de session et nous avons gardé cette méthode pour la suite du projet. Nous avons estimé qu'il était important pour tous les membres de connaître l'état des tâches de tous les autres membres et que tous devaient communiquer, même si certaines tâches n'étaient pas reliées. La connaissance générale de l'état d'avancement du travail est quelque chose que nous trouvions important en plus de pouvoir donner son avis sur le travail des autres membres. Par ailleurs, la figure 7 montre aussi que chaque membre doit participer aux rencontres SCRUM. La simple présence à ces rencontres n'est pas suffisante; il faut participer à celles-ci. Nous les tenions hebdomadairement, parfois 2 ou 3 fois par semaine.

Pour poursuivre, dans le tableau 2, nous avons aussi placé la gestion de l'assignation des tâches comme étant un succès pour l'équipe. En effet, les tâches générales qui devaient être effectuées ont été identifiées en début de session après la première rencontre avec le client. Par la suite, ces tâches ont été divisées en sous-tâches et chaque membre de l'équipe avait pour rôle de choisir les sous-tâches pour lesquelles il voudrait travailler en fonction des intérêts et des forces. Heureusement, nous avions une équipe hétérogène, c'est-à-dire que chaque personne avait des intérêts et des forces différentes des autres membres. Ainsi, l'assignation des tâches n'était pas quelque chose de difficile à faire.

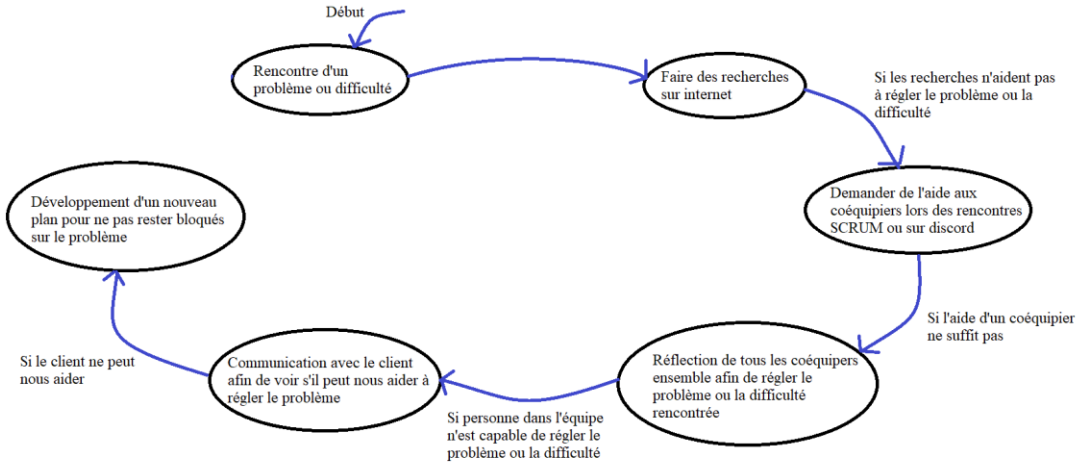

Figure 8: Plan à suivre pour résoudre les problèmes lors du développement

La figure 8 montre le plan que nous avons développé en début de session pour être en mesure de gérer les problèmes qui peuvent survenir lors du développement du code. Ce plan qui a été présenté lors de la mi-session a été quelque chose de positif tout au long du projet. En effet, le fait d'avoir des étapes claires et écrites sur lesquelles tout le monde s'est mis d'accord nous a permis d'avancer dans la bonne direction. Cela a également permis d'améliorer la communication, comme nous allons l'expliquer dans le paragraphe suivant.

Ensuite, la communication entre les membres de l'équipe était une de nos forces tout au long de ce projet. En effet, nous avons mis plusieurs éléments en place pour assurer une bonne ambiance de travail. Le fait que nous nous connaissions déjà, pour la majorité, a contribué au maintien de la bonne communication au sein de l'équipe. Nous avons aussi fait notre possible pour que tous se sentent bien intégrés à l'équipe et que tous puissent participer activement aux prises de décisions. Ce sentiment d'importance dans l'équipe est primordial si on veut atteindre les objectifs que nous nous sommes fixés.

Par ailleurs, le tableau 2 présente aussi la communication avec le client comme étant une force. En effet, la communication est un élément très important dans le maintien d'une bonne relation, surtout avec un client. Nous avons fait le choix, en équipe, que pour le bien du projet, il fallait être le plus transparent possible afin d'établir une relation de confiance avec le client. Que ce soit pour les points positifs ou négatifs, il fallait communiquer avec le client pour ne pas qu'il ait de mauvaises surprises. Nous avons tenté de maintenir cela tout au long du projet et nous pensons que cela fait partie de nos succès. Cela s'applique également avec l'enseignant lors de nos rencontres hebdomadaires qui avaient lieu le mardi en présentiel. Nous tentions de lui expliquer le plus possible l'état d'avancement de nos tâches, les éléments bloquants, les bons points et les points négatifs. Nous voulions également être transparents avec lui.

