

Rapport de soutenance



Nom du projet : Raider's Architecture

Durée du projet : Février 2024 - Juin 2024

Nom du groupe : The Raider's

Chef du projet : Maël ROUSTIT

Nom des membres du groupe :

- Axel OURY
- Damien MARASSÉ
- Maël ROUSTIT
- Romain D'ANGE-BOURGUIGNON

Sommaire

I – Rappel du projet.....	2
II – Tâches à accomplir.....	3
Répartition des tâches.....	3
Prévisions faites pour le cahier des charges.....	4
Détails des tâches.....	5
Algorithme principal.....	5
Algorithmes Display.....	6
L'interface graphique.....	7
L'interface graphique.....	7
La connexions des algorithmes à l'interface.....	9
Gestionnaire de projet.....	11
Site Web.....	11
Introduction.....	11
Le Header.....	12
Les intégrations.....	12
Téléchargements.....	12
Les designs.....	13
III – Prévisions pour la prochaine soutenance.....	14
Les objectifs pour la prochaine soutenance.....	14
IV - L'intérêt du travail en groupe.....	15
V – Conclusion.....	16

I – Rappel du projet

L'objectif de ce projet de S4 est de réaliser un logiciel où l'algorithmique a une part extrêmement importante, cependant le projet reste libre.

Avant toute chose pour réaliser ce projet, nous utiliserons différents supports (Linux, Visual Code,...) qui nous permettront d'écrire l'entièreté du projet en Rust. De plus, pour avoir un suivi constant des différents avancements de chacun des membres du groupe, nous avons créé un serveur sur l'application Discord. Nous avons aussi créé un drive Google que nous complétons au fur et à mesure pour avoir un aperçu des différentes tâches que nous avons accomplies et de celles qui sont en cours de réalisation. Nous utilisons l'application Notion pour nous fixer des deadlines pour certaines parties du projet à effectuer en priorité, mais aussi pour avoir un planning simple et compréhensible pour tout le monde, pour ainsi mieux s'organiser.

Au début, nous étions partis sur un explorateur de fichiers. Celui-ci devait être composé d'une interface graphique qui aurait permis de parcourir les dossiers et les fichiers de l'ordinateur. Il devait nous permettre d'effectuer des actions, comme la création d'un nouveau fichier/dossier, la suppression, le déplacement, la recherche et plein d'autres que l'on peut retrouver sur d'autres explorateurs de fichiers existants. Nous voulions aussi implémenter une recherche par méta données, une prévisualisation de fichiers ainsi qu'un compresseur de fichiers. Malheureusement, ce projet a été plusieurs fois refusé, car il a été jugé comme étant un peu trop facile et il manquait également une IA ou de l'OCR à y implémenter.

Ensuite, nous sommes partis sur une IA permettant d'écrire une partition musicale à partir de l'écoute d'une musique. Elle devait être capable de reconnaître les instruments utilisés séparément, deviner les notes jouées et trouver le rythme de cette musique pour créer la parfaite partition. Le projet devait s'accompagner d'une interface graphique user-friendly. Cependant, ce projet nous a aussi été refusé, car il était considéré trop complexe à réaliser pour le temps que nous possédions pour faire ce projet.

Ainsi, nous sommes donc partis sur une nouvelle idée que nous nous efforcerons de vous présenter tout au long de ce rapport. "The Raiders" est heureux de vous montrer les débuts de leur nouveau projet dénommé "Raider's Architecture". Cette application permet à ses utilisateurs de créer des plans de maison en suivant un certain nombre de contraintes.

II – Tâches à accomplir

Répartition des tâches

Nous avons dû nous adapter à la validation de notre cahier des charges qui a pris beaucoup de temps. C'est pour cela que notre répartition des tâches est la suivante :

Axel :

- L'interface graphique
- La connexions des algorithmes à l'interface

Damien :

- L'implémentation et optimisation de l'algorithme principal (création du plan)

Maël :

- Le site internet
- Gestionnaires de projets dans l'application
- Gestion du projet

Romain :

- L'implémentation des algorithmes secondaires (Display dans un premier temps, et optimisation dans un second)

On ajoutera également qu'il n'y a pas uniquement une personne qui fera une tâche. Il s'agit juste du responsable de celle-ci sinon en pratique tout le monde travaillera sur un peu tout, le but étant de pouvoir s'aider mutuellement un maximum en cas de nécessité et à tout moment.

