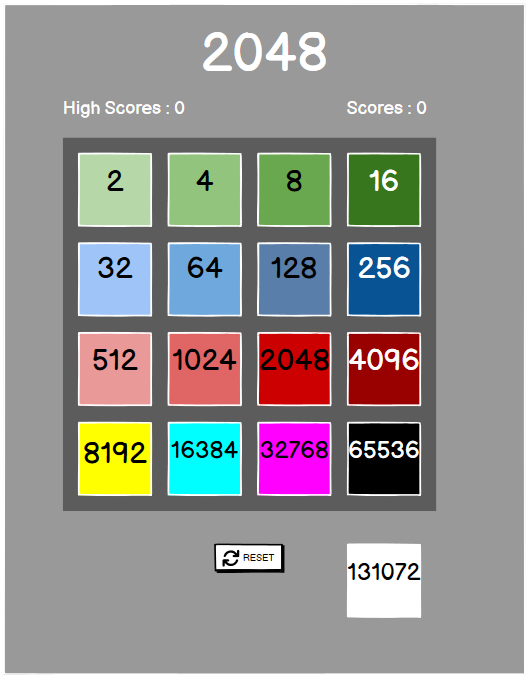


|  |
| --- |
| Jeu 2048 |



Ryan Bersier

Ryan.bersier@cpnv.ch



SI-C1a

23/02/2023

Table des matières

[1 Introduction 3](#_Toc131348552)

[1.1 Cadre, description et motivation 3](#_Toc131348553)

[1.2 Objectifs 3](#_Toc131348554)

[1.3 Planification initiale 3](#_Toc131348555)

[2 Analyse 4](#_Toc131348556)

[2.1 Maquettes 4](#_Toc131348557)

[2.2 Stories / tests d’acceptation 5](#_Toc131348558)

[2.3 Stratégie de test 5](#_Toc131348559)

[3 Implémentation 5](#_Toc131348560)

[3.1 Vue d’ensemble 5](#_Toc131348561)

[3.2 Choix techniques 5](#_Toc131348562)

[3.3 Points techniques spécifiques 6](#_Toc131348563)

[3.3.1 Point 1 6](#_Toc131348564)

[3.3.2 Point 2 6](#_Toc131348565)

[3.3.3 Point … 6](#_Toc131348566)

[4 Tests 7](#_Toc131348567)

[4.1 Tests effectués 7](#_Toc131348568)

[4.2 Erreurs restantes 8](#_Toc131348569)

[5 Conclusions 8](#_Toc131348570)

[6 Annexes 9](#_Toc131348571)

[6.1 Sources – Bibliographie 9](#_Toc131348572)

[6.2 Journal de travail 9](#_Toc131348573)

# Introduction

## Cadre, description et motivation

Le choix a été fait par les enseignants qui ont estimé que le jeu 2048 pouvais être réaliser durant le trimestre 3. Ce que peu m’apporter le projet sont :

* Une amélioration dans le langage de programmation python
* Apprendre la documentation correctement
* Gérer un projet de A à Z

Les travaux déjà effectué sont :

Etape 1

* Une maquette du jeu (fenêtre Windows) en utilisant des objets connus (labels, contrôles texte…).
* Un code avec deux exemples de tableaux/listes en mémoire Au début du jeu (quasi vide, avec 2 tuiles montrant un 2) Contenant toutes les valeurs possibles comme si un jeu avait toutes les tuiles de 2 à 65'536.

Etape 2

* Des tuiles de toutes les valeurs de 2 à 131’072 (=17 cases, de 21 à 217) avec leurs couleurs
* 3 tuiles vides.

