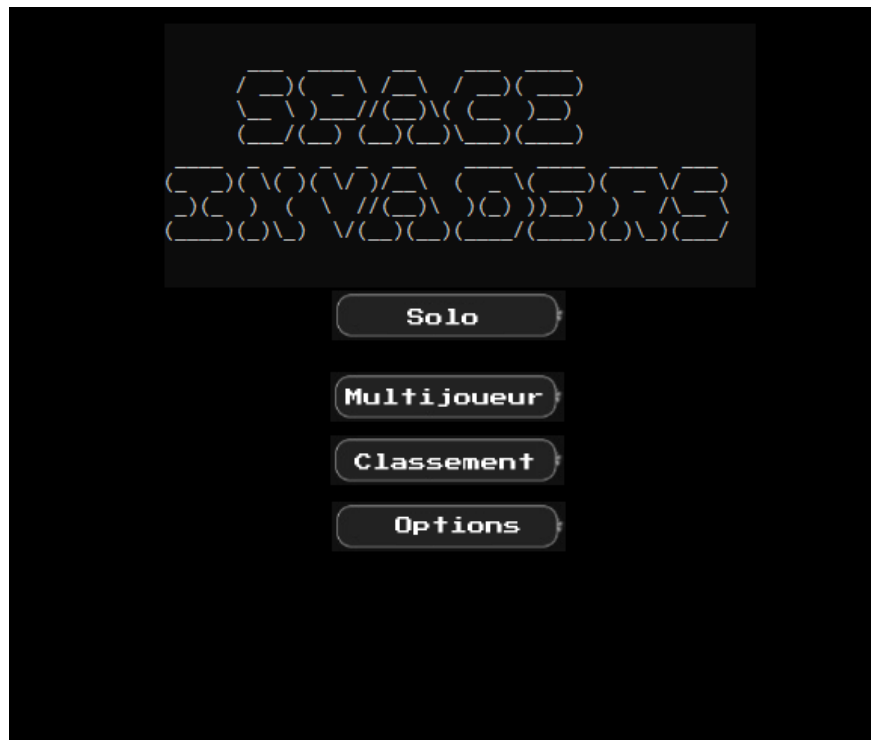


# Projet SpicyInvaders



## Table des matières

7.3.2 Spécificités UX .....	1
1. Contexte.....	1
2. Analyse.....	1
1. Persona .....	1
2. Palette graphique .....	2
3. Eco-conception .....	3
4. Accessibilité .....	3
3. Conception.....	4
Définition des écrans .....	4
Choix effectués .....	4
4. Évaluation .....	6
Test d'utilisabilité.....	6
Conclusion Technique .....	8
Conclusion Personnelle.....	8
7.3.3 Spécificités POO .....	9
Introduction .....	9
Analyse fonctionnelle .....	9
Analyse technique.....	9
Diagramme de base.....	9
Explications (docfx).....	9
Test Unitaire .....	10
Conclusion .....	11
7.3.4 Spécificités DB .....	12
A. Importer les données et le schéma de base de données .....	12
B. Gestion des utilisateurs .....	13
1. Administrateur du jeu.....	13
2. Joueur .....	14
3. Gestionnaire de la boutique .....	15
4. Comment implémenter cela.....	16
C. Réaliser et expliquer en détail les requêtes.....	17
Requête n°1 : .....	17
Requête n°2 : .....	18
Requête n°3 : .....	19
Requête n°4 : .....	20
Requête n°5 : .....	21
Requête n°6 : .....	22
Requête n°7 : .....	23
Requête n°8 : .....	24
Requête n°9 : .....	25
Requête n°10 : .....	26
D. Création des index .....	27
1. Pourtant certains index existent déjà. Pourquoi ? .....	27
2. Quels sont les avantages et les inconvénients des index ? .....	27
3. Sur quel champ (de quelle table), cela pourrait être pertinent d'ajouter un index ? .....	27
E. Backup / Restore .....	28
F. Connexion DB à C# .....	29
8.2 Utilisation d'IA dans le projet.....	31
Partie POO .....	31
Partie DB .....	31
Partie UX .....	32
Autre .....	32

## 7.3.2 Spécificités UX

### 1. Contexte

Le projet « SpicyInvaders » amène à la création d'un prototype cliquable du fait de la spécification UX. La maquette doit donc comprendre un choix entre jouer en multijoueur ou en solo, le choix d'avoir une palette graphique différentes, le choix d'avoir un design différent pour les ennemis, une page contenant les meilleures scores et un choix du gameplay.

### 2. Analyse

#### 1. Persona

##### *a. Contexte*

La création des persona se fait en premier afin de savoir comment créer les maquettes par la suite. Un petit nombre de persona est premièrement créé, 2, d'autre peuvent être créé ultérieurement dans le cas ou d'autre caractéristiques importante à mentionner et à analyser devrait être découverte.

Pour la création de mes Persona j'ai utilisé le Template de Jingdi Ma :

<https://www.figma.com/community/file/994974628735380661>

##### *b. Données récoltés*

Je récupérerai l'âge, le genre, le domicile, le travail et le statut familial en premier.

Je créerai par la suite une biographie de mon persona à l'aide de ChatGPT.

À l'aide de graphique j'estime à quel point mon persona apprécie certains aspects du jeu tel que :

- Faire les meilleures scores que possibles
- Atteindre le plus haut niveau possible
- Explorer des modes de jeu alternatif
- Faire des défis (de la communauté)
- Passer le temps

Je cherche ce que souhaite et ne souhaite pas rencontrer mon persona sur un jeu de Space Invaders. J'établie un pourcentage des appareils utilisés pour jouer à Space Invaders afin de savoir comment adapter mon jeu.

##### *c. Constat grâce à mes Persona*

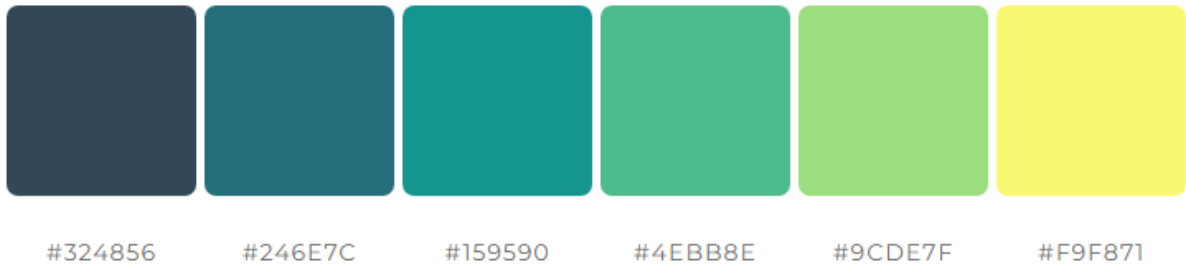
Mon premier persona, Robbie Heineman me fait prendre conscience que le premier jeu Space Invaders étant sorti en 1978 il peut s'adresser à des personnes actuellement âgé et qu'il vaut donc mieux ne pas créer trop d'effet spéciaux pour ne pas déranger l'utilisateur. De plus les jeux Space Invaders étant sorti sur arcade il pourrait être utile d'avoir des contrôles imitant les bornes d'arcade. D'anciens joueurs pourraient aussi souhaiter avoir une version original du jeu.

