Projet P2I2 : WORDLE + Solveur

Christophe HOYAU, Ismaël FALL, Jérémy DA SILVA
Responsable du module : Olivier FESTOR

Mars 2022 - Mai 2022

Table des matières

Chapitre 1

Introduction

Objectif du projet

L'objectif de ce projet est de concevoir une application sur le modèle du jeu WORDLE dans un premier temps. Cette dernière s'implémentera à l'aide de Python, Flask, HTML/CSS et des bases de données, étudiés dans le cadre du module CS54 durant le premier semestre. Il s'agira, par la suite, d'implémenter à l'aide de nos connaissances en C et en structures de données un solveur capable, de lui-même, de trouver le mot cible.

Chapitre 2

État de l'art

2.1 WORDLE

2.1.1 Qu'est ce que c'est

WORDLE est un jeu dans lequel le joueur a des lettres à sa disposition. Son but est de trouver un mot déterminé à l'avance en essayant différentes tentatives. Ces dernières sont limitées et apparaissent successivement dans une grille qui indique combien d'essais il reste au joueur.

Celle-ci expose également, pour chaque lettre de chaque mot si elle est, ou non, présente dans le mot à trouver et, le cas échéant, si elle se trouve au bon emplacement. Le joueur doit ainsi déduire des différentes indications qui lui sont données, quel est le mot recherché. S'il parvient à l'écrire avant d'être à court de tentatives, il a gagné. Dans le cas contraire, c'est perdu.

2.1.2 Pourquoi un tel succès

Le concept connait un succès sans précédent, l'exercice mental qui en résulte présente un côté addictif si bien que des émissions télévisées basées sur ce divertissement sont diffusées à travers le monde entier. On peut prendre l'exemple de Motus en France, qui a connu un succès monumental pendant près de 30 ans de diffusion. En effet, même si une seule personne peut tenter sa chance à chaque essai, chacun peut essayer de déduire le mot mystère à partir des précédentes tentatives.

2.1.3 Fonctionnement et algorithmes

Le principe du jeu se base sur une boucle de tentatives. Une seule d'entre elles se caractérise de la manière suivante. Un mot est reçu en entrée par l'algorithme, celui-ci effectue les tests suivants pour chaque lettre qu'il contient, de gauche à droite :

- Si la lettre appartient au mot et est à la bonne place : Colorier la lettre en rouge
- Si la lettre appartient au mot et n'est pas à la bonne place :
  - Si le mot cible contient n occurrences de cette lettre et que n occurrences de celle-ci sont déjà colorées dans le mot tenté : ne rien faire
  - Sinon : colorier la lettre en jaune
- Si la lettre n'appartient pas du tout au mot : ne rien faire

2.2 Solveur

2.2.1 Fonctionnement et algorithmes

En ce qui concerne le solveur, la manière la plus classique consiste à partir d'une liste de mots possible et de la restreindre en fonction des indices donnés par le jeu lorsqu'il corrige les tentatives du joueur. Néanmoins, il existe de multiples façons de procéder.

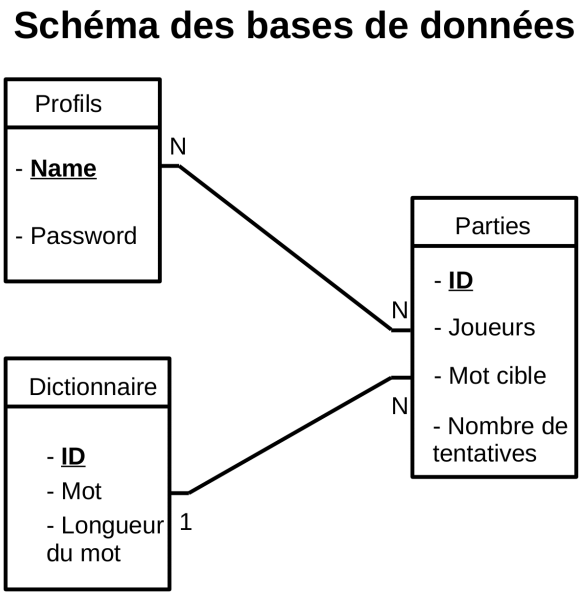
Les plus optimisées d'entre elles vont jusqu'à utiliser un deuxième 'mini' solveur qui testera en externe les lettres restantes lorsque la majorité des lettres d'un mot ont été trouvées mais qu'il reste malgré tout beaucoup de possibilité (par exemple les mots possédant plusieurs conjugaisons)..

L'efficacité s'accentue aussi avec l'analyse de fréquence d'apparition des lettres dans les mots de la langue française.

Chapitre 3

Conception et implémentation de l'application

3.1 Bases de données



Nous avons choisi la configuration suivante pour la base de données de notre application. Dans la table Profils nous stockons les pseudos et mot de passe correspondant pour un compte donné.

La table Parties fait office d'historique, on y retrouve le numéro ID par lequel une partie est identifiée, le pseudo du joueur qui l'a effectuée sous la forme d'une chaîne de caractères, le mot qui y était recherché, également sous la forme d'une chaîne de caractères, et enfin la grille de tentatives effectuées afin d'y parvenir. Ce dernier se caractérise, dans la base de données, par un une chaîne de caractères d'un format particulier obtenu grâce à notre fonction get_tentative.

Enfin, le dictionnaire composé de quelques 300 000 mots français constitue la dernière table. S'y trouvent chaque mot, l'ID lui correspondant et enfin sa longueur.

3.2 Serveur Web

Nous avons développé l'application Wordle sous Python avec l'aide du framework Flask. Nous nous sommes également servis de la librarie Flask-Session pour gérer les sessions utilisateurs.

3.2.1 Page d'accueil, d'inscription, de connexion et de déconnexion

La route /, nous amène sur la page d'accueil, à partir de celle-ci, on peut accéder à la page d'inscription (route : */signup*). On peut alors entrer un pseudo et un mot de passe, qui constitueront un nouveau compte à condition que celui-ci n'existe pas déjà. On peut également se rendre sur la page de connexion (route : */login*), cette étape sera nécéssaire avant de jouer, afin notamment de pouvoir enregistrer les performances dans l'historique. Il est, par la suite, possible de se déconnecter (route : */logout*) via le bouton du même nom.

3.2.2 Page de choix de la difficulté et jeu

Dès lors que nous sommes connectés, il est alors possible de lancer une partie via l'option intitulée "jouer". On est alors redirigé vers la page de sélection de la difficulté (route : */difficulte*) qui propose alors, via un système de cases à cocher, de donner un intervalle entre 5 et 11 qui définira les longueurs min et max des mots qui pourront, par la suite, être proposés.

