Présenté par Kévin Lionnet

SOMMAIRE

01

GENESE

HTML

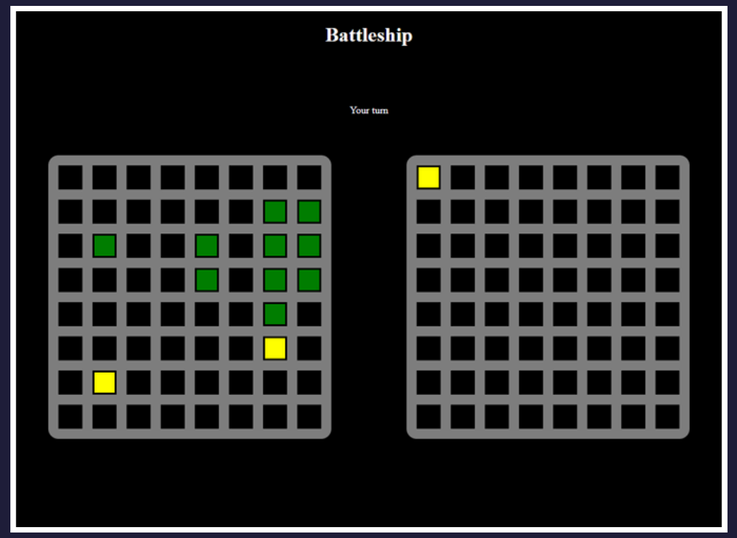
CSS

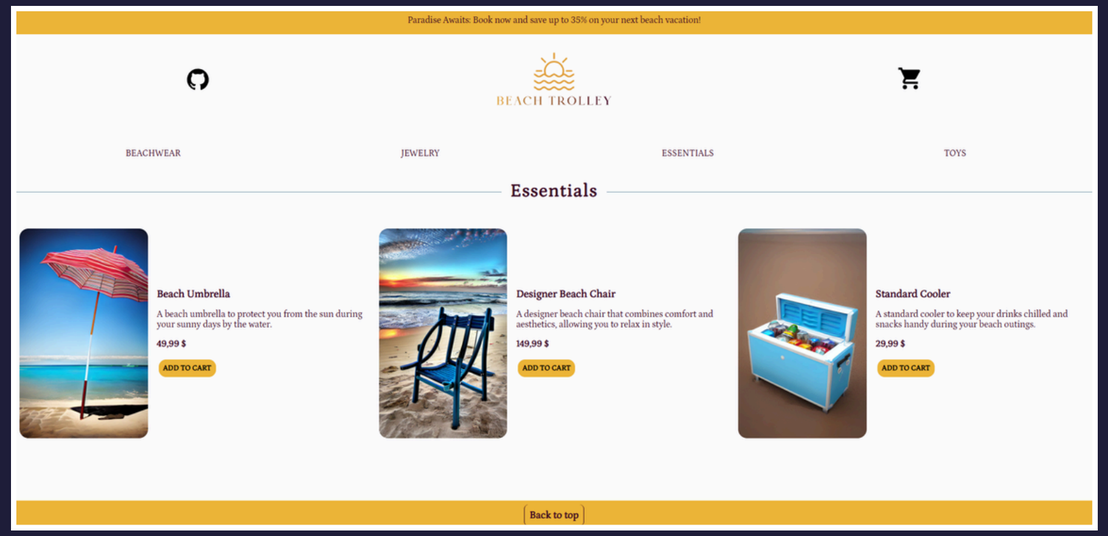
JS

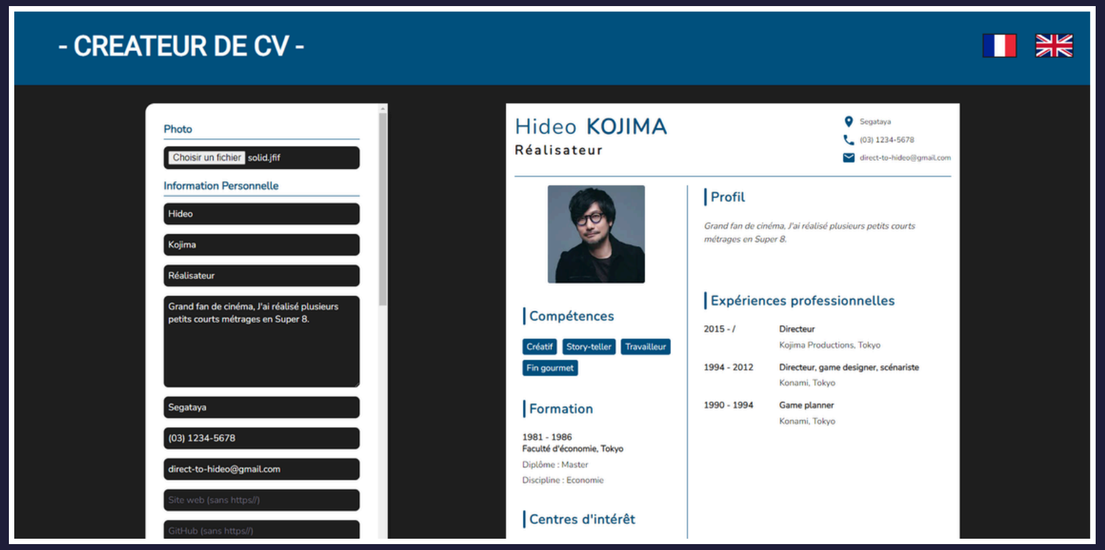
React

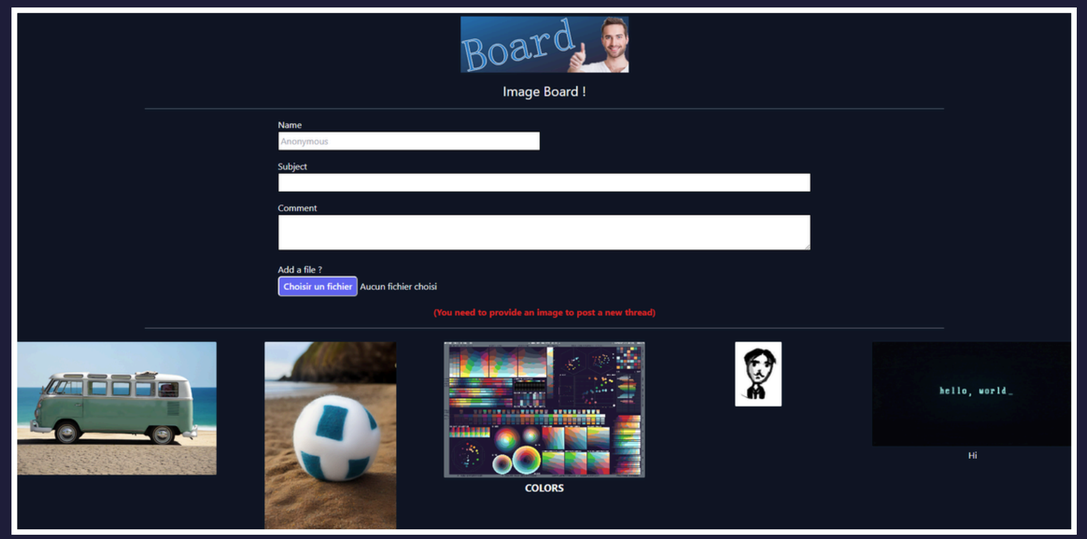
Node.js

MongoDB

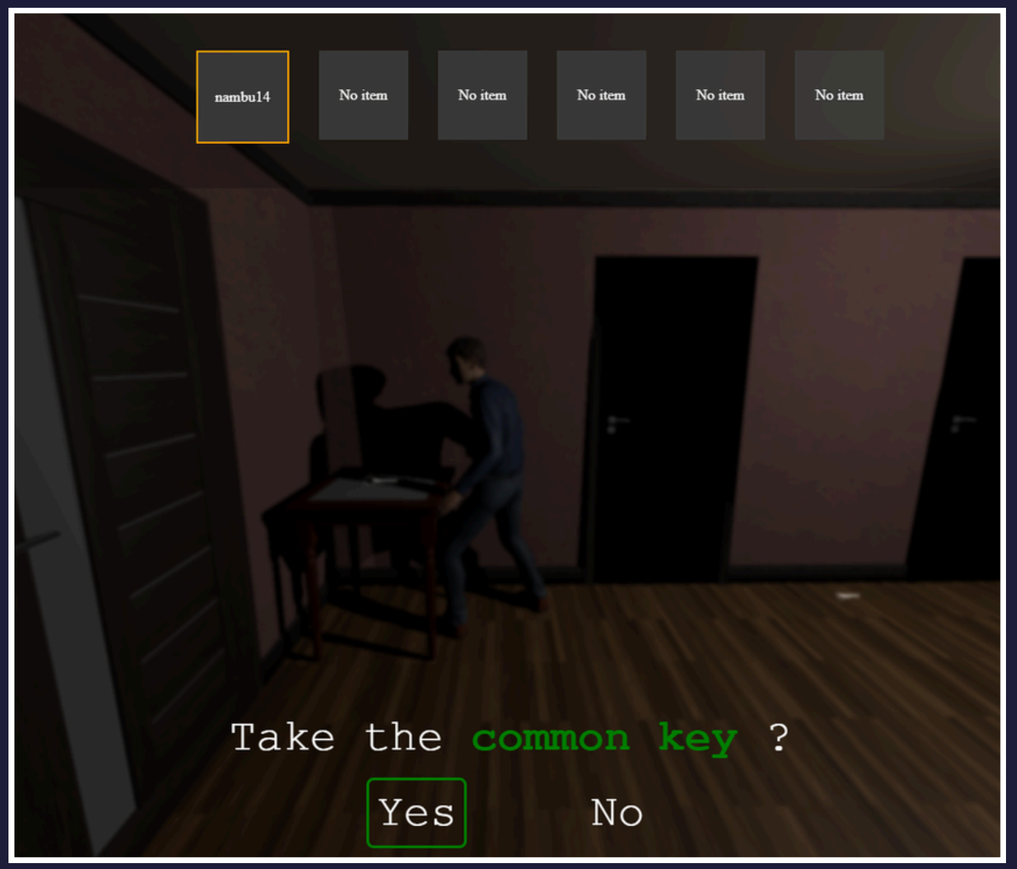

Battleship
Your turn


Paradise Awaits: Book now and save up to 35% on your next beach vacation!
BEACH TROLLEY
BEACHWEAR
JEWELRY
ESSENTIALS
TOYS
Essentials
Beach Umbrella
A beach umbrella to protect you from the sun during your sunny days by the water.