Mon second persona, Martine Dupont, me fait prendre conscience que mes utilisateurs pourraient jouer à Space Invaders depuis leur téléphone. Ils pourraient vouloir jouer à plusieurs en local comme en multijoueur. De nouveau joueur pourrait souhaiter avoir facilement accès à des modifications sur leur partie, des niveaux afin qu'une communauté puisse proposer des ajouts sur le jeu, il pourrait donc être utile que le jeu soit ouvert à la création de mod afin d'améliorer la re-jouabilité.

## 2. Palette graphique

Pour ma palette de couleur je souhaitais créer un contraste entre l'espace et l'humanité afin de réellement créer une impression de lutte pour l'humanité. Ainsi ma palette de couleur commence avec des couleurs plutôt sombre caractérisant l'espace puis commence un dégradé montrant premièrement du bleu caractérisant la Terre ensuite du vert caractérisant la verdure de la Terre puis finalement du jaune caractérisant les humains.

### Generic Gradient



J'ai aussi créé une seconde palette de couleur destiné au daltonien ayant des contrastes de couleurs plus distingué.

### Collective



### 3. Eco-conception

Dans le cadre de l'éco-conception de ce projet :

- ☑ Je me suis tenu au minimum demandé et je n'ai pas rajouté pléthore de fonctionnalités que le client n'a pas demandé.
- ☑ Je n'ai pas créé de Carrousels d'images.
- ☑ J'ai créé des titres de pages pertinents pour que l'utilisateur n'ait pas à charger un plus grand nombre de page pour trouver ce qu'il veut.
- ☑ J'ai favorisé un design simple.
- ☑ Mes pages sont statiques et n'ont pas d'animation inutile.

Afin de savoir sur quel point m'améliorer j'ai pris connaissances des 115 bonnes pratiques d'Eco-conception web ([https://collectif.greenit.fr/ecoconception-web/115-bonnes-pratiques-eco-conception\\_web.html](https://collectif.greenit.fr/ecoconception-web/115-bonnes-pratiques-eco-conception_web.html)), j'en ai retenu une petite dizaine applicable sur des schémas haute-fidélité et je les ai appliqués sur mon schéma.

### 4. Accessibilité

Afin de rendre ma maquette haute-fidélité accessible au plus grand nombre de personne que possible j'ai créé un mode blanc, avec une autre palette de couleur, dans le cas où le joueur aurait des problèmes d'acuité visuelle j'ai aussi créé des textes suffisamment grands, un contraste élevé entre les dits textes et le fond. J'ai également ajouté le choix des sprites pour les ennemis et le joueur

Ma maquette est également structurée de façon à être utilisable par un clavier lors de la navigation et non une souris. Ainsi ma maquette est légèrement plus accessible en cas d'handicap moteur.

Je n'ai rien conçu en cas d'handicap auditif.

Pour les handicaps mentaux j'ai juste évité les animations afin d'empêcher les distractions.

### 3. Conception

#### Définition des écrans

La taille de mes maquettes est en 800x600.

Dans mes maquettes j'ai comme page :

- Un menu principal
- Une page de tableau des scores
- Un page de jeu
- Une page d'option
- Une page de choix de design

Dans mon menu option je peux choisir d'activer le mode blanc

#### Choix effectués

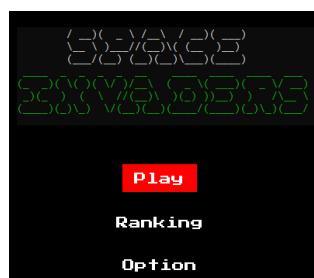
Deux maquettes ont été créées.

La première est un prototype représentant le jeu qui est créé en programmation C# console. Étant donné mon niveau technique et l'utilisation de C# console cette première maquette est plus sobre et comporte ainsi ce qui est demandé dans le cahier des charges de façon à la créer en application console.

La seconde maquette est plus détaillée et ne devra par conséquent pas être implémenter. Elle peut donc comporter n'importe quel élément de n'importe quel complexité.

J'ai ainsi un prototype pratique, qui sera créé en application console et un prototype théorique qui peut être complexe.

Lors de la création de ces maquettes j'ai décidé de commencer en créant un menu d'accueil capable de diriger le joueur ou il le souhaite. Ainsi mon menu pratique comprend les options « Play » pour lancer une partie, « Ranking » pour afficher le classement et « Option » pour accéder aux options. Mon menu théorique remplace le bouton « Play » par deux autres boutons « Solo » et « Multijoueur ». En effet mon menu pratique n'a pas ce choix car il n'y aura pas d'option multijoueur.



1 - Menu d'accueil (pratique)

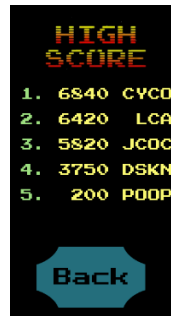


2 - Menu d'accueil (théorique)

Le menu des scores est semblable dans les deux versions. Il contient seulement les 5 meilleures scores avec le nom des joueurs.

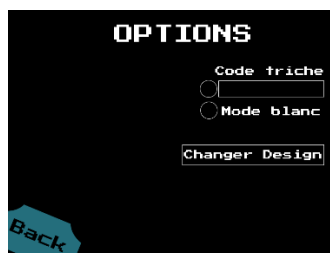


3 - Menu des scores (pratique)



4 - Menu des scores (théorique)

J'ai choisi de mettre une page de choix des Sprite dans la page option de mon prototype théorique. La possibilité d'activer le mode blanc changeant la palette de couleur est dans les options



5 - menu option (théorique)



6 - menu choix design (théorique)

## OPTIONS

Code triche  
  
☒ Mode blanc  
 Changer Design



7 - menu option mode blanc (théorique)

## 4. Évaluation

### Test d'utilisabilité

Afin de tester l'utilisabilité de mon produit j'ai commencé en vérifiant tout les liens de mon menu. Je n'ai pas effectué tout les tests d'utilisabilité possible.

Afin d'effectuer mes test d'utilisabilité j'ai créé un tableau figma dont voici une partie :

Menu	Description	Pre-Condition	Test Steps	Test data	Expected Output	Actual Output	Accept	Fail
home screen	Pouvoir accéder à l'écran de jeu solo	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton "solo"	None	Arriver sur le jeu	.....		
home screen	Pouvoir accéder à l'écran de classement	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton "Classement"	None	Arriver sur le menu de classement	.....		

