

GEN – Mini Projet

Version number 1.1.4

Drunk&Smart – Rapport Intermédiaire

Table of Contents

Version number 1.1.4.....	1
I. Fonctionnement général.....	3
1) Synopsis du jeu.....	3
2) Modes de jeu.....	3
a) Aventure.....	3
b) Challenge.....	4
c) (Très optionnel) Le mode plateforme.....	4
3) Ressources.....	4
a) Points de vie.....	5
b) Points d'expérience.....	5
c) Niveaux.....	5
d) Objets.....	5
e) État d'avancement dans les scénarios.....	6
f) Pool de question à poser.....	6
4) Classe de joueur.....	7
5) Objectifs.....	8
6) Déroulement du Jeu.....	8
7) Déroulement d'un affrontement (joueur contre ordinateur).....	9
8) Déroulement d'un affrontement (joueur contre joueur).....	10
9) Interface graphique.....	10
II. Cas d'utilisation.....	11
1) Diagramme général.....	11
2) Acteurs.....	13
a) Administrateur.....	13
b) Joueur.....	13
3) Scénarios.....	14
a) Joueur contre Joueur.....	14
b) joueur contre serveur.....	15
c) Général.....	16
III. Ébauches du modèle de domaine.....	17
1) Modèle client.....	17
2) Modèle serveur.....	17
IV. Base de données.....	18
V. Serveur et Protocole d'échange.....	18
1) Serveur.....	18
2) Protocole.....	19
VI. Ébauches interfaces utilisateur.....	20
1) Starting Screen.....	20
2) Menu Screen.....	20
3) Adventure Screen.....	21
4) Adventure Screen [Fight].....	21
VII. Gestion de projet.....	22
1) Rôle.....	22

2) Backlog.....	22
3) Plan d'itération.....	22
VIII. Planification détaillée des sprints.....	24
1) Sprint numéro 1.....	24
2) Sprint numéro 2.....	24
Sprint numéro 3.....	25
IX. Bilan Sprint.....	25
1) Sprint numéro 1.....	25
2) Bilans personnels Sprint numéro 1.....	26
3) Sprint numéro 2.....	27
4) Bilans personnels Sprint numéro 2.....	28
5) Sprint numéro 3.....	29
6) Bilans personnels Sprint numéro 3.....	30

I. Fonctionnement général

1) Synopsis du jeu

Le jeu est un jeu de question-réponse (quizz), joueur contre joueur ou joueur contre ordinateur. Les joueurs incarnent toujours un étudiant et l'ordinateur sera soit assistant ou un professeur de la HEIG-VD.

Il existe donc deux mode de jeu : le mode Aventure (joueur vs ordinateur) et le mode Challenge (joueur contre joueur). Les joueurs progressent en répondants correctement, parmi les quatre choix de réponse possible, aux diverses questions qui lui sont posées. Il récupère également les questions auxquelles il a répondu correctement.

Quand un joueur possède un certain nombre de questions, il peut alors commencer à défier d'autres joueurs grâce au mode Challenge.

2) Modes de jeu

Il existe plusieurs modes de jeu disponibles :

- Aventure
- Challenge
- Plateforme (Priorité très faible)

a) Aventure

Ce mode permet de lancer une aventure ou scénario. Chaque scénario porte le nom d'une filière de la HEIG-VD.

Scénarios :

- TIC

- TIN (Priorité très faible)
- EC+G (Priorité très faible)
- COMEM (Priorité très faible)
- HEG (Priorité très faible)

Une fois le scénario choisi l'aventure débute. Un scénario consiste en une succession de niveaux, jusqu'au niveau final. Lors de chacun de ces niveaux, l'étudiant devra affronter un certain nombre d'assistants et les vaincre avant de pouvoir affronter un professeur puis une fois celui-ci vaincu il passe à l'examen final qu'il devra passer dans un temps limité.

Une fois l'ultime professeur (légendaire) et son examen final réussi, le scénario est alors terminé.

Après avoir remporté un certain nombre de victoire dans ce mode, et donc obtenu un certain nombre de question, il est alors possible d'accéder au mode Challenge.

b) Challenge

Ce mode consiste à défier un autre joueur. Chaque joueur pose tour à tour une question qu'il a gagnée lors de l'un de ses modes Aventure à son adversaire. Le temps pour poser la question est pour y répondre est limité afin d'éviter qu'un joueur puisse prendre en otage l'affrontement en laissant tourner l'application jusqu'à ce que l'autre joueur quitte l'application.

(Optionnel) Le joueur qui gagne le Challenge remporte des points de classements lui donnant un classement général parmi tous les joueurs inscrits. Le joueur qui perd, perd alors des points de classement.

c) (Très optionnel) Le mode plateforme

Il s'agit d'un mode alternatif permettant de remporter des objets utiles pour le mode Aventure ou Versus. Il s'agirait d'un mini-jeu de plateforme où le joueur manipulerait son personnage à travers différents niveaux pour récupérer les objets.

3) Ressources

- a) Point de vie
- b) Point d'expérience

- c) Niveau
- d) Objets
- e) État d'avancement dans les scénarios
- f) Pool de question à poser

a) Points de vie

Chaque entité joueur (ordinateur ou joueur « humain ») possède un nombre de point de vie initial (défini par sa classe).

Les points de vie sont perdu soit parce que le joueur a répondu incorrectement à une question posée, soit parce que son adversaire a répondu correctement à une question qu'il lui a posé.

Les points de vie ne peuvent être gagné que grâce à certains objets.

b) Points d'expérience

Seul les joueurs « humain » possèdent des points d'expérience. Ces points sont acquis par le joueur lorsque celui-ci remporte un affrontement. Il ne sont jamais perdu. Une fois un certain palier atteint, le joueur gagne un niveau mais ces points d'expérience retournent à 0. Les paliers à atteindre s'incrémentent de niveau en niveau.

c) Niveaux

Les niveaux sont un indicateur de la menace que représente un adversaire (ordinateur ou humain). Ils sont gagnés une fois un certain palier de point d'expérience atteint. Ils ne sont JAMAIS perdus.

d) Objets

Chaque joueur démarre le jeu avec un ou plusieurs objets initiaux (en fonction de sa classe). Les objets sont des "joker" qui permettent aux joueurs d'obtenir un regain de point de vie, une aide pour une question ou autre allègement de la difficulté.

Les objets peuvent être gagnés :

- En récompense à la fin d'un affrontement remporté (pourcentage de chance)
- Lors du mode plateforme (Très optionnel)

Les objets sont perdus après leur utilisation.

Exemple d'objets :

- Une bière : redonne des points de vie
- Une anti-sèche : enlève une mauvaise réponse des propositions de réponses
- Un certificat médical : permet de passer à une autre question sans dommage aucun.

Ces exemples ne sont ni exhaustifs, ni définitifs.

e) État d'avancement dans les scénarios

Chaque joueur possède un état d'avancement pour chaque scénario qu'il a démarré. Chaque scénario possède des « checkpoints » et lorsqu'un joueur atteint l'un de ces « checkpoints » son état d'avancement est mis à jour. Les endroits et l'espacement des « checkpoints » ne sont pas encore défini. Si un joueur quitte un scénario (ou l'application), pendant un affrontement ou n'importe quand, il reprend son état au derniers « checkpoint » atteint.

f) Pool de question à poser

Il s'agit d'une base de question dans laquelle joueur comme ordinateur peuvent aller chercher une question à poser à son adversaire. Les questions sont toujours liées à leurs propositions de réponses.

Le joueur possède

- Un nombre de point de vie
- Un nombre de point d'expérience
- Un niveau
- Zéro à plusieurs objets pouvant l'aider
- Un état d'avancement dans le jeu
- Un pool de questions durement acquises

Les professeurs/assistants ont

- Un nombre de point de vie
- Un niveau
- Un pool de questions sur le serveur, définies par les administrateurs

4) Classe de joueur

Il s'agit d'une précision concernant le type d'étudiant que le joueur incarne. Par exemple, l'intello, le fêtard, le glandeur, le tricheur etc. *(il s'agit d'exemple et ce n'est pas définitif)*

La classe définira

- Les points de vie initial du joueur
- Les « dégâts » du joueur *(optionnel)*
- Les objets de départ du joueur
- Les objets qu'il a le plus de chance de gagner *(optionnel)*
- Autres avantages à voir *(optionnel)*

5) Objectifs

- Mode aventure

Le but de ce mode est de terminer un à plusieurs des scénarios qui sont proposés. Lors d'un scénario, le but est de réussir tous les niveaux qui le compose. Pour réussir un niveau, il faut remporter tous les affrontements proposés dans celui-ci.

- Mode Challenge

Ce mode propose des affrontements contre d'autres joueurs. Le but est donc de remporter cet affrontement contre le joueur adverse. Le joueur remportant l'affrontement est déclaré vainqueur du Challenge.

- Remporter un affrontement

Pour remporter un affrontement, il faut que les points de vie de son adversaire (joueur ou ordinateur) tombe à zéro.