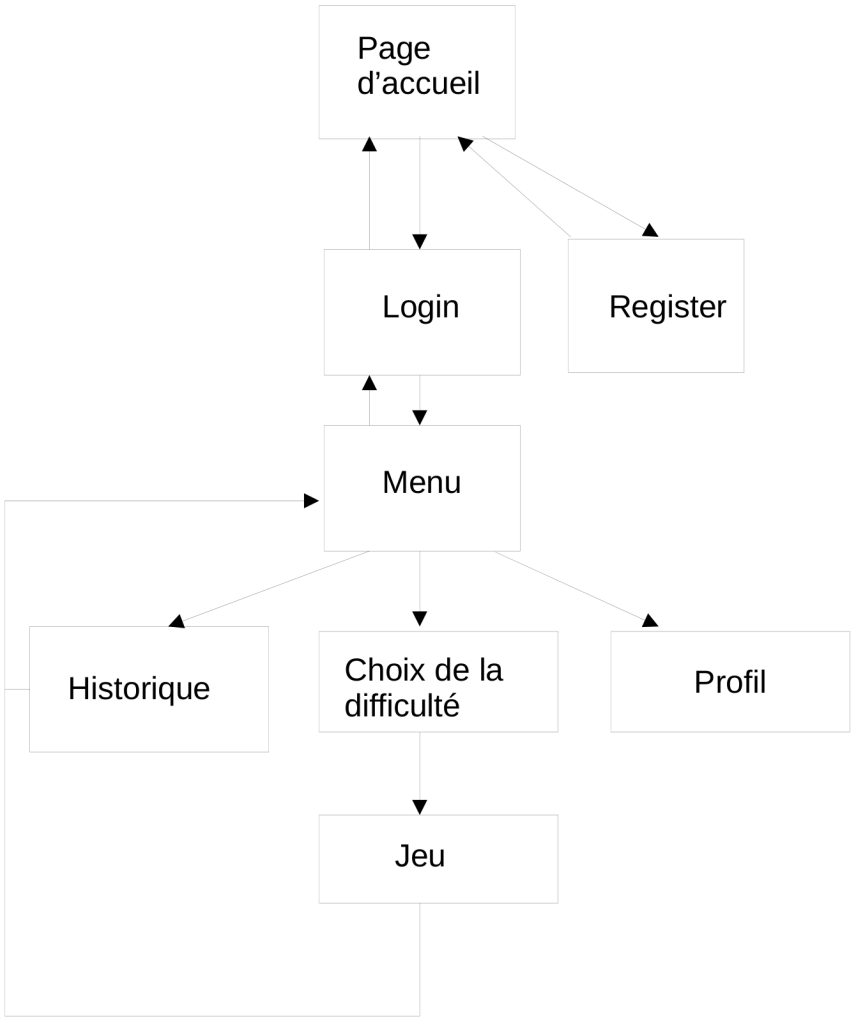
Cet intervalle ayant été soumis au serveur, le jeu, ayant été cherché un mot respectueux des conditions pré-établies, se lance alors, la grille générée est alors de la taille correspondante. Cette page (route : */jeu*), est de nouveau appellée à chaque tentative, que celle-ci soit fructueuse (mot écrit existant bien dans le

dictionnaire) ou non. Dans le premier cas, alors le mot corrigé s'affiche avec les couleurs en conséquence et le prochain mot à soumettre s'affichera en dessous dans la grille. Cette page est particulière puisque à chaque appel d'elle-même, on lui soumet en tant que requête entrante de type 'POST', l'intégralité des précédentes tentatives. Ces dernières sont soumises sous la forme d'un tableau remplis soit de "." lorsqu'aucun caractère n'est présent à cet endroit là du tableau, soit de la lettre qui doit alors s'y trouver.

3.2.3 Page de profil et d'historique

Il est également possible, depuis le menu appraissant sur la partie supérieure de l'écran, d'accéder à la page de profile (route : */profile*) qui nous indique notre nom de compte mais qui nous permet également de nous rediriger vers la page de déconnexion ou bien la page d'historique.

Cette dernière (route : */historique*) permet de visualiser nos précédentes parties sous forme de grilles représentant très exactement la partie qui a été jouée. L'intégralité de notre historique y est donc accessible avec possibilité de faire dérouler la page. Les parties y sont classées de la plus récente à la plus ancienne.



3.3 Algorithmes de traitement

3.3.1 Introduction

Afin d'effectuer les opérations nécéssaires notamment en ce qui concerne la correction des mots, des algorithmes plus ou moins complexes ont du être créés puis implémentés par nos soins .

3.3.2 get_where()

La fonction get_where constitue une composante essentielle dans l'opération de correction d'un mot. En effet, cet algorithme très basique prend en argument une chaîne de caractères et retourne la liste des différentes lettres qui la composent (sans doublons bien évidemment). En outre, elle renvoie également la liste des indices ou se trouvent celles-ci dans la chaîne de caractères soumise en entrée.

3.3.3 valid_where()

La fonction valid_where prend, quant à elle, en argument deux chaînes de caractères : une cible et un essai. Il va alors s'agir, pour le programme, de renvoyer une liste d'entiers correspondants aux places des différentes lettres du mot essai dans le mot cible. Ces valeurs sont soit :

— 2 : La lettre considérée occupe la même place dans le mot cible et dans le mot essai

— 1 : La lettre considérée est bien présente dans le mot cible mais n'y occupe pas la même place que dans le mot essai

— 0 : La lettre considérée ne se trouve pas dans le mot cible

Dans le cas où les deux mots soumis à l'algorithme sont de tailles différentes, celui-ci renvoie la chaîne de caractères suivante : "Le mot n'est pas de la bonne longueur". On peut noter qu'afin d'effectuer cette comparaison des lettres des deux mots, l'algorithme utilise la fonction get_where vue précédemment.

3.3.4 Difficulté

La fonction de difficulté, associée à la route flask "/difficulte", se charge de faire choisir au joueur deux valeurs qui feront office de bornes. C'est entre l'intervalle correspondant ces dernières que l'algorithme se chargera de choisir alétoirement une valeur qui fera alors office de difficulté. Il se chargera par la suite d'ouvrir la base de donnée dictionnaire, d'y analyser chaque élément, et de ne retenir dans une liste que les mots correspondant à la taille précédemment établie. Il s'agira alors de tirer aléatoirement un mot parmi ceux-ci. Ce mot sera alors le mot cible utilisé durant la partie.

3.3.5 Jeu

L'algorithme de la fonction jeu sert à la manipulation des tableaux. On génère premièrement une grille de la taille du mot cible. Lorsque le joueur entre un mot, celui-ci est traité. Par la suite, la fonction se sert de la variable 'tentatives' associée à la session en cours d'utilisiation (ce qui permet de garder

en mémoire les tentatives même si l'on quitte la page). En fonction du nombre d'éléments (chaînes de caractères) de cette liste 'tentatives', le fichier va générer une liste pour chacune de ces chaînes. Le résultat (pour chaque chaîne) est alors celui de valid_where avec des 0, des 1 et des 2. Nous avons alors 2 listes que nous envoyons à jeu1.html. À partir d'ici, nous allons afficher successivement les mots tentés (de notre première liste) avec le bon code couleur (en se basant donc sur la deuxième liste). Au moment où le mot est entré par l'utilisateur, le programme se connecte à la base de données dictionnaire et vérifiie que le mot donné en fait bien partie.

Chapitre 4

Conception et implémentation du solveur

4.1 Présentation détaillée des structures de données manipulées

Les structures de données utilisées dans le dictionnaires sont les suivantes :

- Structure ‘element’ dont les composantes sont une chaîne de caractères et « next », un pointeur vers un autre ‘element’
- Structure de listes chaînées contenant une composante « head » étant un pointeur sur un élément du type ‘element’
- Structure ‘wordle_element’ dotée de trois composantes :
  - Un caractère représentant une lettre considérée
  - Un entier représentant l’état de la lettre correspondante par rapport au mot cible (0, 1 ou 2)
  - Un pointeur « next » vers un autre ‘worlde_element ‘
- Structure ‘wordle_mot’ dotée de 2 composantes :
  - Un pointeur « next » vers un autre ‘worlde_mot’
  - Un pointeur « head » vers un ‘worle_element’
- Structure ‘wordle_tent’ composée d’un élément « head » qui est un pointeur vers un ‘wordle_mot’

4.2 Algorithmes de traitement

4.2.1 Corps du solveur

La fonction même du solveur qui va ensuite employer les algorithmes suivants, effectue elle-même des tests et des comparaisons dont nous allons parler premièrement. Il est intéréssant de noter que pour des besoins d'optimisations, nous choisissons comme premier mot à essayer les mots suivants pour des mots de 5 à 11 lettres :
"saine","nieras","taniser","notaires","relations","soufraient","adulterions". Ces mots ne sont pas choisis au hasard, en effet, ils contiennent chacun des lettres distinctes mais aussi et surtout les lettres apparraissant le plus dans les mots de la langue française en moyenne. Ils permettent donc d'éliminer un maximum de mots de la liste des mots plausibles.