*8 – Partie de mes test d'utilisabilité fait sur figma*

Voici une partie de mes tests d'utilisabilité :

Menu	Description	PreCondition	TestSteps	TestData	ExpectedOutput	Actual Output	Accept	Fail
Home screen	Pouvoir accéder à l'écran de jeu solo	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton "solo"	None	Arriver sur le jeu			
Home screen	Pouvoir accéder à l'écran de classement	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton "solo"	None	Arriver sur le menu de classement			
Home screen	Pouvoir accéder aux options	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton « Option »	None	Arriver dans les options			
Home screen	Pouvoir accéder à l'écran de jeu multi-joueur	Avoir l'application Être dans le menu principal	1. Démarrer l'application 2. Appuyer sur le bouton « Multijoueur »	None	Arriver sur le jeu			
Highscore menu	Voir les 5 meilleures joueurs du score le plus haut	Avoir l'application Être dans le menu de score	1. Démarrer l'application 2. Aller dans le menu de score	None	Voir les 5 meilleures joueurs			
Highscore menu	Revenir au menu d'accueil	Avoir l'application Être dans le menu de score	1. Démarrer l'application 2. Aller dans le menu de score 3. Appuyer sur le bouton Retour	None	Arriver sur le menu principal			
Option	Activer le mode blanc	Avoir l'application Être dans le menu option	1. Démarrer l'application 2. Aller dans les options 3. Appuyer sur le bouton Mode Blanc	None	Le fond devient blanc			



Option	Aller dans le menu de choix des designs	Avoir l'application Être dans le menu option	1.Démarrer l'application 2.Aller dans les options 3.Appuyer sur le bouton Changer Design	None	Le menu de design s'ouvre			
Option	Revenir à l'écran d'accueil	Avoir l'application Être dans le menu option	1.Démarrer l'application 2.Aller dans les options 3.Appuyer sur le bouton retour	None	Arriver sur le menu principal			
Design	Changer le design de l'ennemi	Avoir l'application Être dans le menu Design	1.Démarrer l'application 2.Aller dans le menu Design 3.Appuyer sur le joueur qui n'est pas encadré	None	L'autre ennemi devient encadré			
Design	Changer le design du joueur	Avoir l'application Être dans le menu Design	1.Démarrer l'application 2.Aller dans le menu Design 3.Appuyer sur le joueur qui n'est pas encadré	None	L'autre joueur devient encadré			
Option	Revenir aux options	Avoir l'application Être dans le menu option en mode blanc	1.Démarrer l'application 2.Aller dans le menu Design 3.Appuyer sur le bouton retour	None	Arriver sur le menu option			
Option	Revenir à l'écran d'accueil en mode blanc	Avoir l'application Être dans le menu option en mode blanc	1.Démarrer l'application 2.Aller dans les options 3.Activer le mode blanc 4.Appuyer sur le bouton Retour	None	Arriver sur le menu d'accueil en mode blanc			
White home screen	Pouvoir accéder à l'écran de jeu solo en mode blanc	Avoir l'application Être dans le menu principal en mode blanc	1.Démarrer l'application 2.Aller dans les options 3.Activer le mode blanc 4.Appuyer sur le bouton Retour 5.Appuyer sur solo	None	Arriver sur le jeu en mode blanc			
White home screen	Pouvoir accéder à l'écran de jeu multijoueur en mode blanc	Avoir l'application Être dans le menu principal en mode blanc	1.Démarrer l'application 2.Aller dans les options 3.Activer le mode blanc 4.Appuyer sur le bouton retour 5.Appuyer sur Ranking	None	Arriver sur le jeu (multijoueur) en mode blanc			
White home screen	Pouvoir accéder à l'écran de classement en mode blanc	Avoir l'application Être dans le menu principal en mode blanc	1.Démarrer l'application 2.Aller dans les options 3.Activer le mode blanc 4.Appuyer sur le bouton Retour 5.Appuyer sur Ranking	None	Arriver sur le classement en mode blanc			

## Conclusion Technique

Pour récapituler j'ai :

- Créer des personas dès le début ce qui m'a permis de savoir exactement ce que je comptais faire par la suite.
- Créer une maquette haute-fidélité pour ce que j'ai implémenté en POO.
- Créer une maquette haute-fidélité que je n'ai pas implémenté par faute de connaissances et de temps.
- Créer des tests d'utilisabilité afin de tester ma maquette haute-fidélité non-implémenté.

## Conclusion Personnelle

Lors de ce projet je n'ai pas eu de difficulté technique, j'ai pu apprendre à maîtriser Figma et j'ai surtout pu me lancer dans un projet qui m'a impliqué de m'ordonner dans mon travail.

Par manque d'expérience au début du projet je n'ai pas tout de suite tenu IceScrum à jour et par conséquent je n'ai pas une très bonne explication de ce que j'ai fait dans le cadre UX du projet. Je n'ai pas non plus puis suivi le temps passé sur certaine tâche. Même sans savoir le temps que j'ai passé sur certaine tâche je pense quand même ne pas avoir été très efficace dans le partage de ma charge de travail. La création de mes maquettes m'a pris beaucoup de temps alors que j'en ai passé beaucoup moins sur la création et l'analyse de mes personas.

Je pense que ce projet m'aura surtout appris une méthodologie de travail dans un projet de grande envergure, pour une seule personne et dans un court laps de temps, ainsi lors de futur projet j'espère être plus à même de la gestion de mon temps et de mon efficacité.

## 7.3.3 Spécificités POO

### Introduction

Dans le cadre d'apprentissage de l'ETML nous sommes amenés à effectuer un projet de programmation orienté objet. Dans ce projet je dois créer une réplique du célèbre jeu Space Invaders en version console ou Forms. Pour ma part j'ai choisi de créer mon projet en version console.

Les objectifs de ce projet sont :

- Créer un vaisseau capable de se déplacer et de tirer.
- Créer 10 ennemies se déplaçant sur l'axe vertical.

Pour la gestion du projet c'est IceScrum qui est utilisé. Toutes les Stories doivent être validés par le chef de projet avant de commencer à implémenter ce que décrit la Story. Chaque Story contient un/des tests d'acceptance pouvant être accompagné d'une maquette. Tout le projet se fait en un seul sprint.

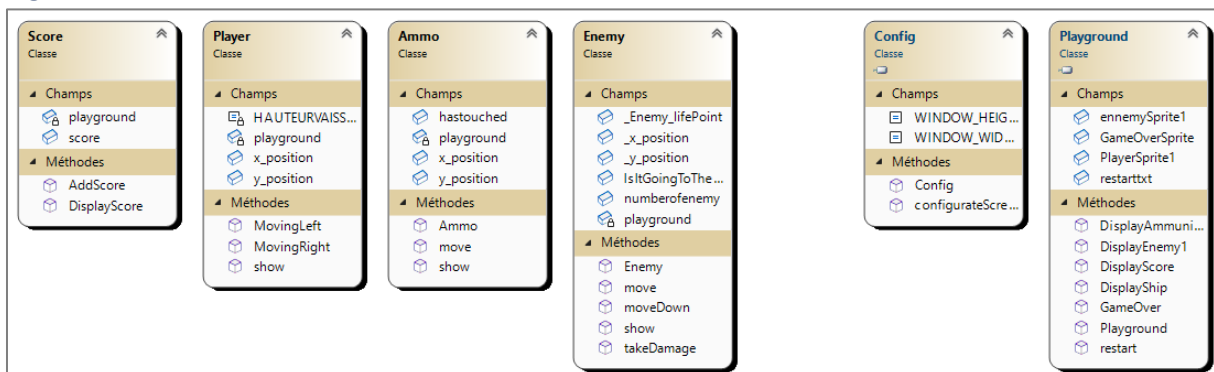
Le projet doit contenir des tests unitaires afin de tester la fonctionnalité de chaque fonction.

### Analyse fonctionnelle

Mon analyse fonctionnel a été généré automatiquement par IceScrub et se situe dans « doc/Analyse\_fonctionnel ».