Prévisions faites pour le cahier des charges

Ci-dessous le tableau récapitulant les prévisions faites lors de la rédaction du cahier des charges (colonne « Période 2 ») et ce qui a réellement été fait (« Validation »).

Tâches	Période 2 (en %)	Validation
L'implémentation de l'algorithme principal	70%	✓
L'implémentation des algorithmes secondaires	40%	✓
Le site internet	90%	✓
Gestionnaire de projet	10%	✓
L'interface graphique	50%	✓

Malgré des pourcentages très faibles à la première soutenance, dû à un retard sur le cahier des charges, nous sommes dans les temps par rapport à ce qui a été annoncé. Nous pensons également pouvoir finir le projet dans les temps.

Détails des tâches

Algorithme principal

Pour structurer notre projet, nous avons adopté une approche basée sur les structures (**structs**), qui nous permettent de représenter la maison et ses pièces de manière organisée. Cette méthodologie offre la flexibilité nécessaire pour intégrer des paramètres supplémentaires, tels que des contraintes spécifiques, tout en développant des algorithmes efficaces pour la gestion et la manipulation de ces données.

La structure principale de notre système est la **House**, qui contient actuellement une liste de pièces appelées **rooms**. Nous envisageons d'ajouter ultérieurement des paramètres tels que le nombre d'étages de la maison (**floors**) et même l'orientation de la maison pour optimiser l'emplacement des fenêtres et éviter une exposition excessive au soleil.

Chaque pièce est représentée par la structure **Room**, qui inclut le type de la pièce (salon, chambre, cuisine, etc.) ainsi que ses dimensions. Nous prévoyons également d'intégrer des fonctionnalités telles que la gestion des connexions entre les pièces, la présence de portes et de fenêtres, ainsi que leur positionnement dans l'espace.

Du point de vue des algorithmes, nous avons déjà mis en place un système de sauvegarde fonctionnel qui peut être adapté pour répondre à différentes contraintes spécifiques. Le programme de sauvegarde convertit une instance de **House** en un fichier texte au format spécifique, où chaque ligne représente une pièce avec ses dimensions.

Lorsque l'utilisateur utilise l'interface graphique pour créer une maison, celle-ci génère également un fichier texte conforme au format attendu, qui est ensuite chargé par la fonction de chargement. Cette fonction lit le fichier ligne par ligne, créant ainsi les instances de **Room** correspondantes, puis assemble une **House** contenant la liste de ces pièces pour être utilisée dans le programme.

Actuellement, l'utilisateur peut créer une maison en entrant manuellement les dimensions de chaque pièce. Cependant, notre objectif final est de développer un algorithme capable de générer automatiquement une disposition de pièces optimale, en tenant compte de diverses contraintes. Ceci nécessitera le développement de plusieurs programmes et une recherche approfondie pour déterminer la meilleure répartition des espaces, entre autres considérations.

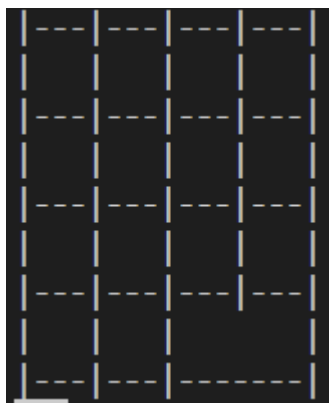
Algorithmes Display

Pour cette première soutenance, nous avons décidé de nous focaliser sur le fait d'avoir un élément complètement fini, plutôt que de s'éparpiller sur pleins de contraintes sans pour autant les réaliser à 100%. Donc nous avons entrepris de travailler complètement sur la contrainte qui est le nombre de pièces, afin d'avoir une ébauche de plan "viable".

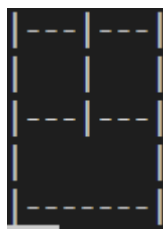
Afin de représenter de la manière la plus claire possible les pièces de notre plan, nous avons opté pour le fait de dessiner les pièces tels des carrés (à base de ']' et de '-' afin de faire les contours de ces carrés), et de les coller afin d'avoir un rendu ressemblant à un véritable plan.

La solution de facilité (qui a été implémentée au tout début de cette période) est de réaliser une maison toute en longueur, c'est-à-dire en collant toutes les pièces côte à côte dans le même sens. Le rendu donnait donc un espèce de rectangle très long, très loin d'un vrai plan de maison.