6) Déroulement du Jeu

1. Le joueur lance le jeu
2. **Si le joueur ne possède pas déjà un compte**
 - Le joueur est invité à le faire
 - Il choisit alors sa classe**Sinon**
 - passe à l'étape 3
3. Le joueur arrive sur son menu principal (dashboard) ou il peut voir:
 - Son niveau
 - Son nombre de point d'expérience
 - Ses objets éventuels

A ce niveau le joueur peut (via des boutons):

- Démarrer le mode Aventure (joueur contre ordinateur) et passe à l'étape 4
- **Si il possède suffisamment de question**, démarrer le mode Challenge (joueur contre joueur) et passe à l'étape 8
- Accéder à la liste de ses haut-faits, ses réussites (**optionnel - sera peut être intégré**)
- Accéder à un tableau des scores (**optionnel - sera peut être intégré**)

4. Le joueur accéder à sa prochaine confrontation, selon son avancement dans le jeu. Il affronte alors le prochain assistant ou professeur. **(voir déroulement d'un affrontement)**

5. **Si le joueur gagne**

- il gagne de l'expérience
- il gagne les questions posées par le professeur
- il gagne Zéro à plusieurs objets
- il peut passer à la suite du mode (avancement dans le jeu)

Si le joueur perd

- il ne gagne rien
- les objets utilisés lors de l'affrontement sont perdus
- il reste au même point dans le mode (il n'avance pas plus dans le jeu)

6. **Si le joueur possède suffisamment de point d'expérience**

- joueur gagne un niveau

Sinon

- passer à l'étape 7

7. Recommencer à l'étape 4 jusqu'à ce que le mode Aventure soit terminé.

8. Le joueur choisit son adversaire (autre joueur)

9. Le joueur lance alors un affrontement **(voir déroulement d'un affrontement)**

10. **Si le joueur gagne**

- il gagne de l'expérience
- il gagne Zéro à plusieurs objets
- il gagne des points de classement **(optionnel - sera peut être intégré)**

Si le joueur perd

- il ne gagne rien
- les objets utilisés lors de l'affrontement sont perdus
- il perd des points de classement **(optionnel - sera peut être intégré)**

7) Déroulement d'un affrontement (joueur contre ordinateur)

1. Le professeur/assistant pose une question sur son domaine d'enseignement

2. Le joueur se voit proposer quatre propositions de réponses

3. **Le joueur peut alors**

- choisir l'une des quatre options et passe à l'étape 4

- utiliser un objet s'il en possède un (**voir exemple d'objet**) et reste à l'étape 3
- 4. **Si le joueur choisi la réponse correct**
 - le professeur/assistant perd des points de vie
- Sinon**
 - Le joueur perd des points de vie
- 5. Si le joueur n'a plus de points de vie → le joueur perd (**fin de l'affrontement**)
- 6. Si le professeur n'a plus de points de vie → le joueur gagne (**fin de l'affrontement**)
- 7. Retour à l'étape 1.

8) Déroulement d'un affrontement (joueur contre joueur)

1. On tire au hasard le joueur qui commence
2. Le premier joueur pose alors une question qu'il possède au joueur adverse
3. **Ce dernier peut alors**
 - choisir l'une des quatres options de réponse et passe à l'étape 4
 - utiliser un objet s'il en possède un (**voir exemple d'objet**) et reste à l'étape 3
4. **Si il choisi la réponse correct**
 - le joueur ayant posé la question perd des points de vie
- Sinon**
 - Il perd des points de vie
5. Si il n'a plus de points de vie → Il perd (**fin de l'affrontement**)
6. Si le joueur ayant posé la question n'a plus de points de vie → il gagne (**fin de l'affrontement**)
7. Retour à l'étape 1 et le joueur posant la question change (et ainsi de suite)

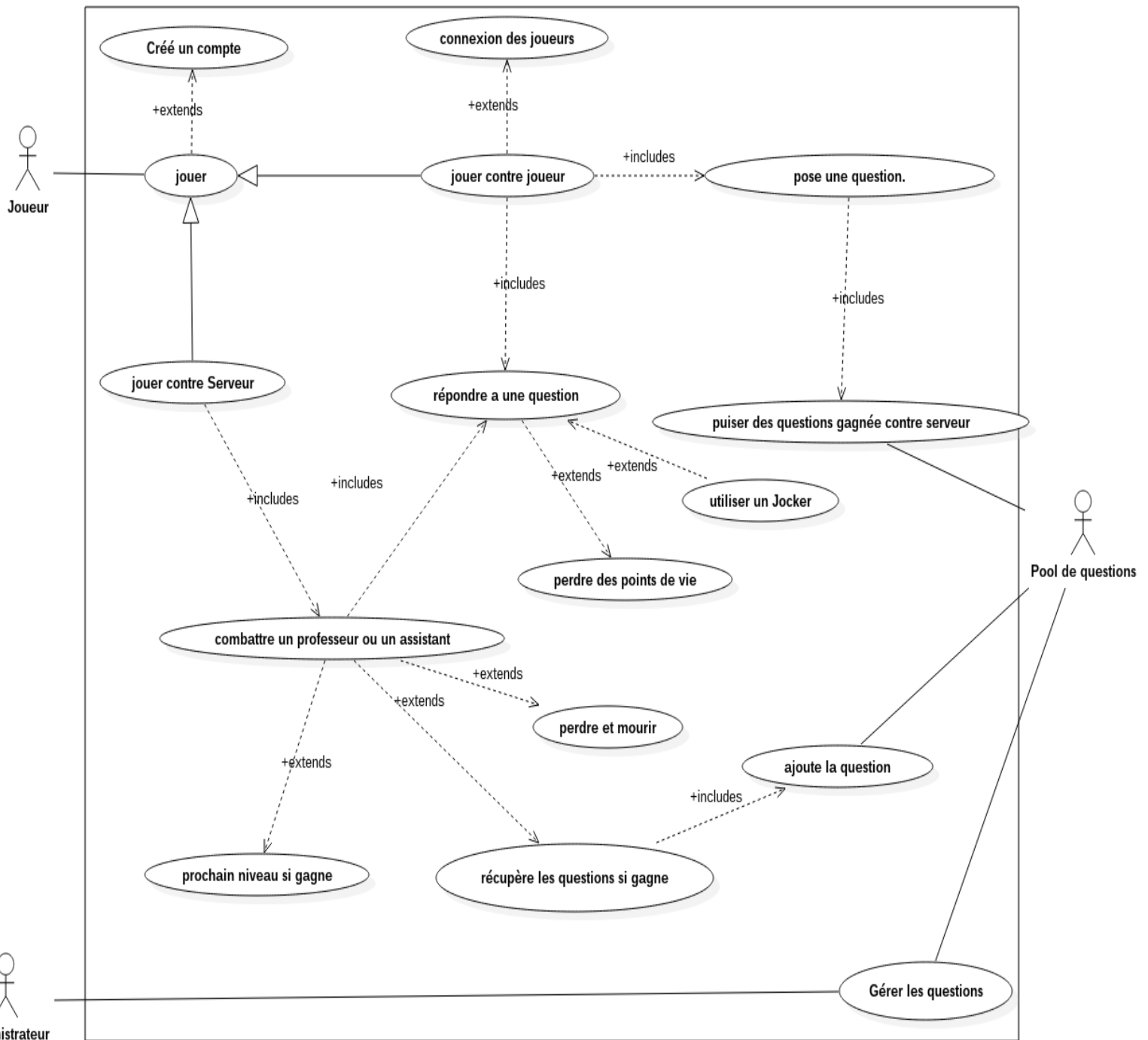
N'importe quand si un joueur quitte l'application pendant l'affrontement, la victoire est concédée à son adversaire (par abandon).