### Analyse technique

#### Diagramme de base



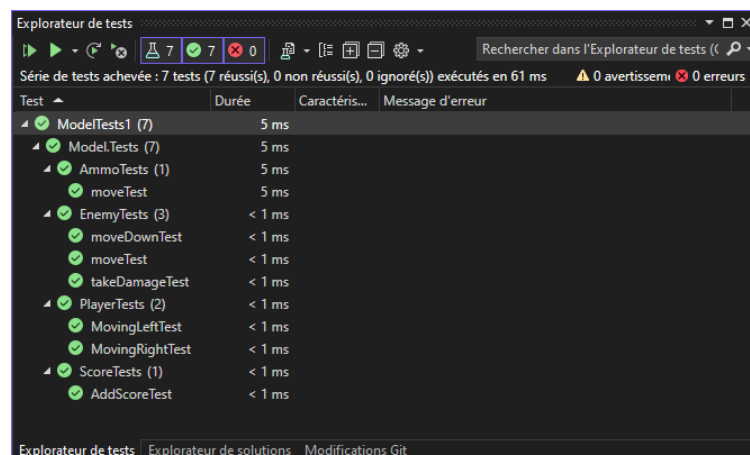
9 - Diagramme de classe « doc/Diagramme\_de\_classe »

#### Explications (docfx)

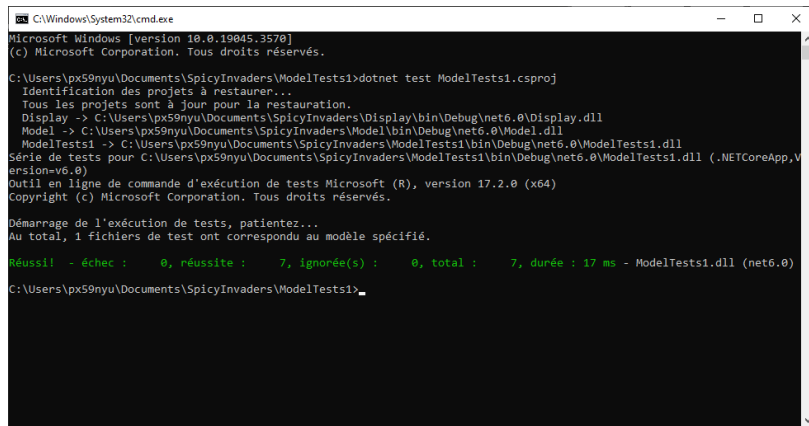
L'explication de mon code a été effectué grâce à docfx. Les fichiers sont dans le dossier « doc/Analyse\_Technique(docfx) » puis sont triés dans des dossiers selon la bibliothèque de classe.

## Test Unitaire

<u>Nom du test</u>	<u>Classe</u>	<u>Description</u>	<u>Condition réussite</u>
moveTest	Ammo	Test que la munition se déplace correctement	La position y de la balle est descendu de 1
moveTest	Enemy	Déplace l'ennemi de gauche à droite et de droite à gauche selon la direction	Test que la position de l'ennemi augmente bien de 1 s'il doit aller à droite et descende bien de 1 s'il doit aller à gauche
moveDownTest	Enemy	Déplace l'ennemi en bas et change la direction	Vérifie que la position y de l'ennemi à augmenter de 1 Vérifie que la direction à changer
takeDamageTest	Enemy	Inflige 1 dégât à l'ennemi	Vérifie que l'ennemi à perdu 1 point de vie
MovingLeftTest	Player	Déplace le joueur à gauche	Vérifie que la position x du joueur est descendu de 1 s'il n'a pas atteint la limite de gauche, sinon il ne doit pas s'être déplacé.
MovingRightTest	Player	Déplace le joueur à droite	Vérifie que la position x du joueur augmente de 1
AddScoreTest	Score	Augmente le score	Vérifie que le score à bien été ajouté de plus 10 multiplié par la vague actuelle qui est généré aléatoirement



### 10 – Test Visual Studio



```
C:\Windows\System32\cmd.exe
Microsoft Windows [version 10.0.19045.3570]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\px59nyu\Documents\SpicyInvaders\ModelTests1>dotnet test ModelTests1.csproj
  Identification des projets à restaurer...
  Tous les projets sont à jour pour la restauration.
  Display -> C:\Users\px59nyu\Documents\SpicyInvaders\Display\bin\Debug\net6.0\Display.dll
  Model -> C:\Users\px59nyu\Documents\SpicyInvaders\Model\bin\Debug\net6.0\Model.dll
  ModelTests1 -> C:\Users\px59nyu\Documents\SpicyInvaders\ModelTests1\bin\Debug\net6.0\ModelTests1.dll
  Série de tests pour C:\Users\px59nyu\Documents\SpicyInvaders\ModelTests1\bin\Debug\net6.0\ModelTests1.dll (.NETCoreApp,Version=v6.0)
  Outil en ligne de commande d'exécution de tests Microsoft (R), version 17.2.0 (x64)
  Copyright (c) Microsoft Corporation. Tous droits réservés.

  Démarrage de l'exécution de tests, patientez...
  Au total, 1 fichiers de test ont correspondu au modèle spécifié.

Réussi! - échec : 0, réussite : 7, ignorée(s) : 0, total : 7, durée : 17 ms - ModelTests1.dll (net6.0)

C:\Users\px59nyu\Documents\SpicyInvaders\ModelTests1>
```

## 11 - Test invite de commande

### Conclusion

Pour conclure, dans le cadre de ce projet j'ai appris des notions de base en programmation orienté objet et comment les utiliser en pratique. J'ai également appris à mieux utiliser Visual Studio 2022 que ça soit pour ajouter des bibliothèques de classe.

Les problèmes que j'ai rencontrés :

Beaucoup de temps passé à faire des stories, pas un problème en soit mais j'aurai dû faire une story « créer des user story » pour connaître le temps passé sur cette tâche, ainsi j'aurai pu évaluer l'efficacité d'IceScrum dans ce projet.

### 7.3.4 Spécificités DB

#### A. Importer les données et le schéma de base de données

Création de la base de données :

```
docker exec -i db mysql -uroot -proot < db_space_invaders.sql
```

## B. Gestion des utilisateurs

### 1. Administrateur du jeu

*Peut créer, lire, mettre à jour et supprimer (CRUD) n'importe quelle table.*

*Gérer les utilisateurs et leurs privilèges.*

```
CREATE ROLE `administrateur_du_jeu`;
```

```
GRANT CREATE, SELECT, UPDATE, DELETE ON *.* TO `administrateur_du_jeu`;
```



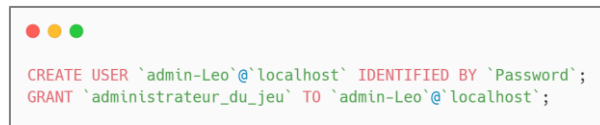
```
CREATE ROLE `administrateur_du_jeu`;  
GRANT CREATE, SELECT, UPDATE, DELETE ON *.* TO `administrateur_du_jeu`;
```

12 - Créer un rôle administrateur et lui donner des droits

Création de l'utilisateur dans le rôle 'Admin-jeu'.

```
CREATE USER 'admin-Leo'@'localhost' IDENTIFIED BY 'Password';
```

```
GRANT `administrateur_du_jeu` TO 'admin-Leo'@'localhost';
```



```
CREATE USER `admin-Leo`@`localhost` IDENTIFIED BY `Password`;  
GRANT `administrateur_du_jeu` TO `admin-Leo`@`localhost`;
```

13 - Créer un utilisateur et lui donner le rôle administrateur

### **Explication des requêtes :**

La première commande me permet de créer un rôle nommé « Admin-jeu ».