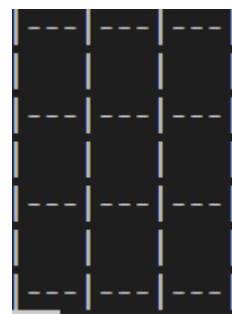
Il est donc paru évident que nous devions implémenter une méthode différente afin d'avoir un plan cohérent. Pour cela, nous avons opté pour un algorithme permettant de réaliser un plan bien plus cohérent, c'est-à-dire avec des pièces pouvant être collées de la gauche vers la droite et également du haut vers le bas. Maintenant, l'algorithme réalise des plans cohérents, avec des pièces organisées de manière réaliste. Voici quelques exemples pour imaginer ces propos qui sont probablement assez flou sans support visuel :



Plan de 14 pièces



Plan de 3 pièces



Plan de 9 pièces

L'algorithme n'est pour le moment pas modulable, c'est-à-dire qu'il ne prend pas en compte les contraintes de taille, de forme ou encore de placement. Cela viendra plus tard, dans les versions futures, et cet algorithme sera perpétuellement amélioré afin d'avoir un rendu propre et répondant à toutes les contraintes auxquelles nous avons pensé.

L'interface graphique

L'interface graphique

Pour que notre idée puisse prendre vie, nous nous devons de posséder une interface simple et compréhensible par tout le monde. Pour réaliser cette interface, nous avons décidé d'utiliser la bibliothèque slint car nous avons appris que celle-ci nous permettait de faire une multitude de choses essentielles à la réalisation de ce projet. Cependant sa prise en main ne fut pas aussi aisée qu'espéré. En effet, son installation fut extrêmement complexe (surtout pour coder l'interface sur MacOS), on a dû s'y reprendre plusieurs fois pour réussir son installation pour pouvoir être utilisé. Nous avons également trouvé que Slint était peu documenté ce qu'on veut dire par là c'est que n'avons trouvé que trop peu d'exemples pour aiguiller dans cette tâche.

On a réussi selon nous à atteindre les 50% qu'on s'était fixé. On a une interface qui possède tous les critères que nous voulions proposer aux futures utilisateurs de notre logiciel.

Raider Architecture

House

The superficy of habitation (m2)

The superficy of living room (m2)

The superficy of kitchen (m2)

The superficy of bathroom (m2)

The superficy of bedroom (m2)

The number of bedroom (0, 1, ect)

The number of bathroom (0, 1, ect)

The number of waitroom (0, 1, ect)

The number of free space (0, 1, ect)

The number of the floor (0, 1, ect)