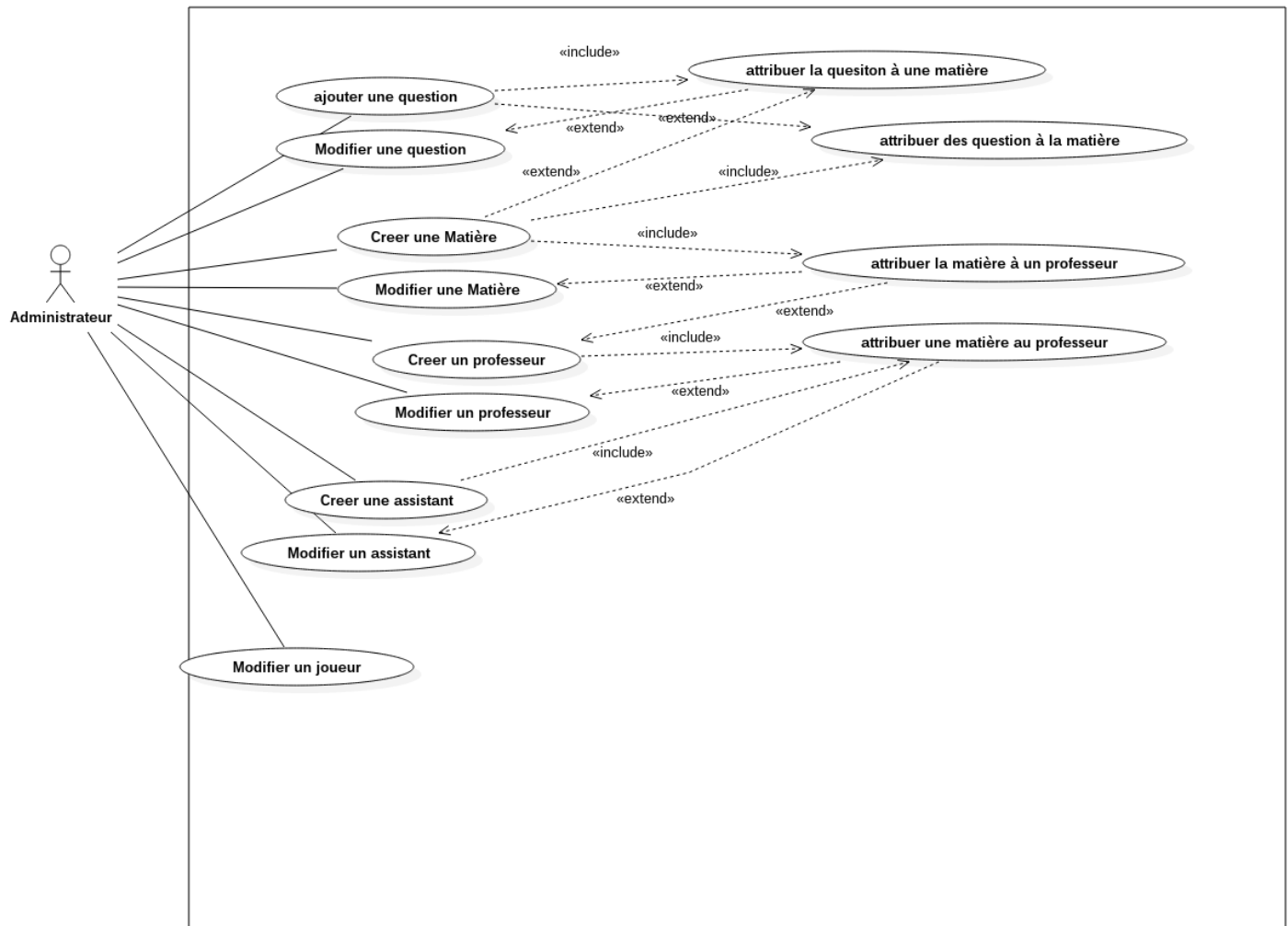
9) Interface graphique

Lors des affrontements, l'interface sera à peu près le même que celui des jeux Pokemon (voir brouillon interfaces utilisateurs - **Point VI**).

II. Cas d'utilisation

1) Diagramme général





2) Acteurs

a) Administrateur

- Ajouter/Créer/Modifier

L'administrateur a le pouvoir d'ajouter, créer, modifier des éléments de la base de donnée tel que des matières définissant le type de question existantes, des questions, des professeurs, des assistants.

- Modifier un joueur

l'administrateur a la possibilité d'intervenir sur le compte utilisateur d'un joueur et d'en modifier des caractéristiques.

- Attribuer

l'administrateur a la possibilité d'attribuer les questions aux matières correspondantes et vice-versa. Il peut aussi attribuer des matières a un professeur ou l'inverse.

b) Joueur

- Jouer

Le joueur peut lancer l'application et choisir soit de jouer contre un serveur, soit de jouer contre un autre joueur.

Pour cela le joueur devra avoir créé un compte via l'interface de création de compte.

- Créer un compte

Cela inclus de passer par l'interface de création de compte afin de se connecter ou de s'ajouter en tant qu'utilisateur de l'application et donc de se créer un compte.

3) Scénarios

a) Joueur contre Joueur

- Si le joueur souhaite jouer contre un autre joueur, il devra attendre la connexion de son adversaire.
- Le mode de jeu "joueur contre joueur" inclus le fait de poser des questions à l'adversaire puis de répondre aux siennes. A travers ce jeu de questions réponses plusieurs événements peuvent se passer tel que perdre et donc perdre des points de classements.

- **gagne objet si gagne**

Si le joueur gagne le "combat" contre son adversaire, celui-ci peut récupérer certains objets pouvant l'aider à avancer dans le jeu.

- **gagne points de classement si gagne**

Si le joueur gagne le "combat" contre son adversaire, il va alors gagner des points de classement.

- **perd points de classement si perd**

Si le joueur perd le "combat" contre son adversaire, il va alors perdre des points de classement.

- **pose une question**

Durant un "combat" contre un adversaire, chaque joueur pose à tour de rôle une question qu'il puise dans la liste de questions à disposition.

- **sélectionner une question parmi celles gagnées**

Lorsque un joueur pose une question, il en choisit une parmi celle gagnée dans le mode histoire.

b) joueur contre serveur

- le mode de jeu "joueur contre serveur" inclus le fait d'avancer dans l'aventure qui inclus de combattre des professeurs/assistants.
- un combat contre un professeur/assistant inclus de répondre aux questions de "l'ennemi" si on répond juste on passe à la question suivante.
- Si un joueur "tue" un ennemi, il gagne un niveau et passe à l'ennemi suivant.
- Si un joueur tombe à zéro points de vie, il perd et meurt. Il doit donc recommencer le combat.
- Si un joueur réussit à combattre un professeur/assistant il gagne toutes les questions répondues justes et les ajoute à son "pool" de questions. Ces questions lui seront utiles en mode joueur contre joueur.

- **combattre une professeur ou un assistant**

Durant le mode histoire "contre serveur" le joueur va devoir d'attaquer à des assistants puis les professeur. Celui-ci devra les battre afin de passer a l'étape suivante.

- **prochain niveau si gagne**

Si un joueur combat un assistant ou un professeur il peut alors passer au niveau suivant lui proposant alors un nouvel adversaire.

- **gagne expérience si gagne**

Si un joueur gagne dans un combat en mode histoire, il va alors gagner en retour des points d'expérience.

- **recupère question si gagne**

Si un joueur gagne dans un combat en mode histoire, Les questions répondues juste sont alors ajoutées a son pool de questions qu'il peut poser dans le mode joueur contre joueur.

c) Général

- **perdre et mourir**

Lorsque que le joueur tombe a zéro point de vie, alors il meurt et devra recommencer le niveau.

- **Répondre a une question**

Répondre au question inclus que si on répond faux, on perd des points de vie. Si on tombe à zéro on perd le combat.

Si un joueur est en difficulté pour répondre à une question il peut choisir d'utiliser un Objet qui va l'aider en modifiant de comportement du jeux et en lui facilitant la tâche (dépend du joker utilisé).

- **perdre des points de vie**

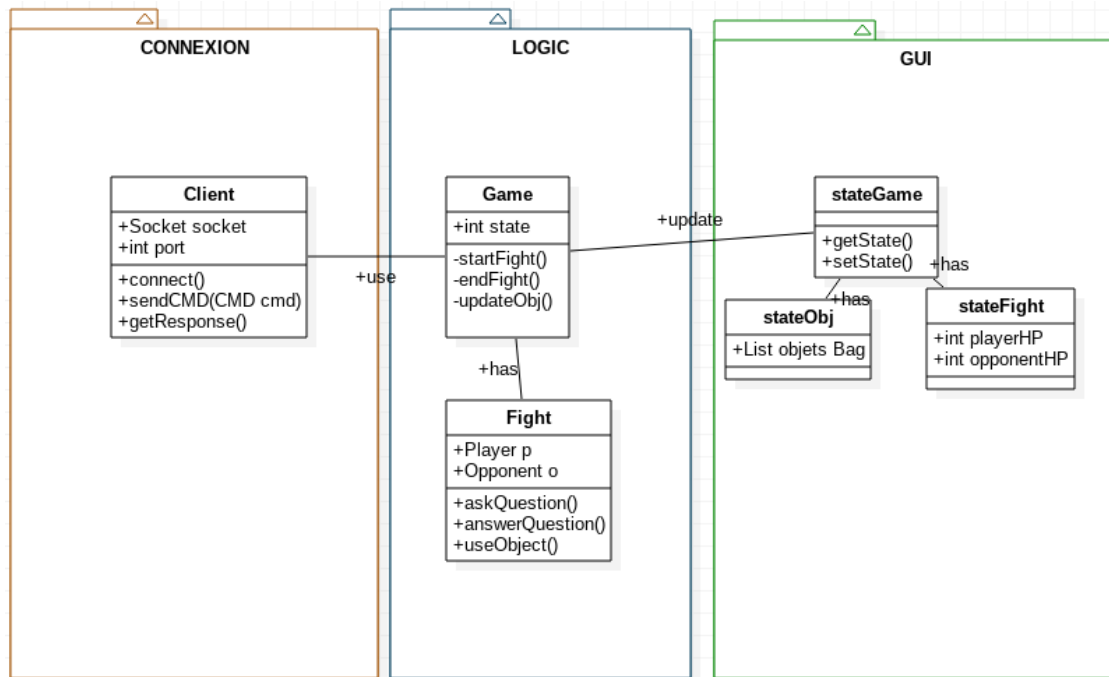
Lorsque le joueur répond faux a une question il perd alors des points de vie.