Etape 3

* Commencement des reconnaissances des lignes pour les déplacements et la fusion

## Objectifs

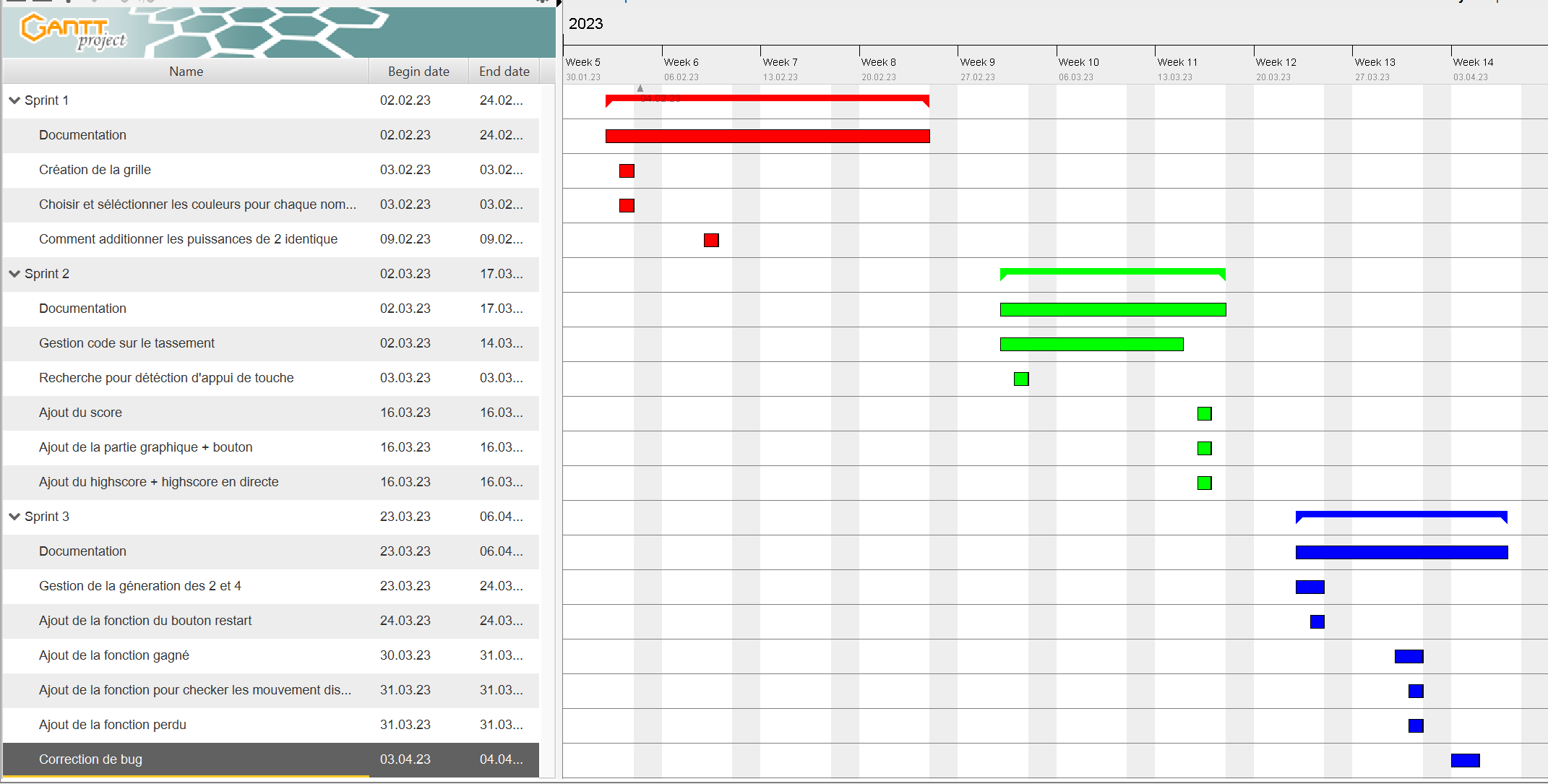
Les Objectifs pour le ICT-431 sont :

* Pouvoir analyser correctement un projet et sa réalisation
* S’organiser, planification des différentes tâches du projet
* Pouvoir exécuter correctement un mandat de demander dans le temps a partie
* Apprendre la documentation de projet pour savoir où en ai le projet

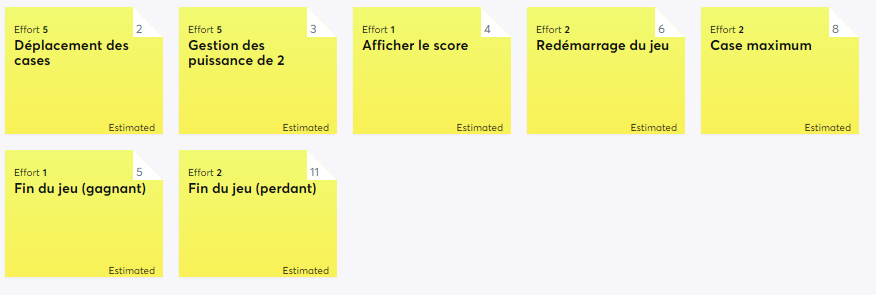
Les Objectif pour le MA-20 sont :

* Créer une application « 2048 » fonctionnelle
* S’améliorer dans la conception d’application en python