La seconde commande donne les privilèges CREATE, SELECT, UPDATE et DELETE sur toutes les tables de toutes les bases de données au rôle 'administrateur\_du\_jeu'.

La troisième commande crée un utilisateur nommé 'Admin-Leo' en localhost et lui donne le mot de passe 'Password'.

La quatrième commande donne le rôle 'administrateur\_du\_jeu' à l'utilisateur admin-Leo.

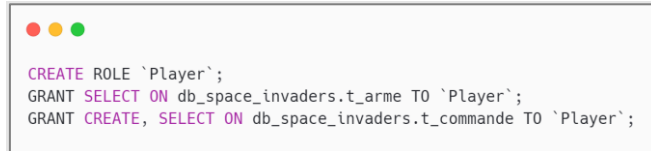
## 2. Joueur

*Lire les informations des armes (pour voir quelles armes il peut acheter).*

*Créer une commande.*

*Lire toutes les commandes.*

```
CREATE ROLE `Player`;  
  
GRANT SELECT ON db_space_invaders.t_arme TO `Player`;  
  
GRANT CREATE ON db_space_invaders.t_commande TO `Player`;  
  
GRANT SELECT ON db_space_invaders.t_commande TO `Player`;
```

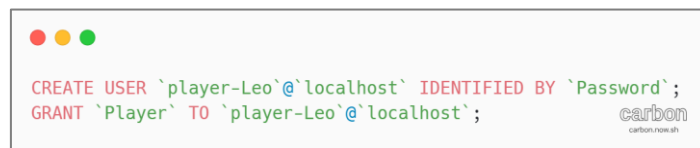


```
CREATE ROLE `Player`;  
GRANT SELECT ON db_space_invaders.t_arme TO `Player`;  
GRANT CREATE, SELECT ON db_space_invaders.t_commande TO `Player`;
```

14 - Créer un rôle joueur et lui donner des droits

Création de l'utilisateur dans le rôle 'Player' .

```
CREATE USER 'player-Leo'@'localhost' IDENTIFIED BY 'Password';  
  
GRANT 'Player' TO 'player-Leo'@'localhost';
```



```
CREATE USER `player-Leo`@`localhost` IDENTIFIED BY `Password`;  
GRANT `Player` TO `player-Leo`@`localhost`;
```

15 - Créer un utilisateur joueur et lui donner le rôle joueur

### **Explication des requêtes :**

La première commande crée le rôle 'Player'.

La seconde commande donne le droit SELECT au rôle 'Player' dans la table 't\_commande' de la base de données 'db\_space\_invaders'.

La troisième commande donne le droit CREATE au rôle 'Player' dans la table t\_commande de la base de données 'db\_space\_invaders'.

La quatrième commande donne le droit SELECT au rôle 'Player' sur la table 't\_commande' de la base de données 'db\_space\_invaders'.

La cinquième commande crée l'utilisateur 'player-Leo' avec l'Host 'localhost' et avec le mot de passe 'Password'.

La sixième commande ajoute l'utilisateur 'Player-Leo' au rôle 'Player'



### 3. Gestionnaire de la boutique

*Lire les informations sur tous les joueurs (pour savoir qui a passé une commande).*

*Mettre à jour, lire et supprimer des armes (ajout de nouvelles armes, modification des prix, etc.).*

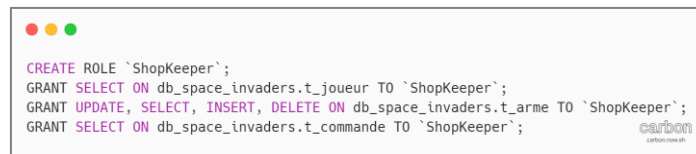
*Lire toutes les commandes*

```
CREATE ROLE `ShopKeeper`;
```

```
GRANT SELECT ON db_space_invaders.t_joueur TO `ShopKeeper`;
```

```
GRANT UPDATE, SELECT, INSERT, DELETE ON db_space_invaders.t_arme TO `ShopKeeper`;
```

```
GRANT SELECT ON db_space_invaders.t_commande TO `ShopKeeper`;
```



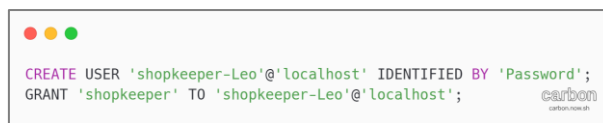
```
CREATE ROLE `ShopKeeper`;  
GRANT SELECT ON db_space_invaders.t_joueur TO `ShopKeeper`;  
GRANT UPDATE, SELECT, INSERT, DELETE ON db_space_invaders.t_arme TO `ShopKeeper`;  
GRANT SELECT ON db_space_invaders.t_commande TO `ShopKeeper`;
```

16 - Créer un rôle marchand et lui donner des droits

Création de l'utilisateur dans le rôle 'shopkeeper'.

```
CREATE USER 'shopkeeper-Leo'@'localhost' IDENTIFIED BY 'Password';
```

```
GRANT 'shopkeeper' TO 'shopkeeper-Leo'@'localhost';
```



```
CREATE USER 'shopkeeper-Leo'@'localhost' IDENTIFIED BY 'Password';  
GRANT 'shopkeeper' TO 'shopkeeper-Leo'@'localhost';
```

17 - Créer un utilisateur marchand et lui donner le rôle marchand

#### Explication des requêtes :

La première commande crée le rôle 'ShopKeeper'.

La seconde commande donne le droit 'SELECT' au rôle 'ShopKeeper' sur la table 't\_joueur' de la base de données 'db\_space\_invaders'.

La troisième commande donne les droits 'Update', 'SELECT', 'INSERT' et 'DELETE' au rôle 'ShopKeeper' sur la table 't\_arme' de la base de données 'db\_space\_invaders'.

La quatrième commande donne le droit 'SELECT' au rôle 'ShopKeeper' sur la table 't\_commande' de la base de données 'db\_space\_invaders'.

La cinquième commande crée l'utilisateur 'shopkeeper-Leo' avec l'host 'localhost' et le mot de passe 'Password'.

La sixième commande ajoute l'utilisateur 'shopkeeper-Leo' au rôle 'shopkeeper'.

#### 4. Comment implémenter cela

Comme vu sur les 3 points précédents l'implémentation de rôle pour la gestion des utilisateurs se fait ainsi.

Tout d'abord la création du rôle. Pour cela la commande `CREATE ROLE 'nom_du_role' ;` permet de créer un rôle.