Finalement, la gestion du temps nous a permis de réaliser toutes les tâches prioritaires pour le client. Nous sommes donc très satisfaits du résultat que nous avons atteint.

Pour ce qui est des difficultés rencontrées, le tableau 3 ci-dessous présente quelques éléments.

Tableau 3: Présentation des difficultés rencontrées tout au long du projet

Éléments considérés comme des difficultés
Intégration du client dans les prises de décision
Compréhension du code de départ que le client nous a fourni
Faire plus de démonstrations sur le fonctionnement du nouveau système

Pour ce qui est des difficultés rencontrées, nous en avons noté 3 et nous allons les développer.

Tout d'abord, le tableau 3 montre que l'intégration du client dans les prises de décision n'était pas chose facile à faire. En effet, le client nous avait mentionné quelques fois que nous ne devions pas hésiter à lui demander son avis sur certains éléments. Bien que nous l'ayons fait quelques fois, nous aurions pu lui demander son avis plus d'avis et le faire participer à nos prises de décisions. Lorsque nous devions prendre des décisions sans le client, nous le faisions toujours en consensus de groupe et pour le bien du projet. S'il le fallait, nous expliquions nos choix au client et nous prenions son avis. S'il n'était pas d'accord avec quelque chose, nous faisions la modification pour qu'il soit satisfait. Bien que nous soyons les experts en programmation, il faut se rappeler que la décision finale appartient au client et il est dans notre rôle de le conseiller du mieux qu'on peut.

Ensuite, une autre difficulté que nous avons rencontrée tôt dans le projet et qui est présentée dans le tableau 3 était la compréhension du code de départ. En effet, comme nous l'avons déjà mentionné, le code a été écrit de façon incrémentale sans aucune documentation. Pour remédier à cette problématique, nous avons tenu une rencontre avec le client qui nous permettait de bien comprendre le code écrit. Une fois la rencontre terminée, nous étions en mesure de commencer le nouveau code du projet.

Finalement, bien que nous tenions des rencontres hebdomadaires avec le client pour lui montrer l'état d'avancement des tâches, nous aurions voulu faire plus de démonstrations pour expliquer comment télécharger l'application que nous avons développée et comment lancer le programme pour la première fois. Bien que tout soit documenté, il aurait été intéressant d'en faire une démonstration visuelle.

Malgré les difficultés rencontrées et les succès que nous avons notés, nous sommes très satisfaits du travail que nous avons effectué en équipe. Nous sommes également confiants que le client sera satisfait du produit développé.

5. Apprentissage en continu (Q12)

5.1 Lacunes identifiées dans ses savoirs et savoir-faire durant le projet

Johnny Khoury:

Pour ce qui est des lacunes que j'ai identifiées durant le projet, j'ai réalisé que j'hésitais souvent à demander de l'aide à mes coéquipiers. En effet, lorsque j'étais bloqué lors du développement du code, je tentais souvent de résoudre le problème par moi-même. Bien que le travail d'équipe soit une de mes forces, je suis également une personne qui aime régler les problèmes auxquels je fais face par moi-même. Toutefois, même si je voulais

régler les problèmes de codage seul, je ne le faisais pas si cela me retardait ou bloquait d'autres membres de l'équipe. Dans le cas où ça prenait trop de temps, je demandais de l'aide à mes coéquipiers. Toutefois, j'ai réalisé que je devais demander de l'aide plus souvent si j'en ai besoin.

Alison Nemr :

Quand j'étais bloquée au début du projet sur ma tâche, je pouvais rester bloquer pendant plusieurs jours au lieu de demander de l'aide ou de travailler sur quelque chose d'autre. Je voulais vraiment finir ma tâche avant de passer à une autre tâche. Cela faisait en sorte que je trouvais que je ne contribuais pas beaucoup.

Mohamed Zakaria:

Au début du projet, lorsque j'avais fini de développer l'interface initiale, on était en attente des données du serveur afin de les afficher sur celle-ci. Il y avait donc une période où je ne pouvais pas continuer sur l'avancement de l'interface.

Alex Hua:

Une de mes lacunes que j'ai identifiées durant ce projet est qu'au début, je ne participais pas assez à aider les autres membres dans leur tâche. Après avoir complété ma tâche, j'essayais plutôt d'optimiser certains petits éléments ou je cherchais d'autres tâches plutôt que d'aller donner un coup de main aux autres membres qui étaient moins avancés.