Lorsque ce mot a été essayé, et ce sera le cas pour tous les autres mots, l'algorithme reçoit en retour la combinaison des places (0, 1 ou 2) renvoyée par l'utilisateur. Il place ces données dans un wordle_mot qui contient alors à la fois les lettres du mot ainsi que leurs états par rapport au mot cible (stockés dans des wordles_elements). Dès lors, il analyse lettre par lettre le mot tenté et pour chacune des informations récoltées sur le placement des lettres, il adapte le dictionnaire de mot possibles en conséquence :

- Si la composante place du wordle_element vaut 2 : on exécute get_word_with_here()
- Sinon si composante place du wordle_element vaut 1 : on exécute get_word_with_not_here()
- Sinon (la composante place du wordle_element vaut forcément 0) : on exécute get_word_without()

4.2.2 get_word_with_here

Il va, dans ce premier cas, s'agir de faire une boucle permettant d'analyser chaque mot du dictionnaire actuel. En parallèle de cela, l'algorithme va ajouter chacun des mots toujours en lice à un tout nouveau dictionnaire qui sera alors retourné. Demeurent 'mots candidats' les mots qui contiennent une occurence de la lettre considérée positionnée à une certaine position bien définie (le mot peut bien sûr contenir plus d'une occurence de la lettre). C'est alors cette condition qui est testée tout au long de la boucle.

4.2.3 get_word_with_not_here

Ici, la fonction va parcourir le dictionnaire actuel de la même manière que vu précédemment. Durant ce processus, l'algorithme ajoutera alors identiquement à ce qui précède chacun des mots toujours en lice à un tout nouveau dictionnaire qui sera alors retourné. En revanche, cette fois les mots à ajouter seront ceux qui possède au moins une occurence de la lettre considérée mais qui ne se situe pas à une certaine place bien définie.

4.2.4 get_word_without

Dans ce dernier cas, il va alors toujours s'agir de faire une boucle permettant d'analyser chaque mot du dictionnaire actuel. Cependant, ici, on ne va choisir de garder que les mots qui ne possèdent pas une seule occurence de la lettre considérée. Finalement, comme les deux précédents algorithmes, celui-ci va se charger d'ajouter chacun des mots toujours en lice à un tout nouveau dictionnaire qui sera alors retourné.

Dans ces 3 cas, l'algorithme devrait normalement s'éxecuter plus ou moins vite en fonction de la taille d'où l'importance de choisir un mot stratégique en tant que premier essai. Cela devrait en outre favoriser grandement la probabilité de réussite du solveur.

Chapitre 5

Tests et performances

5.1 Tests du solveur

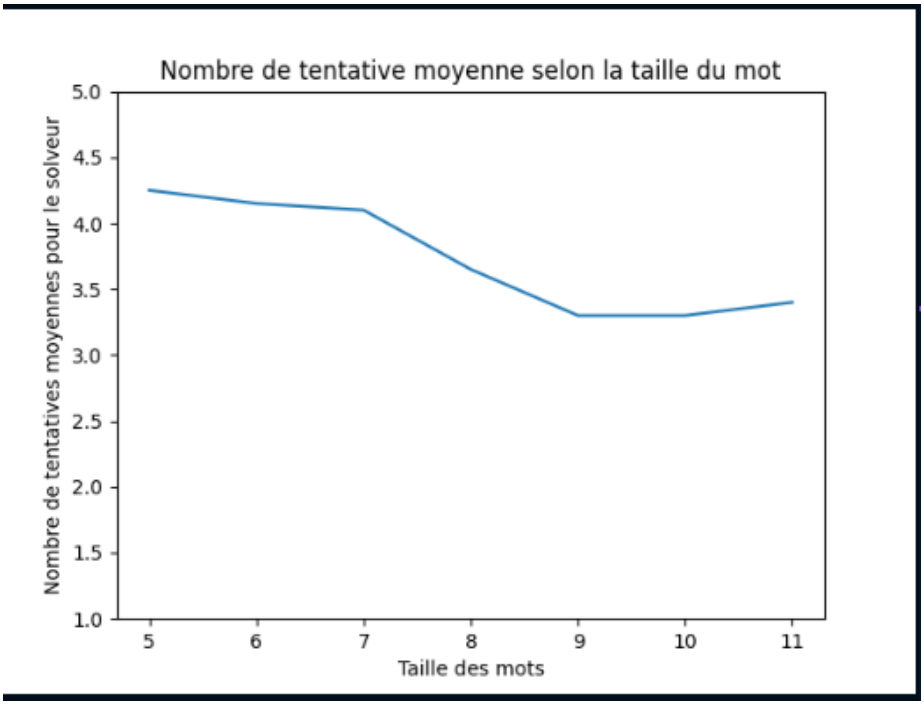
5.1.1 Introduction

Les tests de chacunes des fonctions ont été réalisés en se basant sur le modèle RIGHT BICEP. Nos mesures sont représentées avec le module pyplot de Python. Nous nous sommes ici, interessés à deux principaux critères :

— Le nombre moyen d'essais nécéssaire au solveur avant d'obtenir le résultat (l'obtention du résultat est quasi-systématique, nous répertorions à peine 1% d'échec.

— Le temps mis pour executer les algorithmes qui précédent pour chaque mot rentré.