49,99 $
ADD TO CART
Designer Beach Chair
A designer beach chair that combines comfort and aesthetics, allowing you to relax in style.
149,99 $
ADD TO CART
Standard Cooler
A standard cooler to keep your drinks chilled and snacks handy during your beach outings.
29,99 $
ADD TO CART
Back to top


- CREATEUR DE CV -
Photo
Choisir un fichier
solid.jfif
Information Personnelle
Hideo
Kojima
Réalisateur
Grand fan de cinéma, j'ai réalisé plusieurs petits courts métrages en Super 8.
Segataya
(03) 1234-5678
direct-to-hideo@gmail.com
Site web (sans https://)
GitHub (sans https://)
Hideo KOJIMA
Réalisateur
Segataya
(03) 1234-5678
direct-to-hideo@gmail.com
Profil
Grand fan de cinéma, j'ai réalisé plusieurs petits courts métrages en Super 8.
Compétences
Créatif
Story-teller
Travailleur
Fin gourmet
Formation
1981 - 1986
Faculté d'économie, Tokyo
Diplôme : Master
Discipline : Economie
Centres d'intérêt
Expériences professionnelles
2015 - /
Directeur
Kojima Productions, Tokyo
1994 - 2012
Directeur, game designer, scénariste
Konami, Tokyo
1990 - 1994
Game planner
Konami, Tokyo