Victor Vergeau:

J'ai constaté que prendre simultanément les rôles de gestionnaire et de programmeur peut être particulièrement exigeant, car cela implique de coordonner les efforts de l'équipe, de surveiller les délais et de résoudre les problèmes émergents tout en y participant, ce qui a créé une charge de travail élevée.

De plus, j'ai réalisé que j'avais à perdre du temps et à retarder la réalisation de certaines tâches essentielles en me laissant distraire par d'autres problèmes. Par exemple, je me suis souvent retrouvé à essayer d'améliorer une partie spécifique du code, mais en creusant plus profondément, j'ai découvert d'autres problèmes plus prioritaires qui ont pris le dessus sur l'objectif initial.

Ghazi Ben Achour:

J'ai constaté qu'il m'était difficile m'approprier un projet dans lequel il y avait déjà une base de code existante. La compréhension et la délimitation des tâches à faire ont été un défi majeur pour moi et pour l'équipe au début du projet. Par exemple, lorsque j'avais la tâche de réusiner les composantes de l'environnement, sachant que l'environnement interagissait avec d'autres composantes telles que les agents, il était difficile de se concentrer exclusivement sur cette composante.

5.2 Méthodes prises pour y remédier

Johnny Khoury:

Afin de remédier au problème que j'ai soulevé à la section 5.1, je me suis donné une période pour être en mesure de faire du débogage par moi-même. Après 2 jours de débogage seul sans trouver de solution, je me devais de demander de l'aide à mes coéquipiers pour ne pas retarder le travail. De cette façon, cela m'aidait à moins hésiter à demander de l'aide.

Alison Nemr :

J'attendais aux rencontres SCRUM pour demander aux autres membres s'ils ont besoin d'aide dans leur tâche, ce qui peut prendre des jours avant que je puisse demander.

Mohamed Zakaria:

J'ai demandé aux membres de l'équipe qui s'occupait du serveur de rendre prioritaire l'envoi de données du serveur vers le client afin que je puisse continuer à travailler sur l'interface. Ils ont tout de suite passé à l'action et cela n'a pris que quelques jours afin de terminer l'envoi des données vers l'interface. Pendant ce temps, j'avançais le rapport de mi-session.

Alex Hua:

Pour remédier à cette lacune, je pris l'initiative de demander aux autres membres où ils en étaient rendus dans leurs tâches, afin de les rejoindre et les aider à progresser en programmation en binôme.

Victor Vergeau:

Tout d'abord, nous avons décidé de tenir des réunions d'équipe plus fréquentes pour discuter de l'avancement du projet et pour nous assurer que nous nous concentrions sur les tâches prioritaires. Cela a permis de ne pas me laisser distraire par d'autres tâches qui me semblaient plus prioritaires.

Dans la même optique, j'ai commencé à écrire sur un bloc-notes ce qui doit être fait avant de commencer une tâche pour éviter de me laisser distraire par des détails mineurs.

Enfin, j'ai laissé d'autres membres de l'équipe prendre la responsabilité de gestionnaire à certaines occasions, lorsque j'avais besoin de me concentrer sur le code.

Ghazi Ben Achour:

Pour remédier à ce défi, il fallait mettre un accent particulier sur l'organisation du travail à faire et la répartition des tâches. Il fallait les délimiter d'une façon claire avec l'équipe afin de ne pas se laisser emporter dans d'autres tâches en même temps. Toutefois, grâce à un effort collectif de l'équipe et à l'aide de mes collègues, j'ai réussi à le faire, ce qui m'a permis d'avancer plus rapidement et gagner en productivité par la suite.

5.3 Identifier comment cet aspect aurait pu être amélioré

Johnny Khoury:

Pour améliorer la difficulté que j'ai rencontrée, une solution possible est de travailler en équipe avec une autre personne dans l'équipe sur une tâche. De cette façon, le travail se fait de façon plus rapide et nous pouvons régler les problèmes de codage plus rapidement. Par exemple, en étant 2 personnes sur une tâche, nous pourrons l'accomplir plus rapidement. Il faudra bien sûr s'assurer que la communication soit bonne et qu'il n'y ait pas de perte de temps pour que le travail soit bien accompli.

Alison Nemr :

Pour améliorer la méthode prise pour remédier à ma lacune, j'aurais dû envoyer un message sur notre canal Discord, qui était utilisé pour la communication entre les membres en dehors de l'école, pour savoir si quelqu'un a besoin d'aide.

Mohamed Zakaria:

Pour améliorer la difficulté que j'ai rencontré, j'aurais pu demander aux membres qui travaillaient sur le serveur de m'expliquer comment celui-ci fonctionnait afin que je puisse les aider sur le développement des fonctionnalités du serveur.

Alex Hua:

Pour améliorer la solution prise pour remédier à ma lacune, j'aurais dû prendre plus de temps à comprendre leurs tâches et ce qu'ils ont fait jusqu'à présent pour pouvoir mieux les aider ou même d'essayer de chercher plus de solutions de mon côté.