5.1.2 Nombre moyen d'essais nécéssaires au succès

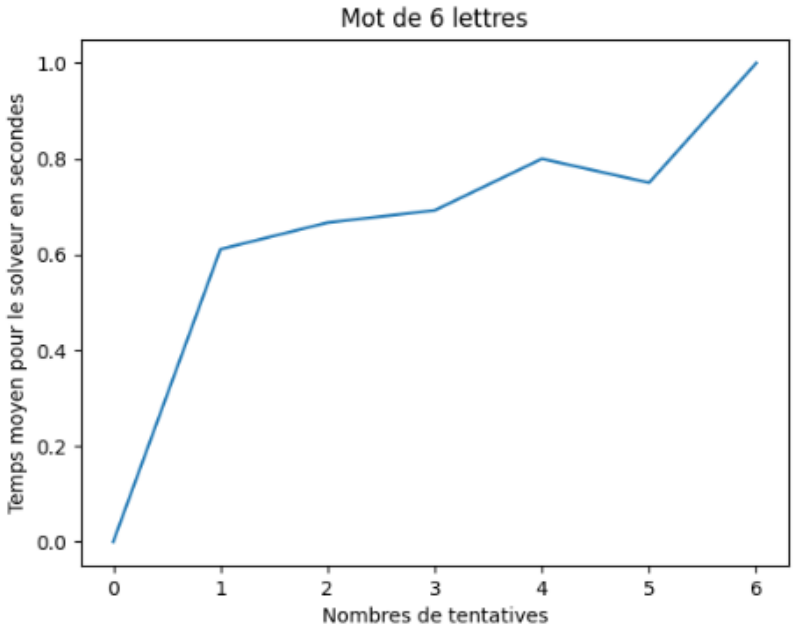


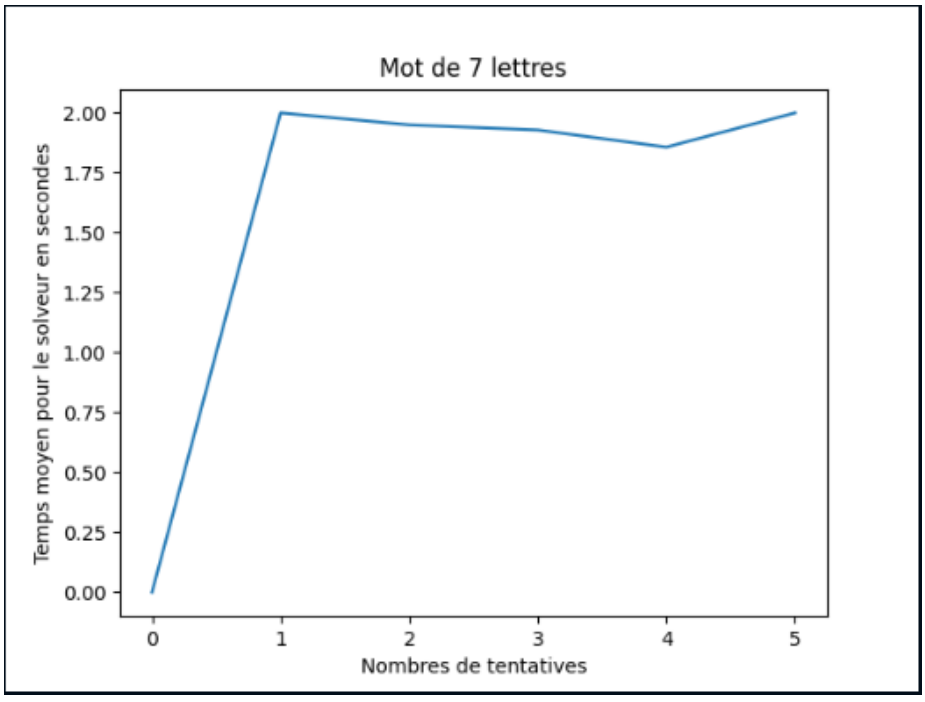
Il est intéréssant de remarquer que le nombre moyen de tentatives effectuées avant d'obtenir le mot tant convoité décroît presque uniformément avec la taille du mot. Intuitivement, nous aurions pu prévoir le contraire en se disant qu'un mot de courte taille devrait mettre moins de temps à être trouvé qu'un grand mot.

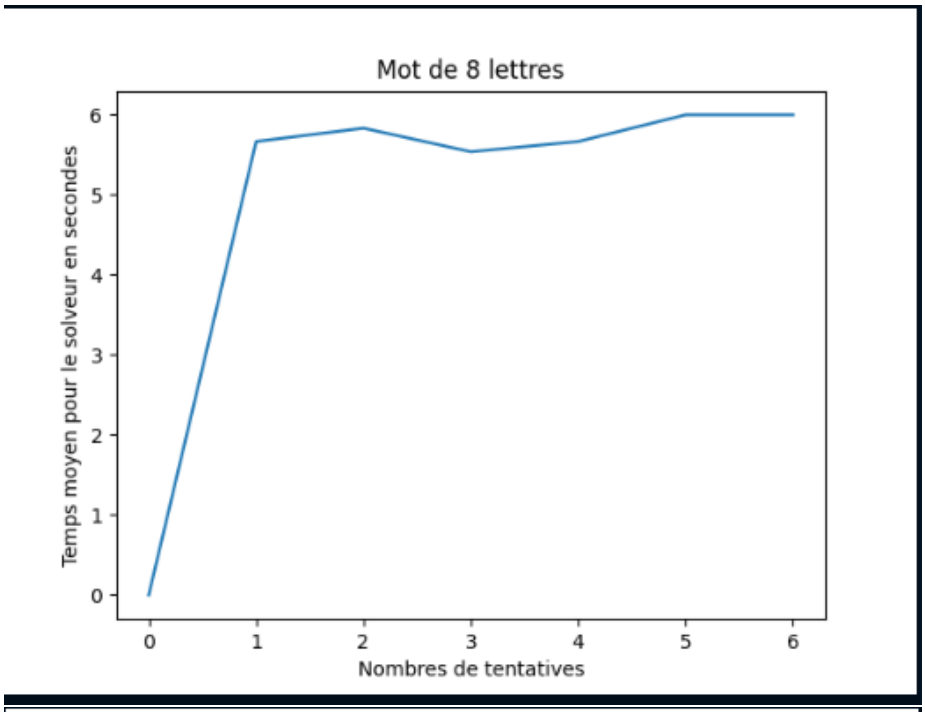
Cependant, en prenant du recul sur nos résultats, ainsi que sur la logique des algorithmes que nous utilisons, il est compréhensible de recevoir de telles données. En effet, plus le mot recherché est grand et plus le nombre de lettres testées à chaque tentative est important. Ainsi, dès le premier essai, nous obtenons d'ores et déjà beaucoup plus de contraintes sur le mot cible et donc d'indices pour la liste de mot plausibles.

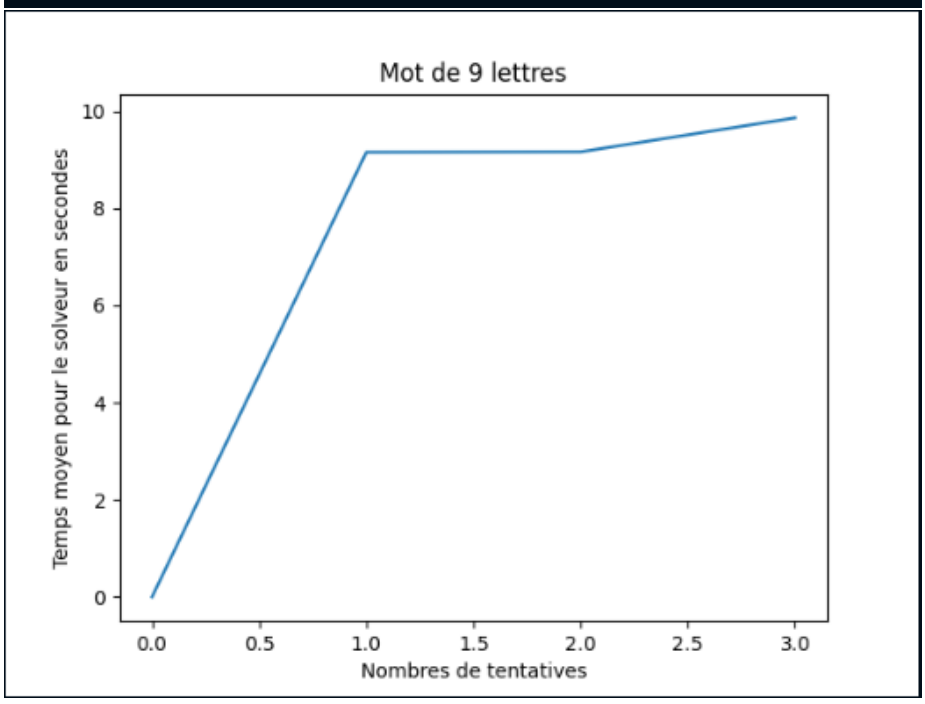
Cette dernière s'amoindrit alors grandement. On note en effet une moyenne de 4,25 essais pour des mots de 5 lettres contre 3,25 pour des mots de 9 à 11 lettres (il est également intéréssant de noter qu'à partir d'un certaine taille, la moyenne d'essais semble devenir constante).