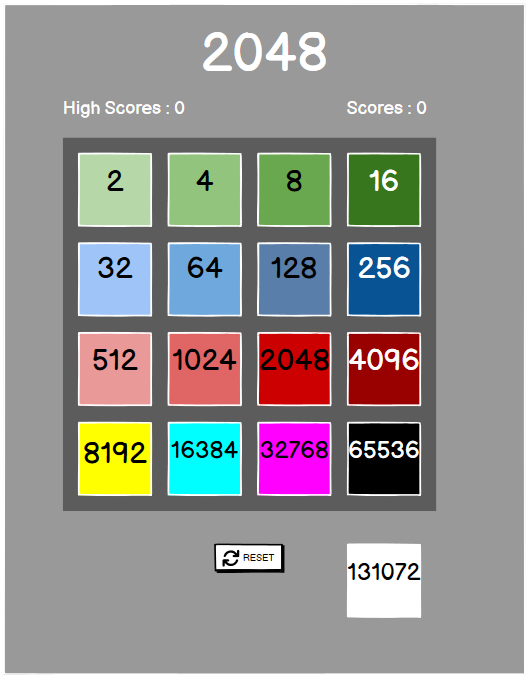
## Planification initiale



# Analyse



## Maquettes



## Stories / tests d’acceptation

https://icescrum.cpnv.ch/p/2048RYANBE/#/project

## Stratégie de test

Pendant la création la majeure partie des tests seront des tests unitaire certaines fonctions pourront être faites en test intégrations et à la fin un test système sera fait.

Ce qui est à mettre œuvre est une installation de python 3 avant les tests et disposé de la dernière version de l’exécutable

Les données de test à prévoir sont ce qui est attendu dans le cahier des charges par le client ainsi que ce qui a été noté dans Icescrum dans les différents sprints

Les personnes qui vont tester l’applications sont Amis, Parents, Camarade de classe ou d’autres classes.

Les tests vont être fait sur ma machine personnels qui est un pc portable sous Windows 11 ainsi que sur les ordinateurs fixe de l’écoles qui sont sous Windows 10

# Implémentation

## Vue d’ensemble

L’utilisateur pourra démarrer un point EXE et il pourra utiliser les flèche, WASD ou wasd pour déplacer les case son but sera d’atteindre 2048. Il pourra aussi choisir de reset sa partie s’il le souhaite grâce à un bouton

## Choix techniques

Matériels : PC fixe de l’école et PC portables personnels

Système d’exploitation : Windows 10 et Windows 11

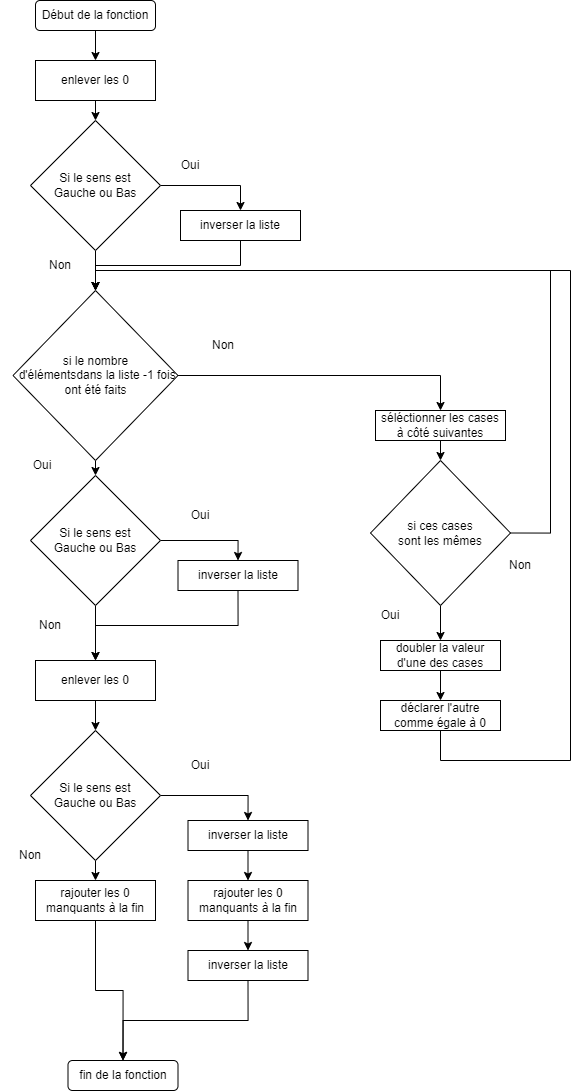
Logiciel : Pycharm Edu et Pycharm Comunity