- **utiliser un objet**

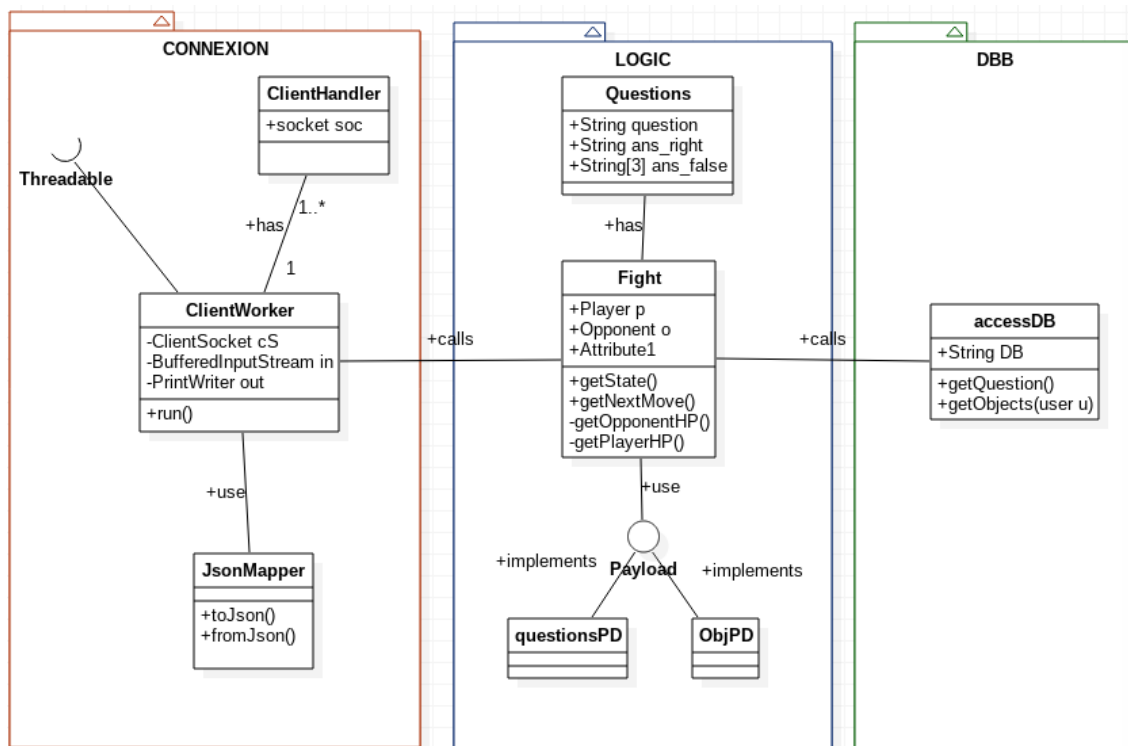
Durant les combats, un joueur peut utiliser un objet se trouvant a ça disposition dans son inventaire. Cet objet peut être de type différent l'aidant alors a répondre a la question ou a re-gagner des points de vie.

III. Ébauches du modèle de domaine

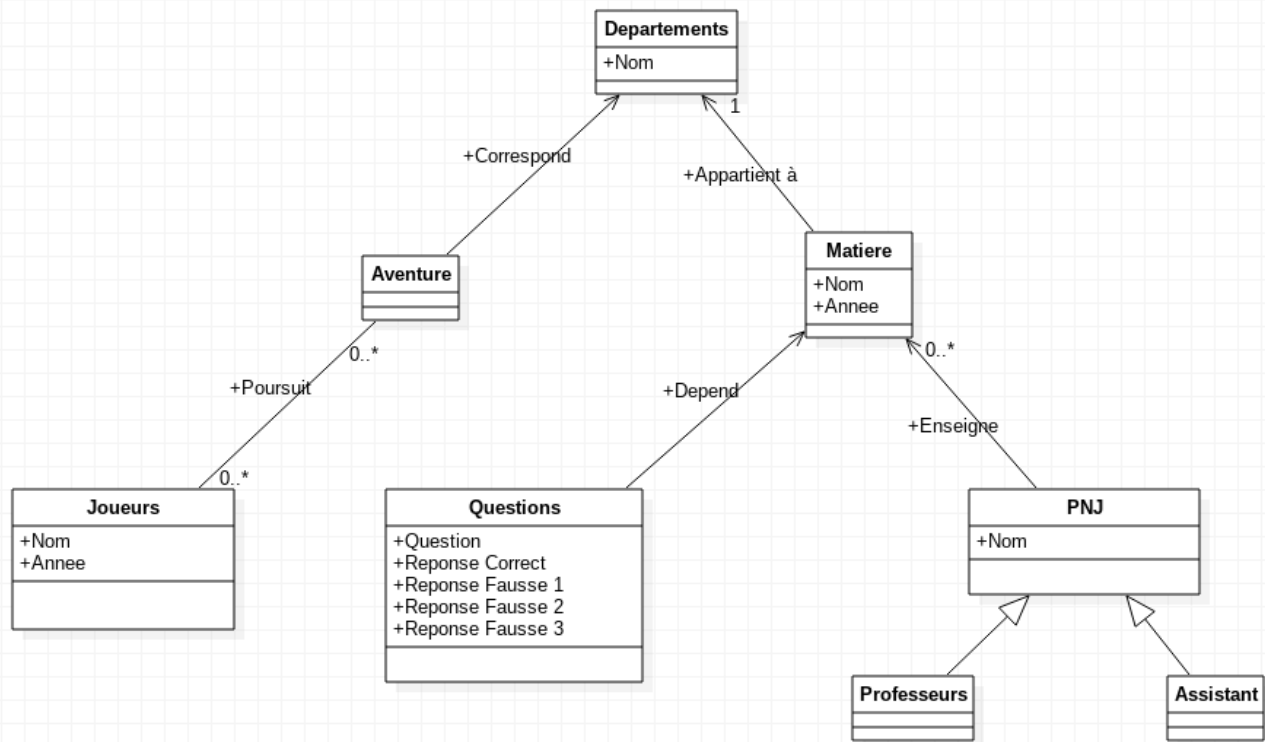
1) Modèle client



2) Modèle serveur



IV. Base de données

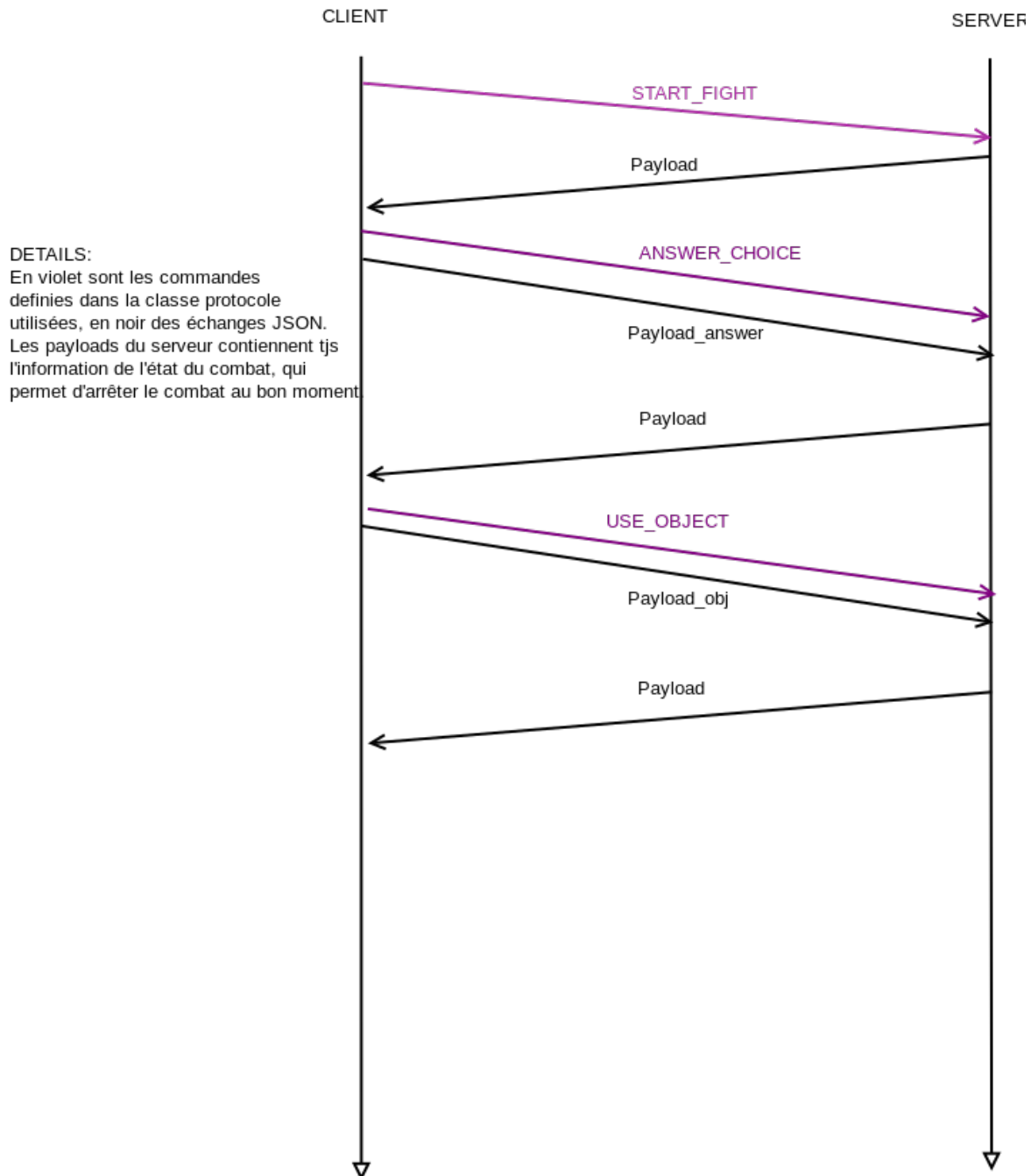


V. Serveur et Protocole d'échange

1) Serveur

- stocke l'avancement de chaque joueur dans le jeu
- gère les interactions des joueurs qui jouent contre l'ordinateur
- gère les échanges entre joueurs qui jouent contre d'autres joueurs
- stocke les questions dans une base de donnée des questions disponibles
- gère les ordinateurs (profs / assistants), choix des questions de chaque