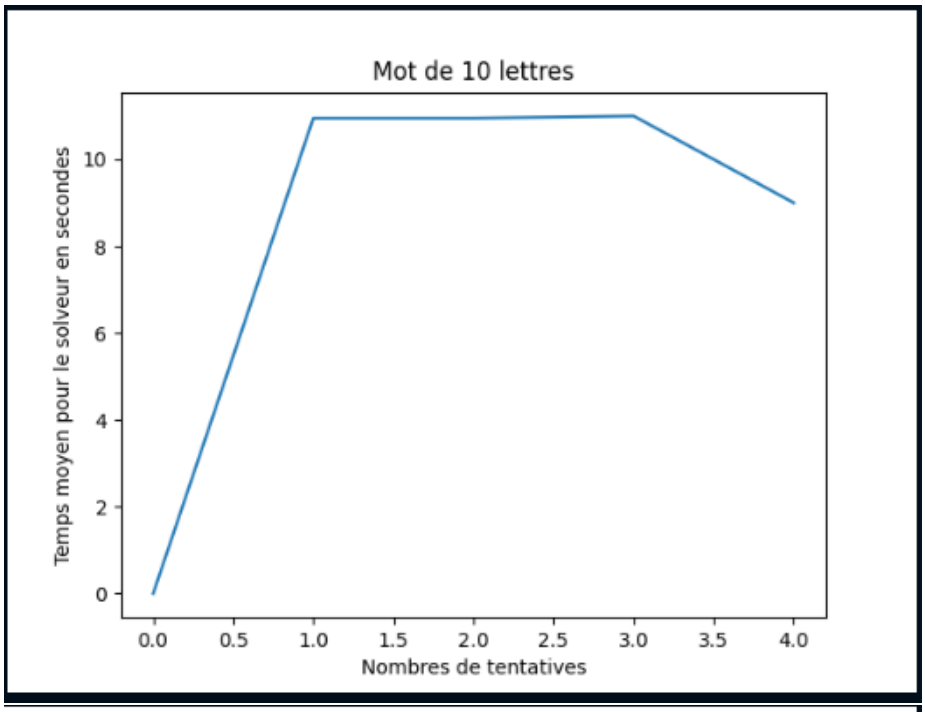
5.1.3 Temps nécéssaire au renvoi du prochain mot

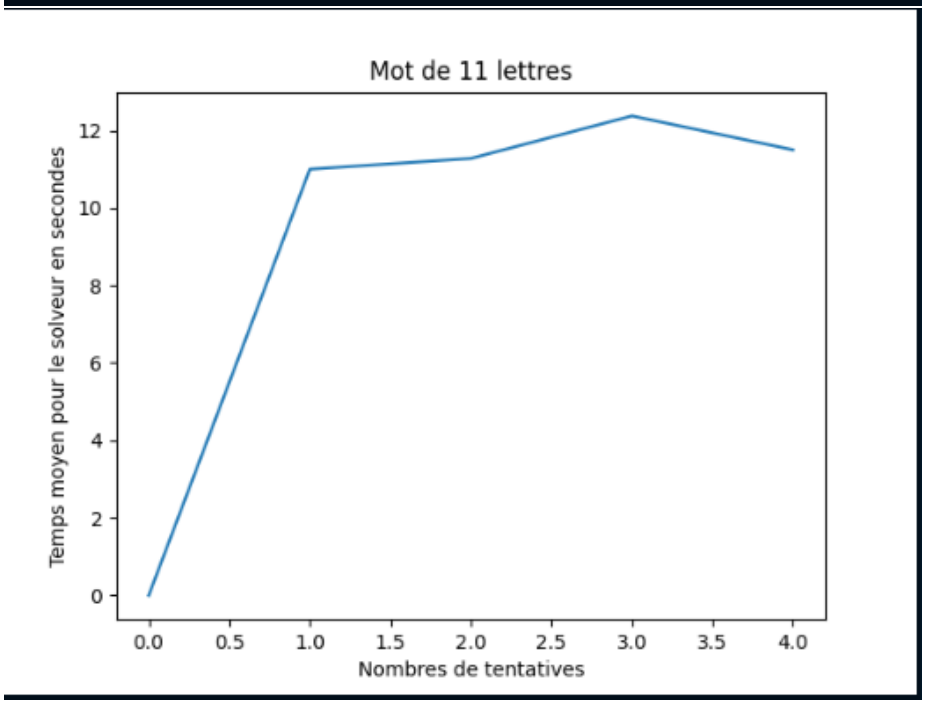





Mot de 8 lettres
Temps moyen pour le solveur en secondes
0
1
2
3
4
5
6
Nombres de tentatives


Mot de 9 lettres
Temps moyen pour le solveur en secondes
0
2
4
6
8
10
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Nombres de tentatives


Mot de 10 lettres
Temps moyen pour le solveur en secondes
0
2
4
6
8
10
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Nombres de tentatives


Mot de 11 lettres
Temps moyen pour le solveur en secondes
0
2
4
6
8
10
12
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Nombres de tentatives

5.2 Analyse en complexité

D'après les résultats précédents, nous pouvons effectuer l'analyse en complexité temporelle. Nous constatons une augmentation significative de la durée d'exécution des algorithmes lorsque la taille des mots considérés augmente. Ceci s'explique assez simplement par le fait que l'analyse lettre par lettre d'un mot prend forcément plus de temps pour un mot de n+1 lettres que pour un mot de n lettres.

En outre, il faut ensuite multiplier cette différence temporelle par le nombre de mot à traiter pour obtenir un temps représentatif de l'écart temporel entre une taille et une autre.

Ainsi, en nous basant sur les graphiques précédennt, si nous considérons les valeurs temporelles moyennes suivantes :

— 0,8 secondes pour des mots de 6 lettres
— 2 secondes pour des mots de 7 lettres
— 6 secondes pour des mots de 8 lettres
— 9 secondes pour des mots de 9 lettres
— 10 secondes pour des mots de 10 lettres
— 11 secondes pour des mots de 11 lettres

Il est alors possible de constater qu'avec l'augmentation de la taille, la durée d'exécution paraît augmenter de façon exponentielle. Cependant, aux alentours d'une taille de 9, cette durée paraît redevenir linéaire.

Ainsi, tout comme précédemment avec les moyennes, l'écart temporel parait ici devenir constant entre les tailles à partir de 9. Nous pourrions alors prévoir que pour des tailles supérieures, la durée d'exécution garderait une complexité linéaire ou deviendrait même constante.

Ce phénomène peut notamment s'expliquer par le fait que le programme crée un nouveau dictionnaire à chaque application de l'algorithme. Ainsi, cette opération pourrait avoir plus d'impact que la taille des mots eux-mêmes, passée une certaine taille...

Chapitre 6

Gestion de projet

6.1 Membres de l'équipe projet

L'équipe projet se compose de 4 membres :

Da Silva Jérémy
Hoyau Christophe
Ismaël Fall

Le chef de projet est Da Silva Jérémy

6.2 Organisation du projet

6.2.1 Organisation immatérielle

L'équipe projet s'est réunie en présentiel à l'école ou par vision-conférence durant les vacances de Pâques. La communication s'est faites en parallèle à travers un système de messagerie instantanée dans lequel un groupe à été créé. Ce groupe était constitué de l'ensemble des membres de l'équipe projet.

6.2.2 Organisation matérielle

Afin de remplir la mission qui nous avait été confiée, nous avons utilisé différents outils mis à notre disposition :

Messenger : Afin de travailler sur le projet de chez nous.

Discord : Afin d'organiser toutes les réunions en dehors de l'école.

VisualStudioCode, flask, Python, C : afin de développer le code du projet.

OverLeaf : Afin de réaliser le rapport du projet ainsi que différents document de gestion de projet.

Gitlab de TELECOM Nancy : Afin de réaliser le rapport du projet ainsi que différents documents de l'équipe projet.

Excel : Afin de tracer le diagramme de Gantt.

License Microsoft Office fournie par l'Université de Lorraine : afin de réaliser une prise de notes lors des réunions.

6.3 Analyse du projet

6.3.1 Matrice SWOT

Cette matrice SWOT concerne essentiellement l'organisation autour du projet et des membres de l'équipe projet et le fond du projet.

Strengths	**Weaknesses**
Bonne cohésion dans l'équipe Bonne expérience du jeu : Pratique régulière du jeu TUSMO identique à WORDLE	Un membre en moins
Opportunities	**Threats**
Aides pour le solveur et idées de features pour le jeu trouvables sur internet	Cumul du projet et des partiels sur les deux dernières semaines

Table 1 - Matrice SWOT

Pour gérer les menances :
-Il faut s'y prendre à l'avance.