Langage de développement et version : Python 3.8 et Python 3.11.2

L’avantage d’utiliser 2 machines avec jamais les mêmes paramètres est de pouvoir voire la comptabilité et le fonctionnement

## Points techniques spécifiques

Voici le diagramme de flux sur la fonction de tassage :



Voici le pseudo-code pour la génération de 2 et 4 :

Import de la bibliothèque Aléatoire

X = Aléatoire entre 0 et 9

Fonction départ jeu

SI Grille = [«», «», «», «»,], [«», «», «», «»,], [«», «», «», «»,], [«», «», «», «»,]

Invocation fonction aléatoire

Invocation fonction aléatoire

Fonction aléatoire

Tant que vrai boucle

Y = Aléatoire entre 0 et 3

Z = Aléatoire entre 0 et 3

SI Grille[Y][Z] = «»

Grille[Y][Z] supprime les «»

SI X = 0

Grille[Y][Z] ajoute 22

Sortie de boucle

Ou alors

Grille[Y][Z] ajoute 21

Sortie de boucle

# Tests

## Tests effectués

**Test du Sprint 1 :**

Affichage de la grille V

Puisque j’ai l’exécutable du jeu  
Quand je le démarre  
Alors la grille de 4x4 case s’affiche

Affichage de 2 nombre 2 et de 1 nombre 4 V

Puisque j’ai l’exécutable du jeu  
Quand je le démarre  
Alors 2 case contenant un 2 et 1 case contenant un 4 sont placés sur la grille

Une autre grille s'affiche de toute les puissances de 2 (2^16) V

Puisque j’ai l’exécutable du jeu  
Quand je le démarre  
Alors toutes les puissance de 2 jusqu'à 2^16 apparaisse sur une autre grille

Affichage de la couleur de fond en fonction du nombre V

Puisque j’ai l’exécutable du jeu  
Quand je le démarre  
Alors chaque nombre prend une couleurs à spécifique à lui

**Test du Sprint 2 :**

Déplacement des tuiles V

Puisque j'ai 4 case par groupe de grille

Quand une case est vide dans le sens de la flèche appuyer

Alors les tuiles se déplace dans le sens de la flèche appuyer

Nombre de déplacement V

Puisque je dispose de l'exécutable du jeu

Quand j'utilise les flèches

Alors la variable du déplacement prend + 1 à chaque appuie

Addition identique V

Puisque je dispose de l'exécutable du jeu

Quand j'utilise les flèches et que 2 tuiles identiques différente de vide se trouve côte à côte ou séparer par du vide

Alors les tuiles s’additionnent

Disparition de tuile V

Puisque je dispose de l'exécutable du jeu

Quand j'utilise les flèches et que 2 tuiles s’additionnent

Alors le deuxième nombre disparaît

Addition du score V

Puisque je dispose de l'exécutable du jeu

Quand je joue et que 2 tuiles s’additionnent

Alors leurs résultats s’ajoutent au score

Reset X

Puisque je dispose de l'exécutable du jeu

Quand je reset le jeu

Alors le score retourne à 0

Démarrage du score V

Puisque je dispose de l'exécutable du jeu

Quand j'exécute le jeu

Alors le score est de 0

Bouton reset X

Puisque je dispose de l'exécutable du jeu

Quand j'appuie sur le bouton

Alors le jeu redémarre avec 2 tuiles 2 sur la grille

Bouton reset afficher V

Puisque je dispose de l'exécutable du jeu

Quand je démarre le jeu

Alors un bouton reset se trouve sur la partie graphique du jeu

Actualisation du highscore V

Puisque je dispose de l'exécutable du jeu

Quand le score est supérieur au highscore

Alors le highscore s'incrémente en même temps que le score

Sauvegarde highscore V

Puisque je dispose de l'exécutable du jeu

Quand je ferme le programme

Alors le highscore reste en mémoire grâce à un fichier txt

**Test du Sprint 3 :**

Bouton reset V (test du précèdent sprint qui étais pas terminé ou avec quelque bug)

Puisque je dispose de l'exécutable du jeu

quand j'appuie sur le bouton

alors le jeu redémarre avec 2 tuiles 2 ou/et 4 sur la grille

Reset V (test du précèdent sprint qui étais pas terminé ou avec quelque bug)