2) Protocole



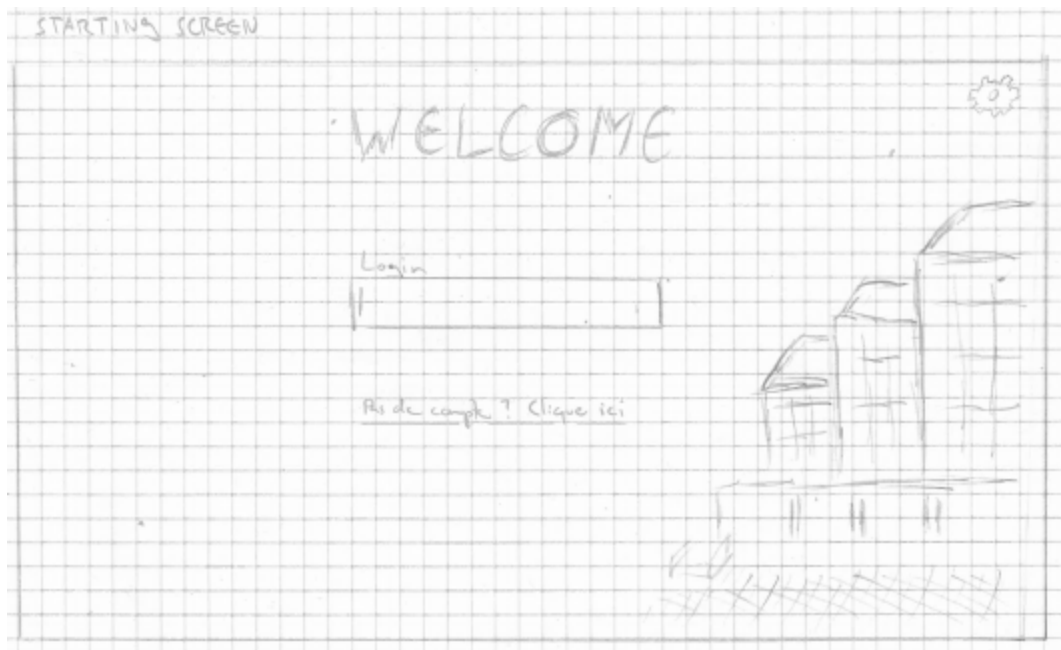
Exemple de payload

```

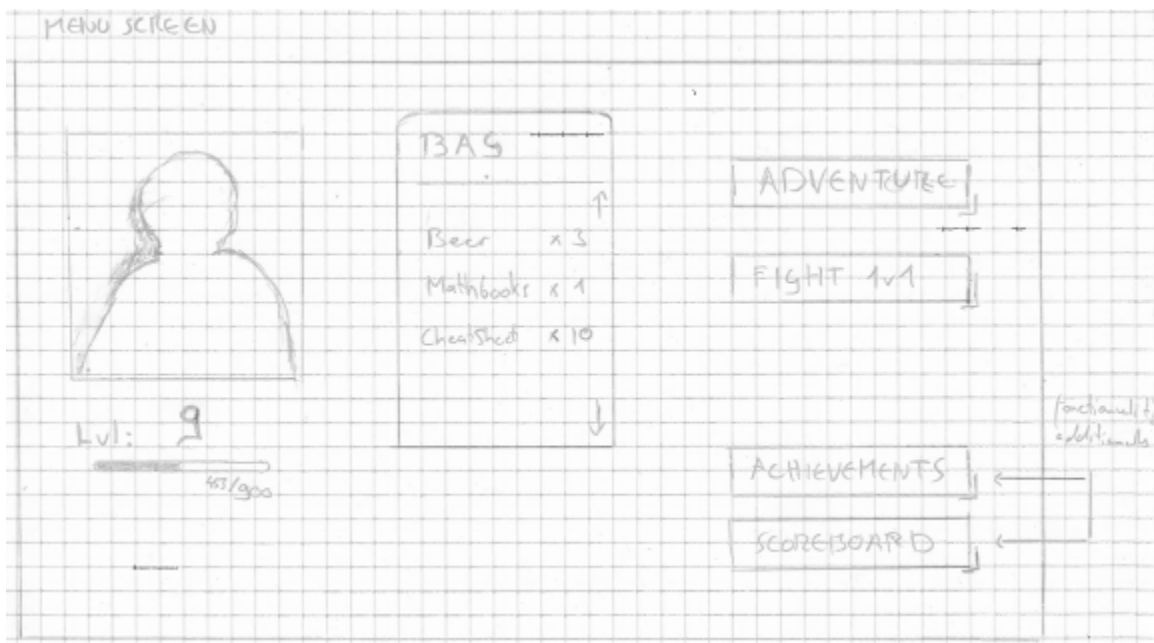
{"payload":
  {"question": "..",
   "ans_right": "..",
   "ans_false": ["1": "..", "2": "..", "3": ".."],
   "opponentHP": 45,
   "playerHP": 80
  }
  "state": "WIN|LOSE|FIGHTING"
}
  
```

VI. Ébauches interfaces utilisateur

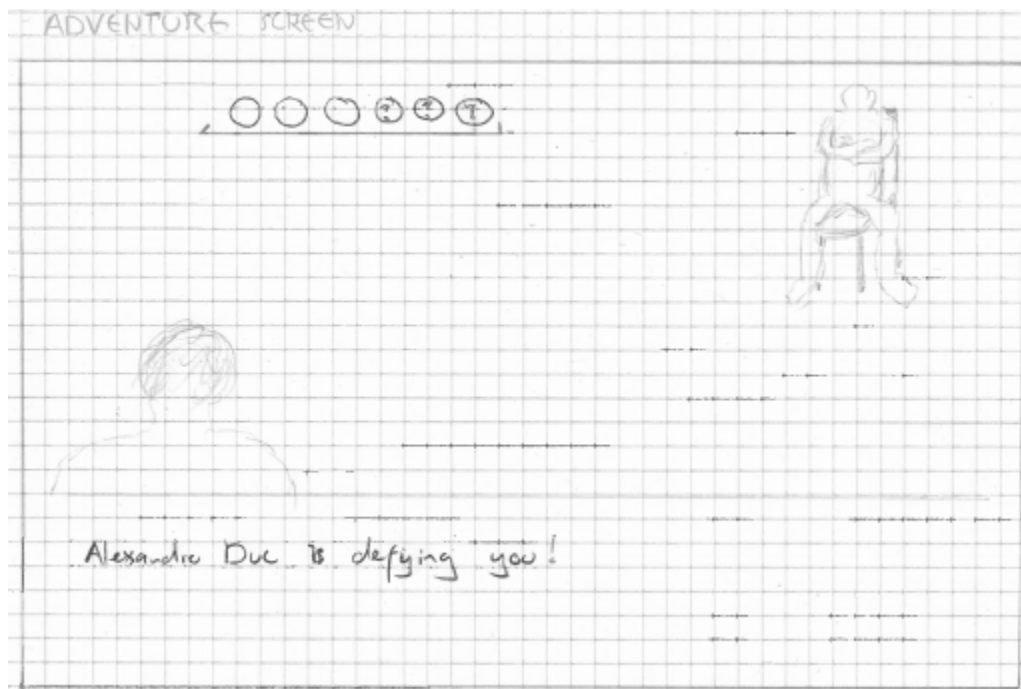
1) Starting Screen



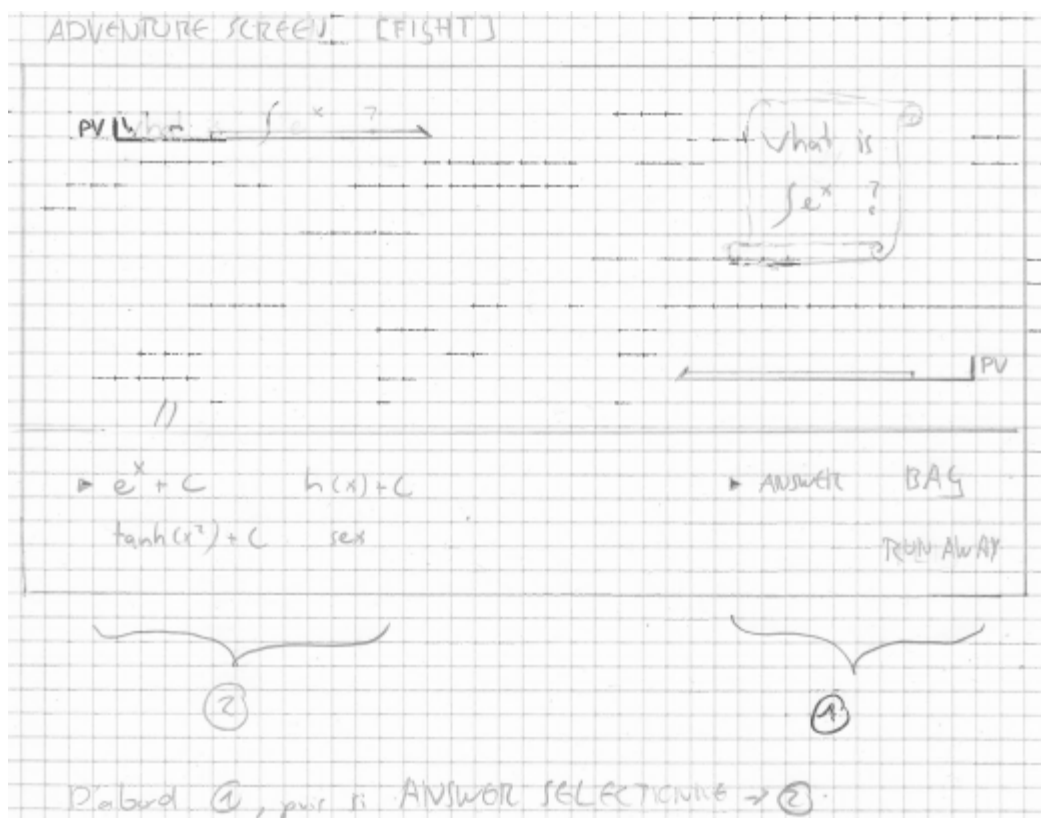
2) Menu Screen



3) Adventure Screen



4) Adventure Screen [Fight]



VII. Gestion de projet

1) Rôle

- Scrum master : Yann
- Product Owner : Joël
- Developper: Yohann, Loïc

2) Backlog



3) Plan d'itération

Tous nos sprints durent 1 semaine.