Board
Image Board !
Name
Anonymous
Subject
Comment
Add a file ?
Choisir un fichier
Aucun fichier choisi
(You need to provide an image to post a new thread)
COLORS
hello, world_
Hi

three.js


nambu14
No item
No item
No item
No item
No item
Take the common key ?
Yes
No

3D models

Fullstack game

Deck based asynchronous combat

02

SYSTÈME DE JEU

Cartes à ordonner avant le combat

01

Conditions
Effets

02

Conditions
Effets

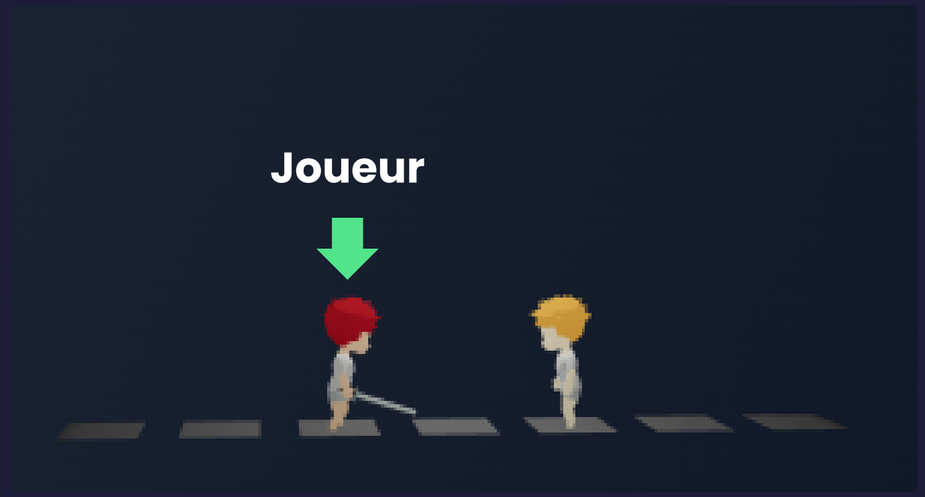
03

Conditions
Effets

04

Conditions
Effets

Test de chaque carte dans l'ordre



01 ✗

Conditions :
Est à une case adjacente de l'adversaire

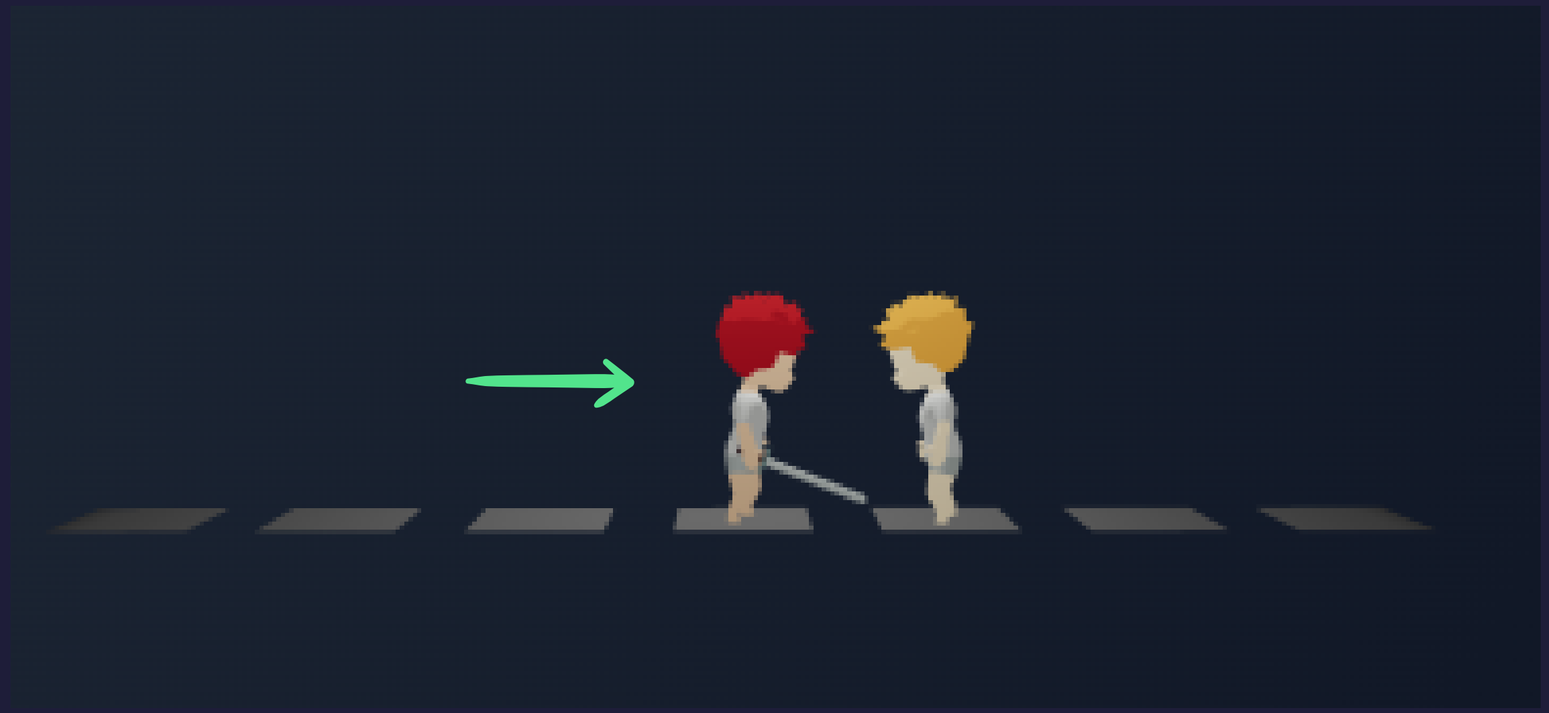
Effets :
Attaque

02 ✓

Conditions :
Ne pas être à moins d'une case de l'adversaire

Effets :
Se déplacer d'une case vers l'adversaire

Réalisation de l’effet, au tour du combattant suivant



Résolution des combats à l’aide des cartes et des statistiques

Système d’équipement

03 OBJECTIFS

OBJECTIF SMART

S	M	A	R	T
Spécifique	Mesurable	Atteignable	Réaliste	Temporel