Par la suite l'on peut donner le droit voulu au rôle grâce à la commande `GRANT` suivi du nom du droit voulu. Si l'on souhaite ajouter plusieurs droits l'on peut utiliser une virgule suivi du prochain droit. Après avoir spécifié le droit l'on écrit « `ON` » puis le nom de la base de données s'il y en a une de particulier suivi d'un point puis du nom de la table s'il y en a une de particulier. S'il n'y a aucune table ou base de données où le droit prendra effet en particulier alors on peut utiliser le caractère joker «`*`». Ainsi si le droit n'est pas spécifique à un lieu en particulier l'on peut très bien écrire « `*.*` » pour échapper à cette restriction. L'on spécifie par la suite à qui l'on donne ce droit avec un « `TO` » `'nom_du_role_ou_de_l'utilisateur' ;`.

Et en dernier il faut assigner des utilisateurs au droits créé. Pour cela l'on utilise à nouveau la commande « `GRANT` » suivi du nom du rôle puis de « `TO` » le nom de l'utilisateur «`@` » l'host de l'utilisateur «`;` ».

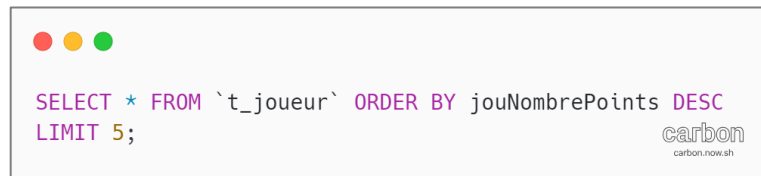
### C. Réaliser et expliquer en détail les requêtes

#### Requête n°1 :

*La première requête que l'on vous demande de réaliser est de sélectionner les 5 joueurs qui ont le meilleur score c'est-à-dire qui ont le nombre de points le plus élevé. Les joueurs doivent être classés dans l'ordre décroissant*

#### Réaliser la requête :

```
SELECT * FROM `t_joueur` ORDER BY jouNombrePoints DESC LIMIT 5;
```



18 - Requête 1

#### Explication des nouveaux éléments de la requête :

`SELECT * FROM `t_joueur`` : permet de sélectionner tous les attributs de la table `t\_joueur`.

`ORDER BY jouNombrePoints DESC` : le `DESC` permet d'afficher les valeurs dans l'ordre décroissant du `ORDER BY` selon les valeurs dans jouNombrePoints.

`LIMIT 5 ;` : me permet de n'afficher que les 5 premiers résultats et le `;` me permet de terminer la commande.

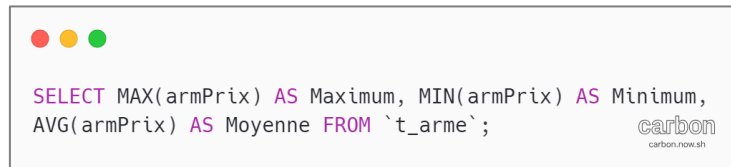
## Requête n°2 :

*Trouver le prix maximum, minimum et moyen des armes.*

*Les colonnes doivent avoir pour nom « Prix Maximum », « PrixMinimum » et « PrixMoyen »*

### Réaliser la requête :

```
SELECT MAX(armPrix) AS Maximum, MIN(armPrix) AS Minimum, AVG(armPrix) AS Moyenne FROM `t_arme`;
```



19 - Requête 2

### Explication des nouveaux éléments de la requête :

`SELECT Max(armPrix) AS Maximum` : permet d'afficher la plus grande valeur dans l'attribut armPrix. La virgule permet de sélectionner d'autre valeurs.

`MIN(armPrix) AS Minimum` : me permet d'afficher le plus petit résultat dans l'attribut armPrix. La virgule permet de sélectionner d'autre valeurs.

`AVG(armPrix) AS Moyenne FROM `t_arme` ;` : me permet d'afficher la moyenne des données dans l'attribut armPrix.

### Requête n°3 :

*Trouver le nombre total de commandes par joueur et trier du plus grand nombre au plus petit.*

*La 1<sup>ère</sup> colonne aura pour nom "IdJoueur", la 2<sup>ème</sup> colonne aura pour nom "NombreCommandes".*

#### Réaliser la requête :

```
SELECT DISTINCT `fkJoueur` AS IdJoueur, Count(`fkJoueur`) AS NombreCommandes  
FROM t_commande GROUP BY IdJoueur;
```



20 - Requête 3

#### Explication des nouveaux éléments de la requête :

`SELECT DISTINCT `fkJoueur` AS IdJoueur,` Permet d'afficher les valeurs de fkJoueur une seule fois par valeur.

`Count(`fkJoueur`) AS NombreCommandes` Permet de compter le nombre de valeur dans fkJoueur.

`FROM t_commande GROUP BY IdJoueur;` Permet de sélectionner la table t\_commande puis grouper par l'identifiant du joueur afin de calculer le nombre de commande par joueur et non pas au total.

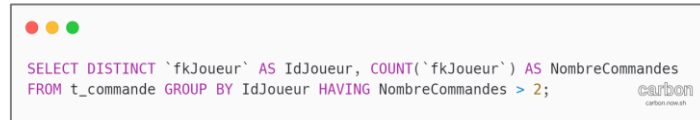
#### Requête n°4 :

*Trouver les joueurs qui ont passé plus de 2 commandes.*

*La 1<sup>ère</sup> colonne aura pour nom "IdJoueur", la 2<sup>ème</sup> colonne aura pour nom "NombreCommandes "*

#### Réaliser la requête :

```
SELECT DISTINCT `fkJoueur` AS IdJoueur, COUNT(`fkJoueur`) AS NombreCommandes  
FROM t_commande GROUP BY IdJoueur HAVING NombreCommandes > 2;
```



21 – Requête 4

#### Explication des nouveaux éléments de la requête :

Ceci est la même commande que celle de la requête 3 hormis l'utilisation de

```
HAVING NombreCommandes > 2
```

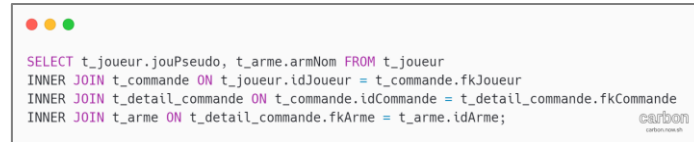
Avec ceci on ne sélectionne que les lignes qui ont un nombre de commande supérieur à 2.

### Requête n°5 :

*Trouver le pseudo du joueur et le nom de l'arme pour chaque commande.*

#### Réaliser la requête :

```
SELECT t_joueur.jouPseudo, t_arme.armNom FROM t_joueur INNER JOIN t_commande ON t_joueur.idJoueur = t_commande.fkJoueur INNER JOIN t_detail_commande ON t_commande.idCommande = t_detail_commande.fkCommande INNER JOIN t_arme ON t_detail_commande.fkArme = t_arme.idArme;
```



22 - Requête 5

#### Explication des nouveaux éléments de la requête :

`INNER JOIN t_commande ON t_joueur.idJoueur = t_commande.fkJoueur` Le `INNER JOIN` permet de créer une jointure entre la table `t_commande` et la table `t_joueur`. Cela permet de sélectionner des éléments sur plusieurs tables. Le `ON` par la suite permet de ne prendre **que** les valeurs sélectionnées auparavant ou la clé primaire de `t_joueur`(`idJoueur`)et la clé étrangère de `t_commande` sont équivalentes. La clé étrangère étant le référencement entre les 2 tables.