Drunk&Smart - Todo

Sprint 1 - Todo

26/04 > 02/05 10 \$

Plan



1 ★ 5 \$
Se connecter au serveur

4 2

3 ★ 5 \$
Combattre un autre étudiant.

4 1

Sprint 2 - Todo

03/05 > 09/05 10 \$

Plan



5 ★ 5 \$
Combattre un professeur

1

2 ★ 5 \$
Poser des questions.

2

Sprint 3 - Todo

10/05 > 16/05 10 \$

4 ★ 5 \$
choisir un étudiant spécifique

2

8 ★ 5 \$
Créer un compte, se connecter

1

Drunk&Smart - Todo

Sprint 4 - Todo

17/05 > 23/05 10 \$

Plan



7 ★ 5 \$
Utiliser l'interface afin de se créer un compte ou se...

1

9 ★ 5 \$
Utiliser l'interface pour combattre un profeseur.

1

Sprint 5 - Todo

24/05 > 30/05 10 \$

Plan



10 ★ 5 \$
Utiliser l'interface pour combattre un autre étudiant.

1

11 ★ 5 \$
Jouer un scénario d'avanture

1

Sprint 6 - Todo

31/05 > 06/06 5 \$

12 ★ 5 \$
Jouer un scénario Etudiant VS Etudiant

1

Drunk&Smart - Todo

Sprint 5 - Todo

24/05 > 30/05 10 \$

Plan



10 ★ 5 \$
Utiliser l'interface pour combattre un autre étudiant.

1

11 ★ 5 \$
Jouer un scénario d'avanture

1

Sprint 6 - Todo

31/05 > 06/06 5 \$

Plan



12 ★ 5 \$
Jouer un scénario Etudiant VS Etudiant

1

Sprint 7 - Todo

07/06 > 13/06 5 \$

13 ★ 5 \$
Administrer l'application

1

VIII. Planification détaillée des sprints

1) Sprint numéro 1

★ 1	Se connecter au serveur	0	0	4	2	...
5 \$		0/4		Planned		
1 FL	Création Executable client	0	0	+	...	
2 FL	Creation Executable serveur	0	0	+	...	
3 YM 2 ⌚	Mise en place protocol communication client-serveur	0	0	+	...	
4 YM 3 ⌚	Mise en place comunication entre deux client avec intermédiaire serveur	0	0	+	...	

S

★ 3	Combattre un autre étudiant.	0	0	4	1	...
5 \$		0/4		Planned		
5 YL 1 ⌚	Création d'objets Etudiant sur serveur et sur client	0	0	x	...	
6 JS 1 ⌚	Création méthode perte de vie	0	0	+	...	
7 JS 1 ⌚	Création méthode gain d'expérience	0	0	+	...	
8 YL 1 ⌚	Création méthode attaquer (répondre question)	0	0	x	...	

2) Sprint numéro 2

★ 5	Combattre un professeur	0	0	2	1	...
5 \$		0/2		Planned		
13 YM 1 ⌚	Création des classe professeurs	0	0	+	...	
14 YM 1 ⌚	Créations des méthodes professeurs	0	0	+	...	
★ 2	Poser des questions.	0	0	2	2	...
5 \$		0/2		Planned		
11 YL 1 ⌚	Creation Base de donnée	0	0	x	...	
12 YL 4 ⌚	implémentation base de donnée	0	0	x	...	

★ 14	Mise au propre et correction du code	0 0 2 0	...
2 \$		0/2	Planned
9 FL	correction du code	0 0	...
1			
10 JS	Mise au propre du code	0 0	...
1			

Sprint numéro 3

★ 4	choisir un étudiant spécifique	0 0 6 2	...
5 \$		0/6	Planned
15 YL	Créer des classes d'étudiants spécifiques	0 0	...
1			
16 FL	Ajouter les items	0 0	...
1			
17 YL	Choix d'étudiant spécifique	0 0	...
1			
18 YL	Gagner des items mode histoire	0 0	...
1			
19 YL	Gagner des items en mode versus	0 0	...
1			
22 YL	Utiliser les items	0 0	...
1			

★ 8	Créer un compte, se connecter	0 0 2 1	...
5 \$		0/2	Planned
20 FL	Se connecter	0 0	...
2			
21 JS	Creer un nouveau Player	0 0	...
3			

Sprint numéro 4

★ 7 5 \$	Utiliser l'interface afin de se créer un compte ou se connecter.	0 0 4 1 ...
		0/4 Planned
23 1 ⌚	Créer et mettre en place les bases d'une interface javaFX	0 0 0 ...
24 1 ⌚	Créer les fenêtres nécessaires pour la connexion de player	0 0 0 ...
25 1 ⌚	Créer les fenêtres nécessaires pour la création de player	0 0 0 ...
26 2 ⌚	mapper le code déjà fait pour les sprints précédents aux fenêtres créées	0 0 0 ...
★ 9 5 \$	Utiliser l'interface pour combattre un professeur.	0 0 2 1 ...
		0/2 Planned
27 2 ⌚	Créer les fenêtres nécessaires pour un combat entre professeur.	0 0 0 ...
28 3 ⌚	mapper le code déjà fait pour les sprints précédents aux interfaces.	0 0 0 ...

Sprint numéro 5

★ 9 5 \$	Utiliser l'interface pour combattre un professeur.	0 0 2 1 ...
		0/2 Planned
27 2 ⌚	Créer les fenêtres nécessaires pour un combat entre professeur.	0 0 0 ...
28 3 ⌚	mapper le code déjà fait pour les sprints précédents aux interfaces.	0 0 0 ...
★ 11 5 \$	Jouer un scénario d'aventure	0 0 3 1 ...
		0/3 Planned
29 1 ⌚	enchaîner des combats.	0 0 0 ...
30 3 ⌚	liste d'ennemis	0 0 0 ...
31 1 ⌚	Vérifier que tout fonctionne.	0 0 0 ...

Sprint numéro 6

★ 12	Jouer un scénario Etudiant VS Etudiant	0/2	Planned
5 \$			
32	Gérer le préparation d'un combat	0/0	Planned
3			
33	Vérifier que les données sont correctes	0/0	Planned
2			
★ 10	Utiliser l'interface pour combattre un autre étudiant.	0/4	Planned
5 \$			
34	Interface graphique Versus (attente)	0/0	Planned
1			
35	Interface graphique Challenge	0/0	Planned
1			
36	Interface graphique combat	0/0	Planned
2			
37	Photo personnages	0/0	Planned
1			
★ 15	Début de refactoring	0/1	Planned
1			
38	Revue totale du code	0/0	Planned
3			

Sprint numéro 7

★ 13	Administrer l'application	0/3	Planned
5 \$			
39	Gérer le numéro de port et host	0/0	Planned
1			
40	Gérer l'ajout de questions	0/0	Planned
2			
41	gérer le pattern du mode histoire	0/0	Planned
2			
★ 16	Correction des bugs existants	0/2	Planned
5 \$			
42	Correction des bugs existants	0/0	Planned
1			
43	propreté du code	0/0	Planned
1			

★ 12	Jouer un scénario Etudiant VS Etudiant	0/2	Planned
1 \$			
32	Gérer le préparation d'un combat	0/2	Planned
1 \$			
33	Vérifier que les données sont correctes	0/2	Planned
1 \$			
New task			
★ 10	Utiliser l'interface pour combattre un autre étudiant.	0/2	Planned
1 \$			
35	Interface graphique Challenge	0/2	Planned
1 \$			

IX. Bilan Sprint

1) Sprint numéro 1

- Histoire id(1) et id(3) planifiées, Les deux histoire ont été terminées
 - velocity: on avait prévu 10 pts d'historie partagés en différentes tâches. Tous les histoire ont été effectuées.
 - Aucun Replanification des tâches effectuées n'est a prévoir. Mais nous ajoutons une histoire pour la réorganisation du code et corrections.
- Nous avons pas prévus la complexité du code pour réussir le sprint, du coup nous n'avons pas programmé de la manière la plus propre. Notre code nécessite un remaniement et des corrections.
 - Le sprint concerné par cela est le 2.
 -

Sprint 2 - Todo
 03/05 > 09/05 12 \$

Plan

5 ★ 5 \$ Combattre un professeur ✓ 1	2 ★ 5 \$ Poser des questions. ✓ 2	14 ★ 2 \$ Mise au propre et correction du code 2
--	---	--

- d) commanditaires généraux: Le sprint a été compliqué, nous avons mal estimé la difficulté de la communication server-client. Cela nous a donc posé plusieurs problèmes et prit plus de temps que prévu.
- e) Nous devrions mieux gérer la difficulté des tâches et travailler sur l'anticipation du travail ainsi que les tests.