Créer un MVP autobattler basé sur des personnages en 3D, où les joueurs construisent des decks d'actions conditionnées.	Prototype 100% fonctionnel avec les fonctionnalités principales Garantir un temps de réponse serveur de 300ms	Miser sur les compétences acquises lors de la formation Hébergement sur des plateformes économiques	En fonction de l'avancement par rapport au planning prévisionnel, recadrer le projet pour ne garder que l'essentiel sans omettre de laisser place à de fonctionnalités futures	Lancement du MVP prévu en août 2025 Mises à jour et maintenance continues après le lancement selon le retour des utilisateurs

04 CONCEPTION VISUELLE

POLICES

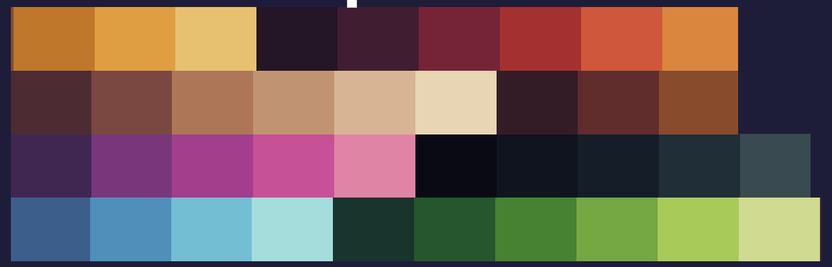
LOGO/TITRES
BANGERS
WHEREAS DISREGARD AND CONTEMPT FOR HUMAN RIGHTS HAVE RESULTED

Alternatif
Alfa Slab One
Whereas disregard and contempt for human rights have resulted

Texte courant
Poppins
Whereas disregard and contempt for human rights have resulted

PALETTE

“Apollo”



Lospec.com
par ***AdamCYounis***

FDFDFD

1E1D39

3c5e8b

e8c170

468232 Valide

A53030 Erreur

MODÈLES 3D


TIZ
BRAVELY DEFAULT


Über falcon blade
Erdrick's shield
Erdrick's helmet
Pallium Regale
Ethereal gloves
Sturdy slacks
Tricksie boots
Meteorite bracer
L
R
START
Elyon
Lv. 99
B Back