Appartement

☐ Garden

☐ Garage

☐ Swimming pool

☐ Parental suite

☐ Terrace

☐ Open kitchen

☐ Elevator

☐ Stairs

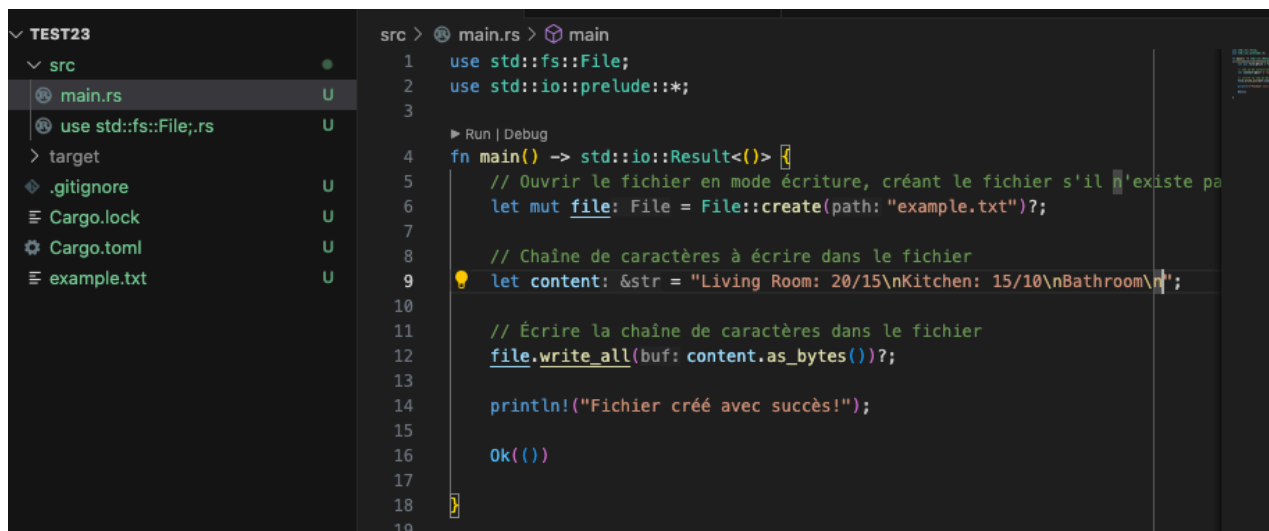
Initialization

Comme vous pouvez le constater, nous avons décidé de tout écrire en anglais pour que ce soit compréhensible pour un maximum de personnes dans le monde et que tout le monde puisse l'utiliser assez facilement. On alterne entre les "LineEdit" pour pouvoir récolter les nombres qui sont des strings et ainsi les récupérer pour les utiliser dans nos algorithmes. Les "CheckBox" quant à elles seront utilisées de la manière suivante, ces boutons renvoie "True" or "False" et de ce fait, on peut se servir de ces issues booléennes pour ajouter ou non l'un des critères que nous proposons. Cependant nous avons rencontré différents problèmes qu'à l'heure actuelle nous n'avons pas pu résoudre et qui empêchent l'interface de tourner sur Linux. Voici l'erreur qui s'affiche : "Error: Other("Could not initialize backend.\nError from Winit backend: Error initializing winit event loop: the requested operation is not supported by Winit\nNo backends configured.")"

On espère que le jour de la soutenance ce problème sera résolu pour que vous puissiez regarder l'interface s'exécuter sur Linux. Cela étant dit, nous nous sommes plus concentrés sur l'esthétique de l'interface que sur sa fonctionnalité et nous faisons tout pour régler ce problème majeur en priorité.

La connexions des algorithmes à l'interface

Concernant la connexion des algorithmes avec l'interface, nous n'avons pas pu les avancer autant qu'on voulait à cause du problème qui est apparu. Nous avons tout de même commencé les fondations, on a commencé à écrire les fonctions pour pouvoir faire le pont entre l'interface et les algorithmes. En voici quelques images des algos que nous avons commencé à faire pour créer une connexion entre les algorithmes et l'interface.

A screenshot of a Rust IDE (likely VS Code) showing a project named 'TEST23'. The left sidebar shows the file explorer with 'src' containing 'main.rs' and 'use std::fs::File;rs', and 'target' containing '.gitignore', 'Cargo.lock', 'Cargo.toml', and 'example.txt'. The main editor shows the code for 'main.rs' with the following content:

```
src > main.rs > main
1 use std::fs::File;
2 use std::io::prelude::*;
3
4 fn main() -> std::io::Result<()> {
5     // Ouvrir le fichier en mode écriture, créant le fichier s'il n'existe pas
6     let mut file: File = File::create(path: "example.txt");
7
8     // Chaîne de caractères à écrire dans le fichier
9     let content: &str = "Living Room: 20/15\nKitchen: 15/10\nBathroom\n";
10
11     // Écrire la chaîne de caractères dans le fichier
12     file.write_all(buf: content.as_bytes());
13
14     println!("Fichier créé avec succès!");
15
16     Ok(())
17
18 }
```

Cet algorithme va permettre la génération d'un document en **.txt** qui pourra être exploité pour faire fonctionner nos autres algorithmes.

```
1 use rand::Rng;
2
3 fn main() {
4     // Créez un générateur de nombres aléatoires
5     println!("Veuillez saisir un nombre :");
6
7     // Créez une nouvelle instance de `std::io::Stdin` pour lire l'entrée de l'utilisateur
8     let input = String::from(" 27 ");
9
10    // Convertir la chaîne entrée en nombre entier
11    let number: f32 = match input.trim().parse() {
12        Ok(n) => n,
13        Err(_) => {
14            println!("Ce n'est pas un nombre valide !");
15            return;
16        }
17    };
18    if number as f32 - 2.0 < 0.0
19    {
20        println!("Impossible");
21        return;
22    }
23    let mut rng = rand::thread_rng();
24
25    // Générez un nombre aléatoire entre 1 et 100
26    let random_number = rng.gen_range(number-number/2.0..=number-2.0);
27
28    // Affichez le nombre aléatoire
29    println!("Nombre aléatoire: {} ", (random_number*100.0).round()/100.0);
30 }
31 }
```