Victor Vergeau:

Pour mieux gérer mon travail de programmation et de gestion, il serait bénéfique de mieux cerner la portée des tâches dès le début du projet. Je pourrais accomplir cela en augmentant le partage de connaissances entre les membres de l'équipe, ainsi qu'en effectuant des tâches d'exploration de code pour mieux estimer les tâches. Cela permettrait également m'empêcher de tomber dans le piège du de découvrir d'autres problèmes en essayant de résoudre un problème en ayant une meilleure vue d'ensemble des priorités du projet.

Ghazi Ben Achour:

Afin de mieux gérer le travail et la difficulté des tâches, il est important d'avoir une idée claire sur les objectifs à réaliser au début du projet. La première phase du projet me semble donc être la plus importante pour bien avancer par la suite. Il me semble aussi

important de subdiviser les tâches en de plus petites sous tâches afin d'avoir des résultats et un avancement plus itératif.

6. Références utilisées

*Note: Certaines références ont également été utilisées pour notre rapport de mi-session. Nous les avons placés dans notre rapport final, car elles nous ont aussi servi pour cette remise. La date de consultation correspond à la première consultation faite, soit celle pour le rapport de mi-session lorsque cela s'applique. Lorsque les dates précises de consultation pour le rapport de mi-session n'ont pas été notées, nous avons seulement écrit le mois de la consultation.

[1] Stack Overflow. (2022) Developer Survey. [En ligne]. Disponible: https://survey.stackoverflow.co/2022/, page consultée le 26 janvier 2023.

[2] Gymnasium. Gymnasium Documentation. [En ligne]. Disponible: https://gymnasium.farama.org/content/basic_usage/, page consultée le 8 février 2023.

[3] Siimec. (2023) Processus-agile. [En ligne]. Disponible: https://www.siimec.com/presentation/notre-approche/processus-agile/, page consultée le 27 janvier 2023.

[4] FastApi. FastApi Documentation. [En ligne]. Disponible: https://fastapi.tiangolo.com/, page consultée en février 2023.

[5] Angular Material. (2010-2023) Guides. [En ligne]. Disponible: https://Material.angular.io/, page consultée en février 2023.

[6] Pydantic. Overview. [En ligne]. Disponible: https://docs.pydantic.dev/, page consultée en février 2023.

[7] Socket.io. (2023) Documentation. [En ligne]. Disponible: https://socket.io/, page consultée en février 2023.

[8] Farama. (2022) Announcing The Farama Foundation. [En ligne]. Disponible: https://farama.org/Announcing-The-Farama-Foundation, page consultée en février 2023.

[9] Polytechnique Montréal. (2021) Citer selon le style IEEE. [En ligne]. Disponible: https://guides.biblio.polymtl.ca/citer_ieee, page consultée en février 2023.

[10] G. Ben Achour, A. Hua, J. Khoury, A. Nemr, V. Vergeau, M. Zakaria, “Rapport de mi-session - Développement d’un environnement OpenAi/Gym pour simulation de réseaux électriques intelligents et soutenables”, Département de Génie informatique, Polytechnique Montréal, Montréal, Québec, Rapport de mi-session 1, 2023.

Dictionnaire

1. Interface utilisateur (section 1.1): **Terme anglais:** Front end. **Définition:** Partie visible de la page web.

2. Côté serveur/logique (section 1.1): **Terme anglais:** Backend. **Définition:** Partie non-visible à l’utilisateur, celle qui contient la logique des fonctions du code.

3. Cadriciel (section 2.5): **Terme anglais:** Framework. **Définition:** Outil qui présente des éléments ou des composantes qui peuvent être utilisés directement.

4. Banc d’essais (section 2.5): **Terme anglais:** Testbed. **Définition:** Terme qui veut dire que l’on simule notre application pour s’assurer qu’il fonctionne bien.

5. Version Beta (section 2.6): **Définition:** Version d’essai remise au client qui nous permettra de recevoir de la rétroaction avant la remise finale.

6. Source libre (section 3.1): **Terme anglais:** Open source. **Définition:** Code libre et disponible pour tout le monde.

7. Gradateur (section 3.1): **Terme anglais:** Slider. **Définition:** curseur qui permet de défiler horizontalement.

8. Bouton d'arrêt complet (section 4.1): **Terme anglais:** Bouton stop. **Définition:** bouton qui permet d'arrêter complètement la simulation.

9. Bouton de lecture (section 4.1): **Terme anglais:** Bouton play. **Définition:** bouton qui permet de commencer/partir la simulation.

10. Bouton d'attente (section 4.1): **Terme anglais:** bouton pause. **Définition:** bouton qui permet d'interrompre temporairement la simulation et de la repartir sans perdre toutes les données.