**Résumé du TPI**

Ce projet ma été donné suite à mon choix de pré-TPI qui était de faire un Tetris en Swift. Pour ce projet, j’ai eu comme mission de créer un jeu à but pédagogique du même style que le célèbre *Space Invaders*.

J’ai d’abord commencé par analyser le projet pour être sûr d’avoir bien compris ce qui était demandé, ensuite nous avons eu, avec mon collègue qui faisait le même jeu mais sous Android, un rendez-vous avec notre chef de projet pour discuter ce qui était exactement attendu et nous avons pu lui poser des questions sur des points pas clairs. Après avoir éclaircis tous les points, j’ai fait la planification initiale puis validée par mon chef de projet. Ensuite, je me suis lancé dans la réalisation des Use case et scénarios ainsi que des maquettes. Après cela, j’ai commencé à coder l’écran du menu d’avant partie avec des données hard codées, au milieu de la seconde semaine, la page était finie et il y avait un bouton « retour » d’intégré. Mon chef de projet m’a donné des données dans un tableau Excel qui contenait des langues et des vocabulaires et j’ai donc dû les intégrer de manière hard codée mais de façon à ce qu’elles simplifient le travail pour l’implémentation du Web Service par la suite. Je me suis donc inspiré de la structure MVC afin de séparer les données du reste. Ce point m’a fait perdre beaucoup de temps car je n’avais jamais eu l’occasion, avec Swift, de créer dynamiquement des objets afin qu’ils soient indépendants les uns des autres. J’ai donc cherchré sur internet différentes manières pour faire ceci et je suis tombé sur une méthode où il faut assigner un « .name » et non un « .id » dynamiquement a l’entité créée. De plus, il fallait utiliser la fonction « .copy » avec les labels à fin d’éviter de créer plusieurs fois le même enfant et cela me permettait de créer plusieurs entités « copiée » n’ayant pas même « .name ».

Ensuite je me suis attaqué au jeu en lui-même, j’ai codé l’élève et j’ai utilisé la même manière que pour les labels vus que les élèves devaient être tous indépendants mais bouger ensemble. Après cela, j’ai commencé à coder le professeur et l’avion avec leur déplacement. J’avais rencontré un problème où le prof ne bougeait plus ou sortait de l’écran de temps en temps parce que le déplacement était géré dans un « while ». En discutant avec mon chef de projet nous avons trouvé une fonction « update » de Swift qui s’exécute à chaque frame du jeu. J’ai alors refactorisé le code et séparé la création de mes Sprites dans d’autres fichiers, cela m’a permis d’économiser beaucoup de temps pour la suite. Après cela, j’ai fait en sorte que ce soit les bons mots qui se trouvent dans les mains des élèves et du côté du professeur. Après, j’ai codé pour que ce soit les mots qui correspondent qui touchent l’élève et ceux qui ne correspondent pas, font avancer l’élève. Avec mon chef de projet et suite au retard que j’avais pris au début, nous avions décidé de supprimer la fonctionnalité où l’on doit toucher 3x l’élève pour aller en pause et le Web Service.

Pour finir, les points principaux du jeu ont été implémentés malgré quelques bugs restant. Il est possible de choisir deux langues (celle du professeur et celle de l’élève) et choisir un vocabulaire correspondant. Les bons mots s’affichent à l’écran, les élèves bougent de la manière voulue ainsi que le professeur et il y a un début et une fin de partie.