Puisque je dispose de l'exécutable du jeu

quand je reset le jeu

alors le score retourne à 0

Case maximum V

Puisque je dispose de l'exécutable du jeu

quand je joue je peux additionner des case jusqu'à 2 puissances 17

alors j'obtiendrait une case contenant 131'072 sur le terrain

Victoire V

Puisque je joue au jeu

quand j'obtiens une case 2048 sur la grille

alors une image apparait pour féliciter le joueur

Continuer V

Puisque je joue au jeu

quand l'écran de victoire apparaît je peux appuyer sur un bouton continuer

alors je reviens a l'état de la grille avant que l'écran de victoire apparaisse

Recommencer V

Puisque je joue au jeu

quand l'écran de victoire apparaît je peux appuyer sur un bouton recommencer

alors le jeu recommence

Quitter V

Puisque je joue au jeu

quand l'écran de victoire apparaît je peux appuyer sur un bouton quitter

alors le jeu se ferme

Défaite V

Puisque je joue au jeu

quand plus aucun mouvement ne peux être fait

alors apparaît un écran de défaite

Recommencer V

Puisque je joue au jeu

quand l'écran de défaite apparaît je peux appuyer sur un bouton pour recommencer

alors le jeu recommence

Quitter V

Puisque je joue au jeu

quand l'écran de défaite apparaît je peux appuyer sur un bouton pour quitter

alors le jeu se ferme

Génération de 2 et de 4 V

Puisque je joue au jeu

quand je fait un mouvement

alors une case 2 ou 4 se génère aléatoirement

Chance d'apparition V

Puisque je joue au jeu

quand je fait un mouvement

alors il y a 90% de chance que la case soit un 2 et 10 % qu'elle soit un 4

Démarrage du jeu V

Puisque je dispose de l'exécutable du jeu

quand je démarre le jeu

alors 2 cases 2 ou/et 4 sont placé de façon aléatoire sur la grille

## Erreurs restantes

Sprint 2 : aucune erreur restante

Sprint 3 : aucune erreur restante

# Conclusions

J’estime avoir atteint tous mes objectifs. Si on avait eu plus de temps j’aurais essayé de faire une application modulaire. J’ai appris beaucoup de choses principalement sur le code ; cependant, pour un tel projet, Je pense qu’il aurait été mieux d’utilisé la librairie Pygame qui est conçu pour faire des jeux, globalement je suis fière de moi et de ce que j’ai pu accomplir car j’estime n’avoir pas eu tant d’aide que ça, malheureusement j’estime avoir mal optimisé mon temps ce qui m’a contraint de travailler chez moi ou de prendre du temps pendant certains cours où on m’a autorisé bien évidement. J’ai rencontré quelques difficultés que je peux énumérer mais pas vraiment de difficulté particulière j’ai eu des difficultés surtout sur la fonction mix de mon code ainsi que sur le check des mouvements il m’a été difficile une fois lancé dans la méthode de gestion de tableau de gérer certains points du code. Les éventuelles améliorations sont une grille modulaire exemple pouvoir choisir si on veut du 2x2 ou du 10x10 ainsi qu’une réelle fenêtre de highscore qui pourrait demander le nom de la personne lorsqu’elle réussi à dépasser l’ancien highscore, ainsi nous aurions toujours un suivi des anciens highscores ainsi que les noms des joueurs.

# Annexes

## Sources – Bibliographie

Référence pour le projet :

* Overflow :

Beaucoup de recherche qui n’ont pas aboutie

* W3school :

Le module random(pour la génération de 2 et 4)

<https://www.w3schools.com/python/module_random.asp>

Gestion de fichier (pour le highscore (mais rien compris))

<https://www.w3schools.com/python/python_file_handling.asp>

* Chaîne YouTube de Graven :

Gestion de fichier (pour le highscore)  
<https://www.youtube.com/watch?v=jOHpZg8k668>  
Gestion des dictionnaire (pas besoin au final)  
<https://www.youtube.com/watch?v=0pgYBjW1OVM&list=PLMS9Cy4Enq5JmIZtKE5OHJCI3jZfpASbR&index=11>

Aide reçu :

* Brad de la SI-C1a
* Judah de la SI-CMI1a
* Martin de la SI-CMI1b
* Frédérique ANDOLFATTO Professeur du projet

Testeur :

* Alexis de la SI-CMI1a
* Christophe de la SI-C1a
* Loïc de la SI-C1a
* Mes parents
* Moi-même

## Journal de travail