2) Bilans personnels Sprint numéro 1

- Yann Lederrey
 - temps prévu versus temps réalisé : le temps prévu était de 2 heures environ pour des tâches prévues simple qui se sont avérées plus compliquées. Au final j'ai pris plutôt 3-4 heures
 - commentaire personnel : Nous avons mal anticipé la conceptualisation du code. Nous aurions dû passer plus de temps sur une feuille et sur des schémas au lieu de partir dans le code, ce qui nous aurait fait gagner du temps.
- Joel Schär
 - temps prévu versus temps réalisé : heures passées : 6 heures, heures prévues 4 heures
 - commentaire personnel : La base n'étant pas encore faite, il a fallu faire tout le travail de conceptualisation du code en très peu de temps. Nous avons donc pris une première direction, que nous avons dû corriger et repenser, afin d'obtenir une structure utilisable. Ces revirements nous ont fait perdre du temps et nous ont compliqué le lancement du projet.
Dans la même idée nous, n'avons pas pris le temps avant le début du premier sprint de mettre sur papier l'entièreté des protocoles de communication entre les clients et le serveur. Nous avons donc implémenté quelque chose d'un peu bancal pour commencer. Du fait que cette structure n'a pas été choisie et validée il est possible qu'elle change encore dans le futur et cela va encore nous faire perdre du temps.
Je pense que nous avons pour le moment réussi à fonder une base, mais que celle-ci nécessite encore un peu de travail de consolidation, afin d'avoir une bonne base de travail sur laquelle construire la suite des fonctionnalités.
- Yohann Mayer
 - temps prévu versus temps réalisé : Imprévu du à baleinev.
 - commentaire personnel : A cause de baleinev je n'ai pas pu avancer sur le projet durant ce sprint. Ceci a été accepté par les autres.
- Loïc Frueh

- temps prévu versus temps réalisé : Heures prévue: 3h (je pensais facile)
Heures passées: 12h (Ah en faite non...)
- commentaire personnel : Ce premier sprint m'a un peu surpris car une tache qui me semblait simple en apparence cachait en faite une complexité non soupçonnée. La Gestion de la connexion au serveur était relativement facile mais la gestion de la mise en communication d'un client vers un autre était quelque chose de totalement nouveau pour moi et on a du à plusieurs reprises réviser le code voir remanier sa structure entière. Mais sinon, j'ai beaucoup appris de ce sprint. Dans l'ensemble je dirais intense mais instructif.

3) Sprint numéro 2

- a) Histoire id(5), id(2) et id(14) planifiées, Les histoires ont été terminées
- b) vitesse: on avait prévu 12 pts d'histoire partagés en différentes tâches. Tous les histoire ont été effectuées.
- c) Aucune Replanification des tâches effectuées n'est a prévoir. Par contre nous avons ajouté au programme l'affichage des points de vies des joueur et adversaires. Ceci n'était prévu dans aucune histoire et a été ajouté pour des raisons de confort de jeu.
 1. Afin que cela soit cohérent j'ai modifié l'histoire d'ID 2 afin que cela soit y spécifié.
 2. Le sprint concerné est le 2.
 - 3.

★ 5	Combattre un professeur
5 \$	
★ 2	Poser des questions et affiche points de vies
5 \$	
★ 14	Mise au propre et correction du code
2 \$	

- d) Commentaires généraux: Ce sprint nous a demandé énormément de temps de debug afin de corriger, améliorer et ajouter le système de question durant un combat multi-étudiants. Le mode histoire a été programmé mais mérite des amélioration durant l'histoire "Jouer un scénario d'aventure". Il nous reste potentiellement un problème de concurrence à corriger.

- e) Notre démarche de travail était bonne. C'est la partie "debug" du code qui nous a demandé énormément de temps afin de comprendre où se trouvaient nos soucis à travers les différents Threads.

4) Bilans personnels Sprint numéro 2

- Yann Lederrey : Temps prévu 5 heures : temps réalisé ~6h30.

J'ai passé beaucoup d'heures sur le debug des soucis dans la communication du mode multiPlayer. Ceci est notre problème récurrent. Je pense qu'il faut que l'on essaye de rendre notre code plus clair/simple afin de diminuer le nombre d'heures de debug. Sinon la communication du groupe est toujours bonne et nous avançons correctement.

- Yohann Meyer : Temps prévu 5 heures : temps réalisé 4 heures.

J'ai découvert dans ce sprint l'essentiel du travail réalisé par mes collègues lors du sprint présent et me suis familiarisé avec toutes les idées implémentées.

Après avoir développé une version du combat contre professeur, je me suis orienté sur la refactorisation du code.

- Loïc Frueh : Temps prévu 4 heures, temps réalisé 12 heures.

Une fois n'est pas coutume, ce deuxième sprint a de nouveau été l'occasion de constater qu'une tâche que l'on pense rapide et relativement simple peut prendre des proportions immenses dès qu'un petit bug entre dans l'équation... Le code a été vite écrit, le débogage par contre lui a pris une éternité ! Je pense sans exagérer que sur les 12h effective de travail sur la tâche, 9h ont été consacrées à déboguer le programme. Ce sprint m'a montré qu'une application client->serveur<-client est très difficile à déboguer.

- Joël Schär : Temps prévu 2-3 heures, temps réalisé 2 heures.

Lors de ce sprint je me suis concentré principalement sur la remise au propre et la relecture du code.

Je commence avec ce second sprint à voir l'efficacité d'avancement d'un projet avec la méthode scrum. Avec des petits objectifs à court terme on perd rapidement de vue l'objectif final et on se concentre sur la tâche en court. Chaque fin de sprint avec une solution fonctionnelle est comme une petite fin de projet. et une petite victoire.

5) Sprint numéro 3

- a) Histoire id(4), id(8) planifiées, Les histoires ont été terminées
- b) vitesse: on avait prévu 10 pts d'histoire partagés en différentes tâches. Tous les histoires ont été effectuées.

- c) Aucune Replanification des tâches effectuées n'est à prévoir. Le sprint prochain sera assez conséquent et nous n'auront pas beaucoup de temps. Nous avons malgré cela fait le choix que garder le sprint tel quel afin de nous motiver. Nous replanifierons après coup si nous n'avons pas réussi à le terminer.
- d) Commentaires généraux: Ce sprint c'est assez bien passé, Nous avons travaillé de manière coordonné. Aucun bug bloquant ont été rencontrés (pour une fois...). Nous avons par contre des difficultés dans le partage correcte du travail. Sans pour autant impacter sur le projet car les personnes qui sont moins impliquées dans ce projet durant un sprint, nous soulage sur le travail d'autre projets.
- e) Nous avons fait une discussion pour revoir la manière dont nous partageons le travail. Par exemple Yohann Meyer, ayant eu des empêchements au premier sprint. Il a prit un retard sur la compréhension du code préalablement écrit et ce retard c'est augmenté au fur et à mesure. Il nous a aidé indirectement malgré tout.

Nous avons décidé de laisser le prochain sprint comme il est à cause de la surcharge de travail de cette semaine. Pour le sprint 5, nous allons discuter avec Yohann Meyer et d'autres si besoin, afin de rattraper le niveau du projet pour que tout le monde puisse être correctement impliqué.

De même que pour les tâches que nous respectons rarement. Le plus souvent on travaille de la manière suivante : Si on a le temps, on avance. Même si ça signifie faire des tâches encore incomplètes et appartenant à un autre membre du groupe.

6) Bilans personnels Sprint numéro 3

- Yann Lederrey : Temps prévu 5 heures : temps réalisé 9h00.

Ce sprint c'est plutôt bien passé, j'ai rapidement et "facilement" avancé sur le travail à faire. J'ai soulagé mes collègues pris par d'autres travaux et avancant sur ce sprint. J'ai surtout travaillé sur la structure général du programme, la création de compte et de player ainsi que les bases pour les items.

- Yohann Meyer : Temps prévu 0 heures: temps réalisé 1 heures.

Après une profonde inspection du code et une introspection de même, je me suis rendu compte que l'apport possible sur le code de ma part ne serait que marginal au mieux et négatif au pire. J'ai donc pris le parti de commenter le sprint et de me concentrer sur l'obtention d'une vision générale pointue.

- Loic Frueh : Temps prévu 4 heures, temps réalisé 0 heures.

Pour ce sprint, je n'ai pas fait la moindre ligne de code.

J'ai pris une petite semaine de vacances en ce qui concerne le projet en tout cas.

- Joel Schär : 3 heures, temps réalisé 7 heures.

Cette semaine j'ai travaillé sur l'implémentation des items pour les joueurs.

Je me suis rendu compte que le code que j'avais écrit il y a que quelques jours, je ne savais plus vraiment ce qu'il faisait et comment il fonctionnait.

On remarque alors que sur une architecture un peu complexe, il est très facile de se perdre. La raison pour laquelle nous arrivons dans cette situation, est que nous devons travailler rapidement pour arriver à terminer les sprints et ne prenons donc pas beaucoup de temps à rendre le code claire. Les étapes de refactorings du code qui devrait être fait après l'implémentation d'une fonctionnalité se révéleraient très utiles au final.

Pour le reste je pense que nos sprints sont efficaces et que nous arrivons au résultat escompté à chaque fois, ce qui prouve que notre méthode de travail reste efficace.