MODÈLES 3D

Blender

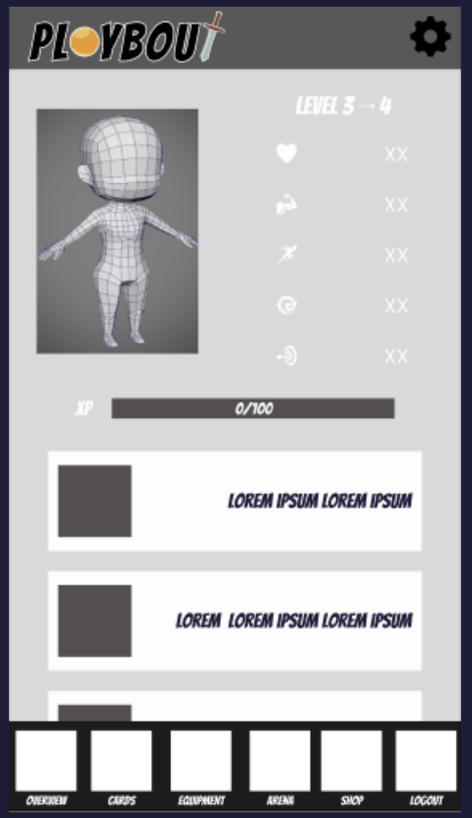
Moins de 1000 polygones