Cet algorithme qu'en a lui est une sorte de convertisseur en fait pour augmenter de manière significative l'aléatoire dans la génération de plan ainsi qu'être sûr d'avoir les paramètres qu'il faut pour faire fonctionner la génération de plan. Ces deux algorithmes n'ont pas été implémentés car ils n'ont pas été totalement terminés dû notamment à un certain manque de temps et le fameux problème que nous avons dû nous préoccuper en urgence.

On a malgré tout les obstacles rencontrés sur ce chemin, nous sommes heureux de vous dire qu'on a réussi à réaliser à plus de 50% l'interface graphique.

Gestionnaire de projet

Dans les 10% prévus pour cette soutenance, nous n'avions pas pour objectif d'implémenter en lui-même le gestionnaire de projet, mais des outils nous permettant de le faire plus naturellement lors de la prochaine période. Nous avons donc conçu un système de sauvegarde de fichiers, c'est-à-dire qu'il est possible de sauvegarder 2 choses, les données rentrées dans l'interface, comme le nombre de pièces, leur taille et les différentes autres contraintes, afin que dans le futur, on puisse recharger ces données et créer de nouveaux plans depuis celles-ci où les modifier.

Mais il est également possible de sauvegarder un plan de maison, afin de pouvoir le recharger plus tard, et tout ça, depuis le gestionnaire de projet. Le but de ce dernier sera donc de pouvoir avoir un accès direct à toutes les créations d'un individu sur notre application. Il devrait être possible de supprimer des projets pour libérer de l'espace mémoire même si le type de sauvegarde choisi, n'est pas très gourmand, nous utilisons des fichiers **.txt**. À terme quand vous ouvrirez l'interface, vous pourrez soit gérer, soit charger un projet de plan existant, ou en créer un nouveau.

Site Web

Introduction

Pour avoir l'entièreté de la 3ème et dernière période de travail consacrée au projet en lui-même, Nous avons décidé de faire l'entièreté du site web sur la 2nd période. Le cahier des charges indique 90% car il n'est pas 100% fini, Nous ajouterons des choses pour le rendu final.

Le site est disponible à ce lien : <https://epita-the-raiders.github.io/architecture/>

Le Header

La première chose qui a été faite sur le site est la création d'un "header". Déjà présent lors de la soutenance précédente avec 1 seul bouton qui permettait simplement de passer de index.html à index_en.html, nous avons ajouté 5 boutons pour que la navigation entre les pages soit plus fluide. Nous avons donc ajouté 4 pages et leurs versions anglaises pour ne pas tout avoir sur la même page. Pour que le switch de langue soit plus simple nous avons parser le lien pour en déduire est-ce qu'il contient "_en" ou non. S'il contient "_en", nous appelons la page du même nom sans "_en", sinon nous passons la page du même nom en ajoutant "_en". De même pour le passage d'une page à une autre, si on est sur une page anglais, l'appuie sur un bouton de redirection appellera la page anglaise et de même pour les pages françaises.

Les intégrations

Pour plus de réalisme et de fonctionnalités, nous avons ajouté un module pour parser et afficher du markdown directement sur le site. J'ai ensuite créé un "objet" pop-up pour afficher le markdown, qui a l'appuis sur le bouton d'ouverture, ouvre une pop-up avec le contenu demandé. Vous pouvez la fermer via un clique à côté de cette dernière, l'appuie de la touche "echap" ou encore un croix placée en haut à droite de la pop-up. J'ai ensuite ré-utilisé la pop-up pour les rapport, produit en pdf j'ai utilisé un `<iframe>` pour les afficher dans ma pop-up.

Téléchargements

Dans la mesure où nous voulions proposer 2 versions du projet, il nous fallait au moins 2 boutons, un pour Linux et l'autre pour Windows. Pour faire ça, nous avons créé une sorte de "mise en avant", quand un des deux boutons est mis en avant, il est grossi comparé à l'autre qui rapetit, le logo correspondant apparaît et une animation quand vous gardez votre souris dessus a été appliquée. Nous avons également ajouté la mise en avant automatique en fonction de votre système d'exploitation, en lisant les informations du navigateur, cela deduit si vous êtes sur Linux ou Windows et met en avant la version correspondante.