Dans cet exercice j'utilise un `INNER JOIN` pour ne récupérer que les valeurs reliés à d'autres tables.

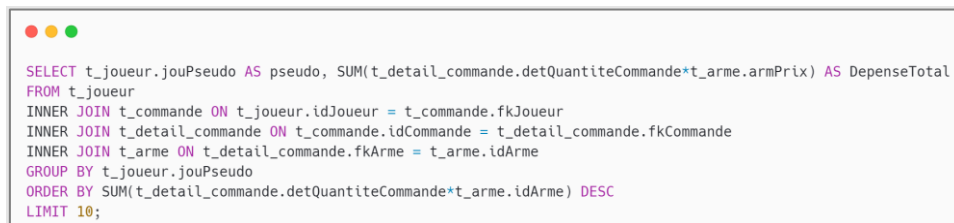
### Requête n°6 :

Trouver le total dépensé par chaque joueur en ordonnant par le montant le plus élevé en premier, et limiter aux 10 premiers joueurs.

La 1<sup>ère</sup> colonne doit avoir pour nom "IdJoueur" et la 2<sup>ème</sup> colonne "TotalDepense".

### Réaliser la requête :

```
SELECT t_joueur.jouPseudo AS pseudo, SUM(t_detail_commande.detQuantiteComma  
nde*t_arme.armPrix) AS DepenseTotal FROM t_joueur INNER JOIN t_commande ON  
t_joueur.idJoueur = t_commande.fkJoueur INNER JOIN t_detail_commande ON t_c  
ommande.idCommande = t_detail_commande.fkCommande INNER JOIN t_arme ON t_de  
tail_commande.fkArme = t_arme.idArme GROUP BY t_joueur.jouPseudo ORDER BY S  
UM(t_detail_commande.detQuantiteCommande*t_arme.idArme) DESC LIMIT 10;
```



```
SELECT t_joueur.jouPseudo AS pseudo, SUM(t_detail_commande.detQuantiteComma  
nde*t_arme.armPrix) AS DepenseTotal  
FROM t_joueur  
INNER JOIN t_commande ON t_joueur.idJoueur = t_commande.fkJoueur  
INNER JOIN t_detail_commande ON t_commande.idCommande = t_detail_commande.fkCommande  
INNER JOIN t_arme ON t_detail_commande.fkArme = t_arme.idArme  
GROUP BY t_joueur.jouPseudo  
ORDER BY SUM(t_detail_commande.detQuantiteCommande*t_arme.idArme) DESC  
LIMIT 10;
```

23 - Requête 6

### Explication de la commande :

Utilisation de 2 **INNER JOIN** afin de relier 2 tables qui sont séparées par une table. Dans ce cas la table arsenal est entre la table joueur et arme, elle contient les clefs primaires de la table joueur et de la table arme.

J'effectue une multiplication dans le SUM car une arme peut avoir été commandé 2 fois.



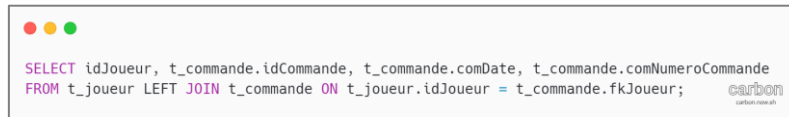
### Requête n°7 :

*Récupérez tous les joueurs et leurs commandes, même s'ils n'ont pas passé de commande.*

*Dans cet exemple, même si un joueur n'a jamais passé de commande, il sera quand même listé, avec des valeurs 'NULL' pour les champs de la table 't\_commande',*

### Réaliser la requête :

```
SELECT idJoueur, t_commande.idCommande, t_commande.comDate, t_commande.comNumeroCommande FROM t_joueur LEFT JOIN t_commande ON t_joueur.idJoueur = t_commande.fkJoueur;
```



24 - Requête 7

### Explication de la commande :

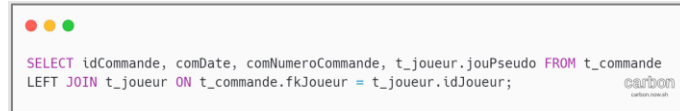
Utilisation de `LEFT JOIN` pour faire une jointure au lieu d'un `INNER JOIN` afin de récupérer tous les joueurs même s'ils n'ont aucun lien avec la table commande.

### Requête n°8 :

*Récupérer toutes les commandes et afficher le pseudo du joueur si elle existe, sinon montrer 'NULL' pour le pseudo.*

### Réaliser la requête :

```
SELECT idCommande, comDate, comNumeroCommande, t_joueur.jouPseudo FROM t_co  
mmande LEFT JOIN t_joueur ON t_commande.fkJoueur = t_joueur.idJoueur;
```



25 - Requête 8

### Explication de la commande :

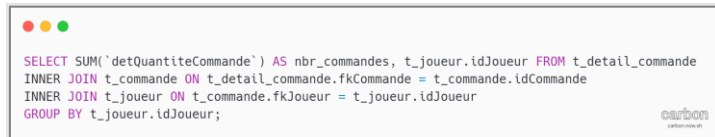
Utilisation d'un `LEFT JOIN` afin de récupérer toutes les valeurs de la table commande même si elles ne sont pas liées à la table joueur. De cette façon si la commande n'est reliée à aucun joueur elle sera quand même affichée et le nom du joueur restera 'NULL'.

### Requête n°9 :

*Trouver le nombre total d'armes achetées par chaque joueur (même si ce joueur n'a acheté aucune Arme).*

### Réaliser la requête :

```
SELECT SUM(`detQuantiteCommande`) AS nbr_commandes, t_joueur.idJoueur FROM  
t_detail_commande INNER JOIN t_commande ON t_detail_commande.fkCommande = t  
_commande.idCommande INNER JOIN t_joueur ON t_commande.fkJoueur = t_joueur.  
idJoueur GROUP BY t_joueur.idJoueur;
```



26 - Requête 9

### Explication de la commande :

Afin de sélectionner le nombre total d'arme acheté je fais un `SUM` de la quantité commandé car une commande peut comporter plusieurs armes. Par la suite je rejoin les tables nécessaires pour pouvoir accéder de la table `t_detail_commande` dans laquelle il y a le nombre de commande et la table `t_joueur` ou je peux obtenir l'identifiant et le nom du joueur.

### Requête n°10 :

*Trouver les joueurs qui ont achetés plus de 3 types d'armes différentes.*

#### Réaliser la requête :

```
SELECT t_joueur.jouPseudo, COUNT(DISTINCT t_arme.idArme) AS nb_arme_differe  
nte FROM t_joueur LEFT JOIN t_commande ON t_joueur.idJoueur = t_commande.fk  
Joueur LEFT JOIN t_detail_commande ON t_commande.idCommande = t_detail_comm  
ande.fkCommande LEFT JOIN t_arme ON t_detail_commande.fkArme = t_arme.idArm  
e GROUP BY t_joueur.idJoueur HAVING COUNT(DISTINCT t_arme.idArme) > 3;
```



```
SELECT t_joueur.jouPseudo, COUNT(DISTINCT t_arme.idArme) AS nb_arme_differe  
nte  
FROM t_joueur  
LEFT JOIN t_commande ON t_joueur.idJoueur = t_commande.fkJoueur  
LEFT JOIN t_detail_commande ON t_commande.idCommande = t_detail_commande.fkCommande  
LEFT JOIN t_arme ON t_detail_commande.fkArme = t_arme.idArme  
GROUP BY t_joueur.idJoueur  
HAVING COUNT(DISTINCT t_arme.idArme) > 3;
```

27 - Requête 10

#### Explication de la commande :

Dans cette commande j'utilise trois LEFT JOIN afin de pouvoir créer une union avec toutes les tables, ainsi je peux récupérer toutes les informations puis n'en prendre que ceux qui ont plus de 3 armes différentes.