MAQUETTES

Figma

Zoning

Wireframe

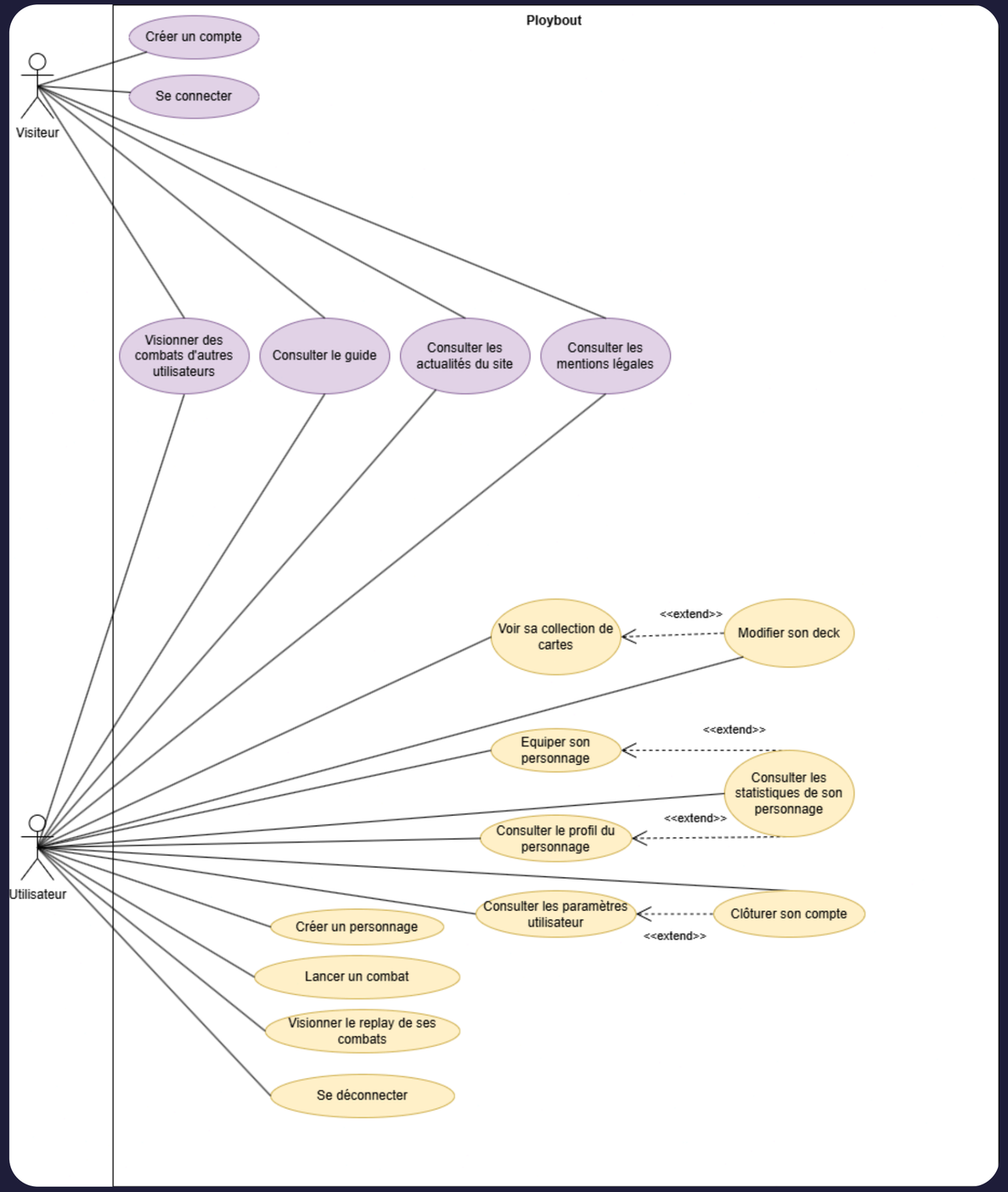