Les designs

Nous avons fait un gros travail de design sur le site pour qu'il soit le plus lisible possible et agréable à regarder. J'ai tenté d'ajouter des animations sur les boutons, notamment ceux du header, les boutons GitHub, etc. Pour cela je me suis rendu sur un site web qui proposé des boutons en HTML/CSS et nous les avons modifiés pour qu'ils collent à notre site. Nous avons choisi un couleur bleu pas trop flash qui permet de faire un mix entre un thème clair et un thème sombre

III – Prévisions pour la prochaine soutenance

Les objectifs pour la prochaine soutenance

Tâches	Période 2 (en %)
L'implémentation de l'algorithme principal	100%
L'implémentation des algorithmes secondaires	100%
Le site internet	100%
Gestionnaire de projet	100%
L'interface graphique	100%

L'objectif principal pour la période à venir est bien évidemment de finir ce projet à 100%, notamment grâce à l'optimisation et la complexification des algorithmes nous permettant d'arriver au résultat final. L'interface actuelle sera finalisée avec un gestionnaire de projet, les algorithmes seront opérationnels, et l'application pourra fonctionner sur linux et windows.

Au vu de notre avancement actuel, cet objectif nous semble réalisable et atteignable, et nous allons bien sûr tout faire pour que ça soit le cas.

IV - L'intérêt du travail en groupe

Notre projet "Raider's Architecture" va nous apporter de grandes choses en termes de valeurs.

Nous allons devoir communiquer, nous allons devoir discuter, nous allons devoir argumenter comme une véritable bande d'amis cherchant à choisir la meilleure série à regarder. Chacun donne son avis et, ensemble, on décide ce qui fonctionne le mieux, ce qui est le plus intéressant à rajouter, quelle méthode semble être la meilleure.... Cela fait ressortir l'esprit d'équipe, nous permettant d'être soudés et de travailler ensemble dans la même direction, afin d'atteindre le but commun qui est de réaliser toutes nos tâches. Les moments difficiles sont comme des défis de jeux vidéo : on n'abandonne pas, on s'entraide afin de pouvoir les surpasser.

Bien sûr, tout n'est pas tout rose, il nous arrive de nous énerver à propos de certains détails, il nous arrive également de se tromper sur la direction à prendre par rapport à certains programmes algorithmiques. Quand une idée ne marche pas, on cherche à s'adapter, par exemple en expérimentant une autre approche. On apprend tous ensemble, et chacune de nos erreurs sont des petits coups de pouce vers le succès.

En fin de compte, notre projet "Raider's Architecture" n'est pas juste une simple machine à réaliser des plans de construction. C'est bien plus que cela : c'est une création d'équipe, qui montre le fruit d'une lourde réflexion sur un problème plus que complexe où chacun a apporté et apporte encore ses idées et ses compétences. C'est cela le véritable travail d'équipe : non pas une simple collaboration, mais une expérience qui nous fera à coup sûr grandir.

Et puis pour rappel, notre groupe se nomme "The Raiders", en référence au jeu Tomb Raider, parce que nous aimons tous les jeux vidéo et que la recherche d'un sujet à la fois intéressant et challengeant algorithmiquement a été une véritable aventure. Tout comme ce projet qui sera notre incroyable épopée vers quelque chose qui nous rendra tous fiers.

V – Conclusion

Depuis notre soutenance précédente, nous avons réalisé pas mal de tâches et notre projet s'oriente désormais dans un chemin qui devrait nous permettre d'offrir un rendu propre et répondant à vos attentes, mais également aux nôtres. Nous avons travaillé d'arrache-pied afin de rattraper le retard que nous avons accumulé lors de notre précédente soutenance, et sommes désormais en bonne voie pour atteindre notre objectif final. Nous espérons que le fruit du travail que nous avons fourni jusqu'à maintenant vous a permis de partager notre engouement pour ce projet, et que cela vous aura convaincu. Merci beaucoup d'avoir pris le temps de lire ce rapport, et nous espérons de tout coeur que celui-ci vous aura plu. Le groupe "The Raiders" vous souhaite une excellente journée et vous dit à très bientôt pour notre dernière soutenance, où nous aurons l'occasion de vous présenter notre projet final.