Pour gérer les faiblesses :
-Travailler plus ensemble.

Pour profiter de nos opprtunités :
-On s'inspire des solveurs que l'on peut trouver sur internet.

Pour utiliser nos forces :
-L'implémantation des algorithmes est assez simple pour nous, puisque l'on connait le fonctionnement du jeu.

6.3.2 Matrice RACI

Le séquencement projet et cette matrice RACI correspondent à la partie application du projet.

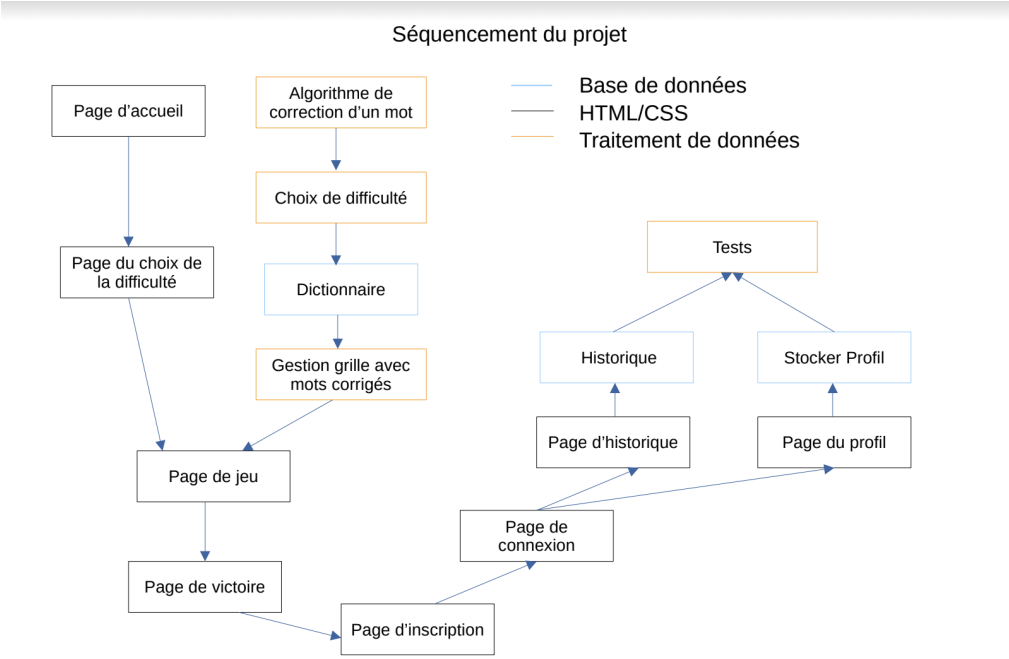


Table 2 - Séquencement projet Application

Lot de travail	**Jérémy**	**Ismaël**	**Christophe**
Page d'accueil	R		RA
Page du choix de difficulté		RA	
Page de jeu	R	R	RA
Page de victoire			RA
Page d'inscription	RA		
Page d'historique	R	R	RA
Page du profil	R	R	RA
Gestion grille avec mots corrigés	R	RA	R
Algorithme de correction d'un mot	RA		
Choix de difficulté		RA	
Tests	RA	R	R
Dictionnaire	RA		
Historique	R	RA	
Profil	R	R	RA

Table 3 - Matrice RACI Application

Le séquencement projet et cette matrice RACI correspondent à la partie solveur du projet.

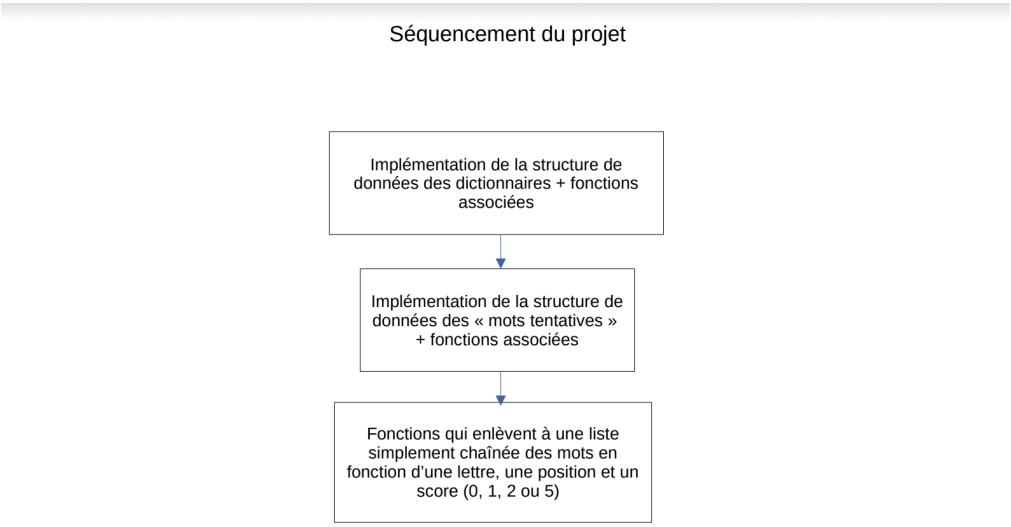


Table 4 - Séquencement projet Application

Lot de travail	**Jérémy**	**Ismaël**	**Christophe**
Implémantation de la structure de données des dictionnaires + fonctions associées	R	RA	R
Implémentation de la structure données des "mots tentatives" + fonctions associées	R	R	RA
Fonctions qui enlèvent de la structure de données des mots en fonction d'une lettre, une position et un score (0,1,2 ou 5)	RA	R	R

Table 5 - Matrice RACI Solveur

6.3.3 Le diagramme de Gantt

Ce diagramme de Gantt correspond à la partie application du projet.

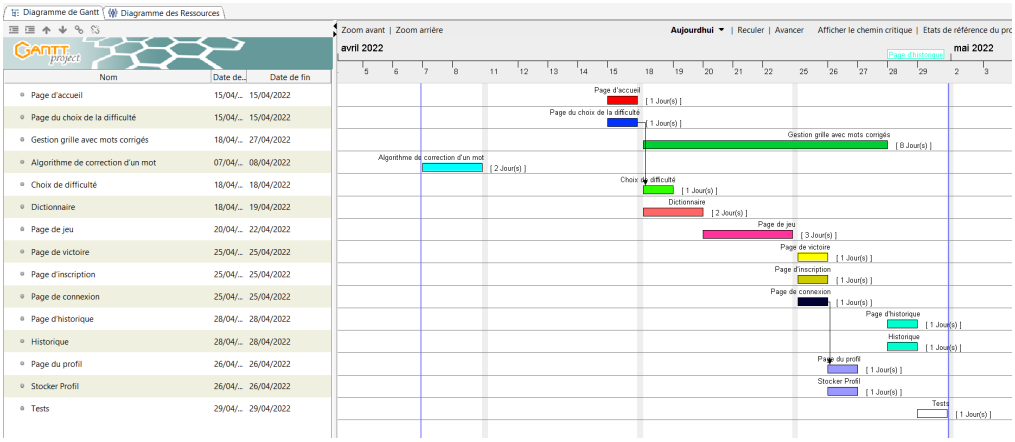


Figure 1 - Diagramme de Gantt Application

Ce diagramme de Gantt correspond à la partie solveur du projet.

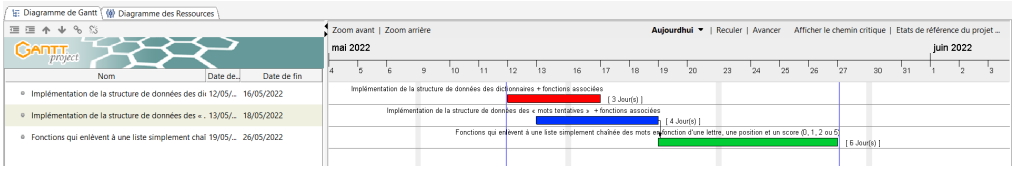


Figure 2 - Diagramme de Gantt Solveur

6.4 Comptes-Rendus

Les comptes-rendus seront mis en annexes afin d'économiser de la place.