Afin de ne pas sélectionner deux fois la même arme si le joueur l'a acheté plusieurs fois j'utilise un **DISTINCT** dans le **COUNT** tel que **COUNT(DISTINCT t\_arme.idArme)**.

## D. Création des index

### 1. Pourtant certains index existent déjà. Pourquoi ?

MySQL crée automatiquement des indexes sur les clefs primaires et sur les clefs étrangères.

Ces indexes sont créés automatiquement car les clefs primaires et les clefs étrangères sont fréquemment utilisés lors de requêtes, spécifiquement lorsque l'on souhaite créer des jointures

### 2. Quels sont les avantages et les inconvénients des index ?

#### **Avantages :**

Les indexes permettent de structurer les données afin d'effectuer des recherches dans les données beaucoup plus rapidement.

#### **Inconvénients :**

Les indexes prennent de la place en mémoire

Ils ralentissent les requêtes car l'index doit se remettre à jour à chaque changement :

- D'insertion
- De modification
- De suppression

### 3. Sur quel champ (de quelle table), cela pourrait être pertinent d'ajouter un index ?

Sur les clefs primaires et les clefs étrangères (dans le cas où elle n'ont pas été automatiquement créées par le SGBDR utilisé). Cela permettrait d'effectuer des requêtes plus rapidement lorsque l'on souhaite effectuer des jointures. De plus des indexes peuvent être rajoutés.

## E. Backup / Restore

### Backup de la base de données :

```
Mysqldump -uroot -proot nom_db > nom.sql
```

```
Mysqldump -u root -p nom_db > nom.sql
```

### Explication de la Backup

`Mysqldump` permet de spécifier que c'est l'utilitaire « Mysqldump » qui est utilisé.

`-uroot` spécifie que l'utilisateur est `ROOT`

`-p` spécifie que l'on va rentrer le « *password* » (mot de passe). Rien n'est mis à la suite afin de ne pas le divulguer.

### Restore la base de données :

```
Mysql -uroot -p < nom.sql
```

### Explication du Restore :

`Mysql` spécifie que la commande est une commande Mysql.

`-uroot -p` permet de se connecter à Mysql.

`<` signifie qu'une importation va se faire du fichier situé à droite de la flèche.

`nom.sql` est le nom du fichier qui est restauré.

## F. Connexion DB à C#

Afin d'effectuer une connexion de la base de données au programme C# j'ai effectué un clic droit sur la solution « Spice Invaders » puis j'ai choisi « *Gérer les packages NuGet pour la solution...* ». J'ai ensuite recherché « *MySQL.Data* » et je l'ai installé. C'est ensuite dans la classe « *Storage* » que j'ai effectué la connexion. Pour cela j'ai premièrement écrit :

```
using MySql;  
using MySql.Data;  
using System.Data.SqlClient;
```

Ces lignes de code me permettent d'utiliser le package NuGet « *MySql.Data* ». Ensuite afin de se connecter à la base de données j'utilise ceci :

### Explication du connection String :

Afin d'effectuer la connexion à la base de données j'ai utilisé un connection string, c'est un string qui va contenir les informations nécessaires à la connexion de la base de données tel que :

**Server=localhost;database=SpicyInvaders;UID=root;password=root;port=6033;**

Ainsi lorsque je souhaite me connecter à la base de données je crée une nouvelle connexion avec comme paramètre ce string.

```
Connection = new MySqlConnection(str);
```

À la place du « *str* » j'aurai aussi pu écrire le string en lui-même, j'ai décidé de ne pas le faire afin que mon code soit plus propre.

Les paramètres spécifiés pour garantir la connexion à une base de données sont :

Le serveur, dans ce cas localhost.

Le nom de la base de données

L'UID, Unique Identifier, c'est l'identifiant unique de l'utilisateur

Le mot de passe de l'utilisateur

Le port pour se connecter, le port est 6033 car c'est le port par défaut de docker SQL.

## Conclusion Technique

En débutant la partie DB du projet SpicyInvaders j'ai d'abord dû appliquer des connaissances que j'avais déjà, requêtes SQL. J'ai par la suite dû acquérir des compétences sur la création des utilisateurs et des rôles, la création et compréhension des indexes, l'utilisation de backup et de restore grâce à mysqldump et apprendre à faire la connexion entre un programme C# et une base de données, le cahier des charges ayant été légèrement modifié il a fallu effectuer cette connexion.

## Conclusion Personnelle

Dans ce projet je n'ai pas eu de grande difficulté mais je pense avoir passé trop de temps à retravailler mon rapport et à l'arrivée du délai final j'ai dû me dépêcher de faire la connexion entre la base de données et le programme C#.

Ce projet m'a appris des bases pour la gestion de base de données avec MySQL et m'a rappelé comment effectuer des requêtes dans une base de données.



## 8.2 Utilisation d'IA dans le projet

### Partie POO

Dans le cadre programmation orienté objet de ce projet je n'ai pas utilisé d'IA. En cas de difficulté j'ai préféré m'orienter sur des forums ou demander de l'aide à des connaissances.

J'ai néanmoins essayé d'utiliser ChatGPT en lui demandant de me créer des ASCII ART de vaisseau. J'ai rapidement vu que cela ne fonctionnerait pas et j'ai vite abandonner cette idée.

### Partie DB

Je n'ai pas utilisé d'IA pour la partie DB de mon projet SpicyInvaders.

## Partie UX

Dans le cadre UX de ce projet j'ai utilisé l'IA ChatGPT (<https://chat.openai.com/>) afin de :

Créer une biographie de mes 2 personas. ChatGPT m'a donné des biographies trop longues que j'ai dû reformuler.

Afin d'affiner mes recherches j'ai donné les informations nécessaires à la création du persona tel que ceci :

J'aimerais que tu crées la biographie de ce Persona qui est un humain jouant à Space Invaders, voici quelques informations complémentaires :

Prénom : Martine

Âge : 35 ans

Mère de famille

Passe beaucoup de temps dans les transports dû à son travail

Joue pour passer le temps Jouait anciennement à Candy Crush mais a fini tous les niveaux Joue à Space Invader depuis son téléphone

Je n'ai pas utilisé ChatGPT pour autre chose.

J'ai néanmoins utilisé l'IA Craiyon (<https://www.craiyon.com/>) qui est une IA génératrice d'image afin d'obtenir des vaisseaux spatiaux dans le but d'avoir plusieurs Sprite.



28 – Résultat final

## Autre

J'ai utilisé l'IA « ChatGPT » (<https://chat.openai.com/>) afin de savoir comment faire tenir mon sommaire sur une seule page, cela ne m'a pas aidé.