5) Sprint numéro 4

- a) Histoire id(7), id(9) planifiées, Seulement l'histoire 7 : Utiliser l'interface afin de se créer un compte ou se connecter a été terminée.
- b) vitesse: on avait prévu 10 pts d'historique partagés en différentes tâches. Seulement 5 points d'historique ont été effectués.
- c) Nous avons prévu une re planification sur les deux prochains sprint pour rattraper notre retard.
 - 1) Nous déplaçons l'historique (9) : Utiliser l'interface pour combattre des professeurs au Sprint 5 Afin de ne pas accumuler de retard. Pour nous laisser la possibilité de faire ce rattrapage nous avons déplacé l'historique (10) Utiliser l'interface pour combattre un autre étudiant au Sprint 6 (initialement au Sprint 5).
 - 2) Les sprint concernés sont le 4,5 et 6.
 - 3)

Sprint 5 - Todo
24/05 > 30/05 10 \$

Stars	Task	Progress	Status
★ 11 5 \$	Jouer un scénario d'aventure	0/3	Planned
★ 9 5 \$	Utiliser l'interface pour combattre un professeur.	0/2	Planned

Sprint 6 - Todo
31/05 > 06/06 10 \$

Stars	Task	Progress	Status
★ 12 5 \$	Jouer un scénario Etudiant VS Etudiant	0/1	Planned
★ 10 5 \$	Utiliser l'interface pour combattre un autre étudiant.	0/1	Planned

- d) Commentaires généraux: Malheureusement à cause d'une énorme source de travail et la possibilité de replanifier notre projet, nous avons fait le choix de mettre le projet en deuxième priorité durant ce sprint. Nous savons que nous arriverons à rattraper ce retard. Nous avons tout de même fait un minimum pour limiter le retard.
- e) Une discussion est prévue avec le groupe pour parler des problèmes émis au bilan du sprint précédent.

6) Bilans personnels Sprint numéro 4

- Yann Lederrey : Temps prévu 5 heures : temps réalisé 6h00.

Ce sprint étant au mauvais moment, J'ai proposé à mes collègues de se concentrer sur leurs problèmes et je me suis chargé de même en place la base de JavaFX pour notre projet. J'ai donc été le seul à travailler sur le projet durant ce Sprint. Cela ne m'a pas posé de problème car cela venait d'un accord commun.

- Yohann Meyer : Temps prévu 0 heures: temps réalisé 0 heures.

Vacance pour cause de surcharge de travail.

- Loic Frueh : Temps prévu 0 heures, temps réalisé 0 heures.

Vacance pour cause de surcharge de travail.

- Joel Schär : 0heures, temps réalisé 0heures.

Vacance pour cause de surcharge de travail.

7) Sprint numéro 5

- a) Histoire id(11), id(9) planifiées et terminées
- b) vitesse: on avait prévu 10 pts d'histoire partagés en différentes tâches. l'entiéreté a été effectué.
- c) Malgré le fait que nous avons correctement fait ce sprint nous sentons le besoin d'ajouter une histoire technique pour le sprint 6 et une plus grosse pour le 7 qui sera du refactoring.
 - 1) Nous ajoutons donc l'histoire technique refactoring au sprint 6 puis au sprint 7.
 - 2) Les sprint concerné sont le 6 et 7
 - 3)

Sprint 6 - Todo		Plan			
31/05 > 06/06 11 \$					
★ 12 5 \$	Jouer un scénario Etudiant VS Etudiant	0	0	0	1 Planned
★ 10 5 \$	Utiliser l'interface pour combattre un autre étudiant.	0	0	0	1 Planned
★ 15 1 1 \$	Début de refactoring	0	0	0	0 Planned

Sprint 7 - Todo		Plan			
07/06 > 13/06 10 \$					
★ 13 5 \$	Administrer l'application	0	0	0	1 Planned
★ 16 5 \$	Correction des bugs existants	0	0	0	0 Planned

- d) Commentaires généraux: Tout c'est bien passé dans ce sprint. Nous avons revus notre structure et nos algorithmes sur notre serveur afin de convernir à l'utilisation des interface mais cela c'est fait correctement. Yohann Meyer et revenu dans la partie et s'investit à fond surtout de par son envie de faire un gros coup de nettoyage du code. Nous avons perdu du temps sur la revue du serveur de par notre manque d'expérience en JavaFx et notre manque d'anticipation lors de la programmation du serveur.
- e) Pas de problème à signaler.

8) Bilans personnels Sprint numéro 5

- Yann Lederrey : Temps prévu 3 heures : temps réalisé 6h30.

Ce sprint c'est bien passé pour ma part j'ai bien remotivé les troupes sur le projet et nous avons réussis à retravailler le serveur pour arriver à utiliser correctement notre interface graphique.

- Yohann Meyer : Temps prévu 4 heures: temps réalisé 5 heures.

Un démonstration des limites de la méthode agile appliquée à une team de developper junior.
Un travail conséquent de refactoring nous attends, mais je suis satisfait d'avoir pu comprendre l'entiereté de la situation actuelle.

- Loic Frueh : Temps prévu 4 heures, temps réalisé 4 heures.

Ce sprint à été pour moi l'occasion d'apprendre que parfois il vaut mieux directement commencer avec la modélisation et l'implémentation de l'interface graphique avant de commencer à coder la communication client serveur ou encore les différente méthode de combat entre joueurs.

Le plus gros du travail à été de transformer notre code existant afin qu'il puisse s'adapter à la réalité qu'est JavaFX.

Sinon rien de spécial, pas de trop grosse surprise pour cette fois mis à part le point précédemment cité.

- Joel Schär : 2 heures, temps réalisé 3 heures.



















Dans ce sprint nous avons remarqué le problème que peut poser la méthode scrum lorsque l'on a peu d'expérience. En effet nous avons commencé par aller dans une direction avec comme objectif de réaliser les histoires des premiers sprints. Mais n'ayant pas l'expérience suffisante pour directement se diriger dans la bonne directions nous rencontrons constamment des difficultés liée à une modélisation qui ne correspond pas à notre projet. Dans le cas de ce sprint c'est le passage d'un schéma synchrone vers un schéma asynchrone nécessaire à l'utilisation d'une librairie graphique qui nous à mis des bâtons dans les roues. Nous avons donc du nous diriger vers une sémantique événementielle et casser les blocs séquentiels au maximum.

9) Sprint numéro 6

- a) Histoire id(12), id(10) planifiées mais pas terminée, seulement l'histoire d'id (15) a été terminée.
- b) velocity: on avait prévu 11 pts d'histoire partagés en différentes tâches. Seulement 1 point d'histoire a été terminé.
- c) Vu que les histoires étaient presque terminées nous avons choisis de juste les ajouter au dernier sprint.
 - 1) Nous avons donc ajouté ces histoire au sprint 7.

2) Les sprint concerné est le 7.

3)

 Sprint 7 - Todo 07/06 > 13/06 20 \$		Plan			
★ 13	Administrer l'application	 0	 0	 0	 1 ...
5 \$		Planned			
★ 16	Correction des bugs existants	 0	 0	 0	 0 ...
5 \$		Planned			
★ 12	Jouer un scénario Etudiant VS Etudiant	 0	 0	 2	 1 ...
1 \$		0/2		Planned	
★ 10	Utiliser l'interface pour combattre un autre étudiant.	 0	 0	 4	 1 ...
1 \$		0/4		Planned	

- d) Commentaires généraux: Nous avons du faire face a un gros bug pour implémenter graphiquement le combat entre les étudiants et nous n'avons pas réussi a corriger cela à temps. Nous sommes optimiste sur le fait de corriger cela avant la présentation. Sinon le refactoring a bien commencé et certaines micro-fonctionnalités ajoutées.
- e) Pas me problème dans la gestion du groupe ou du travail à signaler.

10) Bilans personnels Sprint numéro 6

- Yann Lederrey : Temps prévu 5 heures : temps réalisé 10h30.
 - Personnellement ce sprint a été intéressant au début pour moi car je me suis “amusé” sur l'interface graphique mais le bug bloquant arrivé par la suite m'a prit beaucoup d'énergie et de temps. Surtout que malheureusement cela n'a pas pu être terminé.
- Loic Frueh : Temps prévu 5 heures, temps réalisé 5 heures
 - Un sprint un peu plus decevant que les autres en ce qui me concerne, je n'ai pas réussi à implémenter ma tâche dans le temps imparti mais je garde espoir de les resoudre lors du sprint en cours. Pas grand chose à

dire de plus si ce n'est que j'ai pu constater l'importance du refactoring quand il s'agit de debugger un bout de code par la suite.

- Joel Schär : 2 heures, temps réalisé 3 heures.
 - Avancement positif et réjouissant des visuels de l'interface graphique. Nous restons cependant confronté à des éléments bloquant dans l'implémentation des parties multi joueurs.
On voit de plus en plus le besoin pressant de faire un refactoring du code afin d'y voir claire.
- Yohann Meyer : Temps prévu 4 heures: temps réalisé 5 heures.
 - Refactorisation et précision de l'interface graphique, habile ajout de classe abstraite et progression vers le produit fini: un sprint qui garde en tête la globalité du projet. Un petit roadbump sur la trajectoire prévue dans le passage terminal-GUI, une conséquence de la technique agile lorsque l'on dispose d'une jeune team de développement. Au final, une vision positive de la progression générale.