6.5 Charte Projet

Présentation du groupe, l'équipe est composée de quatres élèves de première année du cursus ingénieur de TELECOM Nancy.

- Da Silva Jérémy : Chef de projet.
- Hoyau Christophe : Responsable de communication Coordinateur.
- Fall Ismaël : Secrétaire.

6.5.1 Description du client

Le projet est effectué à la demande d'un membre de l'équipe éducative de TELECOM Nancy.
M. Olivier Festor, Professeur en informatique.

6.5.2 Problématique

Problématique principale et prioritaire :

- Acquérir de nouvelles connaissances et compétences en algorithmique, en base de données, en web et en gestion de projets.
- Réaliser une base de données.
- Réaliser un ensemble de fonctions python.
- Réaliser un ensemble de fonctions en C.
- Réaliser un code en html et css pour la tenue du site web.
- Rédiger un rapport en Latex synthétisant le travail.

6.5.3 Objectifs

- Exploitation correcte des outils de gestion de projet (Gantt, Charte de projet, RACI, SWOT)
- Maîtriser les outils mis à notre disposition (gitlab, overleaf, Python, C, bibliothèques données en support, flask, Visual Studio Code)
- Affiner la maîtrise de flask et réaliser un code html, css et python viable
- Maîtriser la prise en main du C programming.

6.5.4 Rédaction du rapport

Le rapport a été fait au fur et à mesure de l'avancée du projet mais la grande partie a été finalisée lors des derniers jours du projet. Chaque membre s'est occupé des parties qu'il a traitées lors de la réalisation du code et chaque membre a relu l'ensemble du rapport.

Chapitre 7

Conclusion

7.1 Travail réalisé

Les principales fonctionnalités (page login, historique, choix du nombre de lettre du mot que l'on cherche) ont été implémentées sur l'application avec succès, ainsi que le solveur en C. On note cependant que certaines fonctionnalités telles que la possibilité de changer le nombre de tentative, ou encore renvoyé un mot au hasard parmis la liste de mot possible du solveur n'ont pas été ajoutées pour des raisons de temps et de mauvaise organisation.

7.2 Bilan global du projet

Points positifs	- Implémentation des fonctions principales - Travail en autonomie
Points à améliorer	- Le site est perfectible au niveau de la mise en page en ajoutant du java pour enlever la box de requete - Certaines fonctionnalités n'ont pas été implémentées
Expérience collective	- La cohésion au sein du groupe a été renforcée - Nous savons déléguer les tâches - Utilisation de Gitlab dans un contexte de projet

Chapitre 8

Annexes

8.1 Comptes-rendus

Compte-rendu de la réunion du 7 Avril 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Discord
Horaires	14h-15h
Maître de séance	Jérémy DA SILVA
Secrétaire	Ismaël FALL
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Implémentation de l'application

2) Découpage et répartition des tâches

***

1. Implémentation de l'application

Jérémy prend la parole et explique que les algorithmes servant à comparer un mot tenté avec le mot cible peuvent être implémentés assez rapidement et sans difficulté. Le groupe s'accorde sur le fait que cela peut facilement être réalisé par une seule personne. Christophe ajoute qu'il ne faudra pas oublier de refuser les mots de la mauvaise longueur.

Ismaël poursuit en soulignant qu'il faudra également s'occuper de l'aspect esthétique de l'application ainsi que la relier à une base de données dans laquelle il y aurait d'une part un dictionnaire de mot français mais également les mots ayant déjà été cibles.

Jérémy rappelle qu'il faut une page de login, ce qui s'implémenterait sous Python et Flask.

2. Découpage et répartition des tâches

Christophe se propose de traiter la majeure partie de l'implémentation Web/Flask. Jérémy, quant à lui s'occupera des algorithmes concernant la comparaison d'un mot . Enfin, en plus de la rédaction du compte-rendu de la séance, Ismaël se chargera de commencer le corps de l'application. Cela englobe notamment l'accueil, les menus, le lancement d'une partie, l'initialisation du mot cible dans une partie, le nombre d'essais... Lorsque Jérémy aura fini les fonctions à sa charge, Ismaël s'en servira pour mettre en place le jeu.

To-do list pour la prochaine réunion :

- Création du fichier app.py, initialisation des modules en rapport avec Flask et sqlite : **Christophe**
- Commencer le corps de l'application : **Ismaël**
- Fonctions de traitement d'un mot **: Jérémy**

Date et heure de la prochaine réunion : 13/12/2021 à 18h.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n'a été déformé ou oublié ! Si c'est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU

Compte-rendu de la réunion du 15 Avril 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Discord
Horaires	17h-18h
Maître de séance	Jérémy DA SILVA
Secrétaire	Christophe HOYAU
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Fin de l'implémentation d'une partie de jeu

2) Création compte + connexion

***

1. Fin de l'implémentation d'une partie de jeu

Christophe évoque la deadline fixée au 30 avril, qui sera le passage à l'oral. Jérémy commence par présenter la fonction qu'il a codé. Cette dernière prend en argument deux chaînes de caractères : un mot cible et un mot essai. Elle renvoie alors une liste composée de 2, de 1 et de 0 respectivement aux emplacements correspondants, pour le mot essai, à une lettre du mot cible bien placée, une lettre du mot cible mal placée et une lettre absente du mot cible. Ismaël explique qu'il a codé le système de grille avec possibilité de rentrer un mot cible puis d'obtenir, dans celle-ci, le mot corrigé à partir de l'algorithme précédent à l'aide d'un code couleur précis.

Christophe ajoute avoir, par-dessus ces modifications, ajouté le fond du site et avoir recadré la grille au centre de la page. Le groupe s'accorde à renoncer à l'implémentation d'un mode 2 joueurs, qui entre la gestion des différents comptes connectés et l'absence d'un membre de l'équipe serait compliquée. Il prévoit alors d'achever la partie jouable à l'aide d'une fonction qui gérera les entrées du joueur.

Jérémy poursuit avec la base de donnée dictionnaire qu'il a créée à partir d'un fichier de plus de 300 000 mots de la langue française ainsi que d'un script de sa conception. Christophe rebondit en ajoutant qu'il a ajouté tous les imports mysql nécéssaire aux différents appels de base de données.

Ismaël conclut ce premier point en planifiant de créer une page sur laquelle le joueur devra choisir 2 valeurs qui représenteront les bornes de taille entre lesquelles le mot sera choisi aléatoirement.

2. Création compte + connexion

Jérémy informe le groupe qu'il maîtrise le module flask sessions depuis le premier projet et qu'il va implémenter une page pour créer un compte ainsi qu'une autre pour se connecter. Néanmoins la connexion à un compte ne sera pas obligatoire pour jouer.

Christophe rappelle au groupe que plus tard chaque compte aura accès à ses données personnelles via une page « profil » ainsi qu'aux parties jouées via la page « historique ». Il faudra donc créer des bases de données en conséquence. Ainsi, pour avoir un historique il faudra se connecter.

To-do list pour la prochaine réunion :

- Implémentation de la fonction jeu qui sera appelée récursivement jusqu'à ce que le joueur gagne ou que la grille soit pleine **: Christophe**
- Implémentation de la page de difficulté variable puis choix d'un mot dans le dictionnaire : **Ismaël**
- Création des pages signup/login et ajout de la vérification que le mot soumis est bien dans le dictionnaire **: Jérémy**

Date et heure de la prochaine réunion : 25/04/2022 à 14h.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n'a été déformé ou oublié ! Si c'est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU

Compte-rendu de la réunion du 25 Avril 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Discord
Horaires	14h-15h
Maître de séance	Christophe HOYAU
Secrétaire	Ismaël FALL
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Bilan sur la jouabilité

2) Gestion des profils et historique des parties

***

1. Bilan sur la jouabilité

Christophe prend la parole et demande à Ismaël ou il en est. Ce dernier répond avoir terminé la gestion de la difficulté, le programme va bien chercher les mots concernés dans la base de données dictionnaire et les envoie à la partie implémentée par Christophe. Celui-ci explique alors qu'une partie est totalement jouable, il n'y a plus de soucis. Jérémy termine ce bilan en faisant la démonstration de ses pages register, login et logout. La gestion des sessions est totalement prise en compte.

2. Gestion des profils et historique des parties

Christophe ajoute qu'il va maintenant falloir stocker les parties jouées dans un historique pour chaque compte. Jérémy s'était déjà occupé de générer une base de données recensant les informations d'un compte. Il est donc possible de les restituer dans une page de profil. Ismaël se dévoue pour créer la base de données correspondant aux parties jouées. Christophe et Jérémy s'occuperont des pages d'affichages de ces informations.

To-do list pour la prochaine réunion :

- Page d’historique : **Christophe**
- Créer la base de données historique : **Ismaël**
- Page de profil : **Jérémy**

Date et heure de la prochaine réunion : Indéterminé.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n’a été déformé ou oublié ! Si c’est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU

Compte-rendu de la réunion du 10 Mai 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Télécom Nancy
Horaires	16h-17h
Maître de séance	Ismaël FALL
Secrétaire	Christophe HOYAU
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Algorithme du solveur

2) Définition des structures de données utilisées

***

1. Algorithme du solveur

Ismaël commence la réunion en demandant si le groupe a des idées pour l'algorithme à implémenter afin d'obtenir la fonctionnalité demandée. Il rappelle alors que cette dernière se caractérise par l'obtention du mot à trouver lors d'une partie de WORDLE. Jérémy propose alors de reprendre le dictionnaire précédemment utilisé dans l'application et de procéder par élimination lors de chaque essai de mot. Christophe fait alors remarquer qu'il pourra exister des problèmes lorsqu'une lettre est présente dans le mot mais une seule fois mais qu'on essaye un mot qui la contient en double. Jérémy répond que cela sera un cas à traiter dans le cas d'une lettre présente (une fois mais pas deux).

2. Définition des structures de données utilisées

Christophe soumet l'idée d'employer des structures de données similaires à celles étudiées dans les différents TP. Le groupe se met alors d'accord sur l'utilisation de listes chaînées dans un premier temps, afin de stocker les mots de la taille sélectionnée initialement dans ce qui nous fera office de dictionnaire.

Dans un second temps, Jérémy propose de créer d'autres structures qui, cette fois, serviraient à la comparaison avec le mot corrigé par l'application WORDLE. Ismaël rajoute que cela pourrait se présenter sous la forme suivante : une structure d'éléments contenant une lettre et sa place qui seraient stockés dans une structure de mots, eux-mêmes stockés

dans une structure de liste de mots essayés. Ismaël se propose d'implémenter la structure de listes chaînées, Christophe s'occupera des structures de comparaison de mots tentés avec l'aide de Jérémy qui penchera également sur la façon d'utiliser celles-ci.

To-do list pour la prochaine réunion :

- Implémenter les structures de mots wordle et les fonctions associées **: Christophe**
- Implémenter les structures de listes chaînées et les fonctions associées : **Ismaël**
- Réfléchir à la façon dont faire les fonctions de réceptions des mots corrigés **: Jérémy**

Date et heure de la prochaine réunion : 26/05/2022 à 10h.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n'a été déformé ou oublié ! Si c'est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU

Compte-rendu de la réunion du 26 Mai 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Chez Ismaël
Horaires	10h-10h45
Maître de séance	Jérémy DA SILVA
Secrétaire	Ismaël FALL
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Bilan sur les structures de données

2) Réduction des mots restants en fonction des résultats obtenus

***

1. Bilan sur les structures de données

Christophe prend la parole et explique au groupe que les structures de données des wordles éléments, des worldes mots et des wordles tentatives et les fonctions associées sont opérationnelles. Ismaël poursuit en montrant qu'il a également terminé les structures de listes chaînées et leurs fonctions. De multiples tests ont été faits et tout fonctionne.

2. Réduction des mots restants en fonction des résultats obtenus

Jérémy enchaîne alors en expliquant ce qu'il pourrait faire pour la prochaine réunion : il va implémenter différentes fonctions interagissant avec les structures dictionnaires via les structures wordle permettant de réduire la liste de mots possibles en fonctions des résultats envoyés. Il y en aurait 3 + 1 qui elle permettrait d'obtenir un mot directement depuis le fichier contenant l'ensemble du dictionnaire français. Les 3 fonctions en question permettraient :

- De limiter la liste à tous les mots contenant une certaine lettre à un endroit précis
- De limiter la liste à tous les mots contenant une certaine lettre qui ne se situe pas à un endroit précis
- De limiter la liste à tous les mots ne contenant une certaine lettre

Jérémy, qui avait déjà en partie débuté leur implémentation, se propose de les achever avant la prochaine réunion. Avec l'aide d'Ismaël et Christophe, il faudra ensuite créer la

fonction solveur qui emploie ces fonctions et qui crée le lien avec le facteur humain (il faudra rentrer les résultats manuellement).

To-do list pour la prochaine réunion :

- Se pencher sur la fonction solveur : **Christophe**
- Aider Jérémy dans l'implémentation de la fonction de recherche des mots d'une certaine taille dans le fichier dictionnaire : **Ismaël**
- Achever les fonctions de filtrage de la liste de mots : **Jérémy**

Date et heure de la prochaine réunion : 30/05/2022 à 10h.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n'a été déformé ou oublié ! Si c'est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU

Compte-rendu de la réunion du 30 Mai 2022

-

Projet Pluridisciplinaire d'Informatique Intégrative

Type de réunion	Réunion d'avancement
Lieu	Chez Ismaël
Horaires	10h-11h30
Maître de séance	Ismaël FALL
Secrétaire	Jérémy DA SILVA
Membres présents	Christophe HOYAU Jérémy DA SILVA Ismaël FALL
Membres absents	Cyrielle LACRAMPE-DITER

Ordre du jour

1) Bilan du solveur

2) Tests pratiques

***

1. Bilan du solveur

Jérémy prend la parole et explique avoir passé énormément de temps sur les fonctions de réduction de la liste. Il explique que plusieurs erreurs d'adressage ont eu lieu et qu'il n'était pas évident de les trouver. Il a également eu des problèmes avec les doublons. Cependant, avec l'aide d'Ismaël et Christophe, il a terminé la fonction solveur. Le groupe décide maintenant de s'attaquer à une batterie de tests.

2. Tests pratiques

Christophe lance l'application WORDLE sur son mac pendant que Jérémy lance le solveur sur son pc. Le groupe décide d'essayer des mots de 5 lettres et teste plusieurs parties à l'issue desquelles le solveur gagne toujours. Ismaël fait alors remarquer que maintenant qu'il n'y a plus de problèmes de doublons, le solveur ne perd jamais. Le groupe décide de passer à 7 lettres. Une erreur se produit alors, après quelques minutes de recherches, il s'avère que nous avons simplement oublié de changer le paramètre de taille dans le solveur. Finalement, le groupe s'attaque à la rédaction du rapport final.

Date et heure de la prochaine réunion : Indéterminé.

Signature : [Indiquez votre nom]

Ne signez que si vous estimez que le compte-rendu est un reflet fidèle de la réunion et que rien n'a été déformé ou oublié ! Si c'est le cas, signalez-le aux secrétaires de séance.

- Jérémy DA SILVA
- Ismaël FALL
- Christophe HOYAU