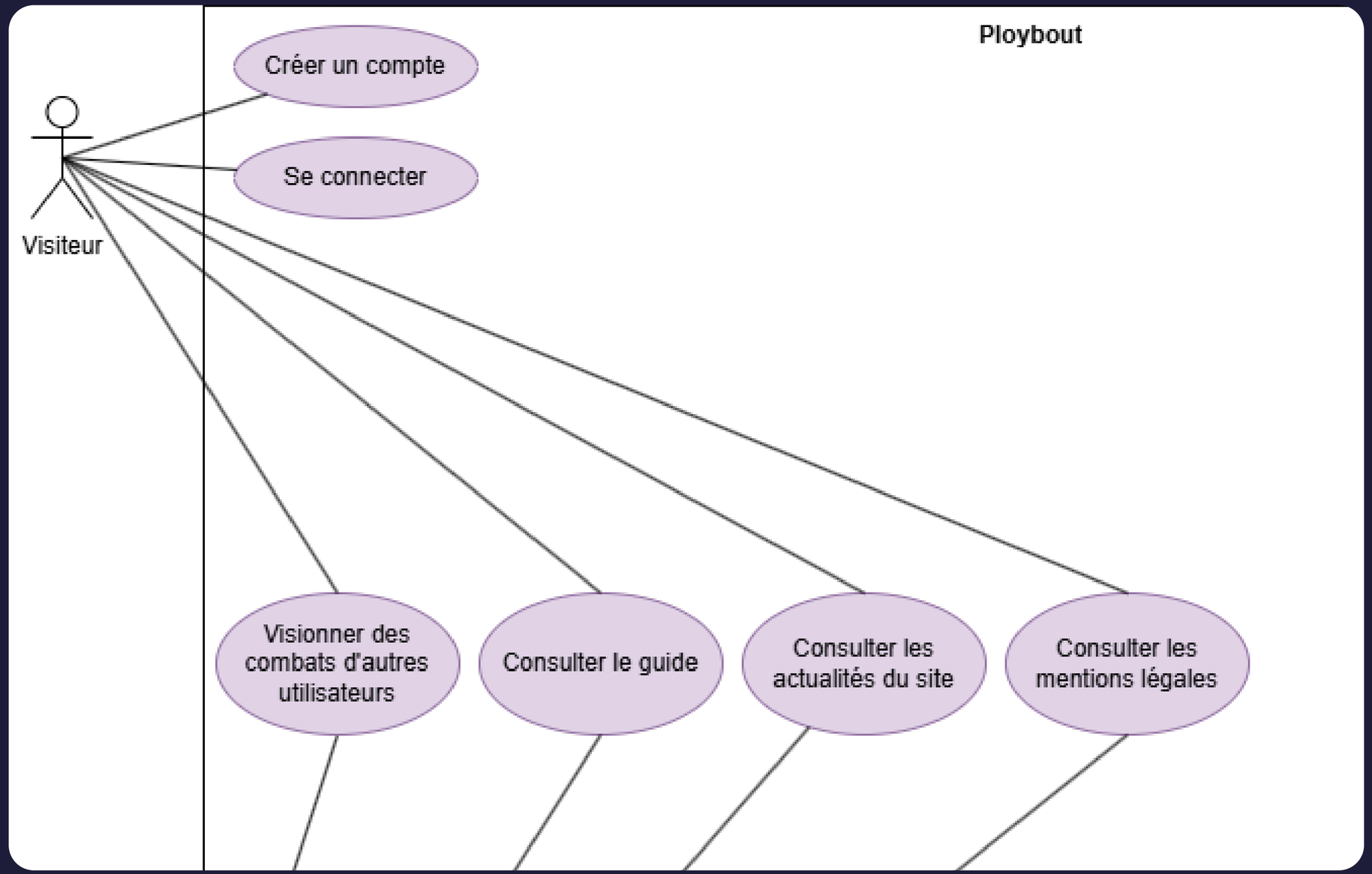
Mockup

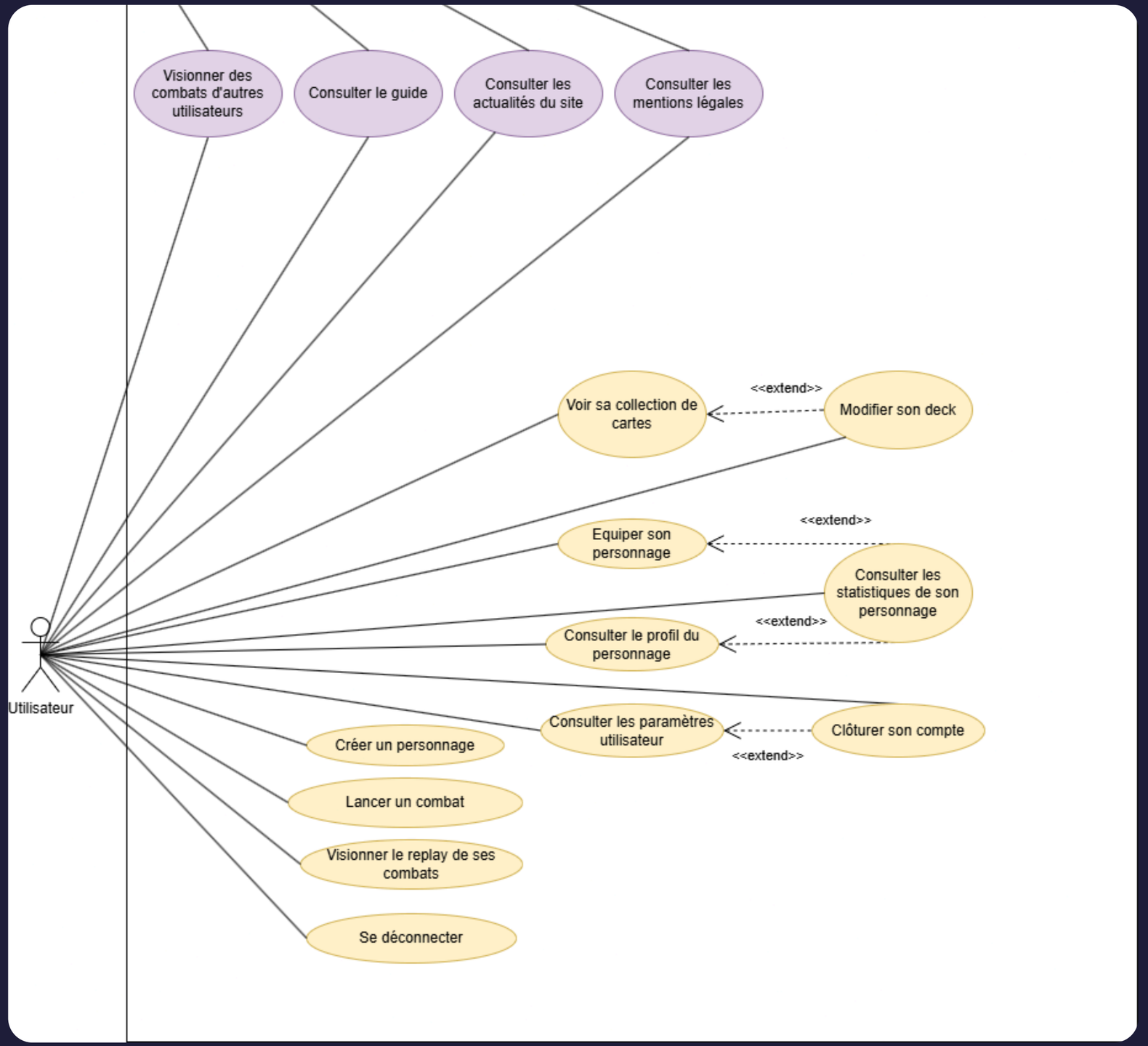
05

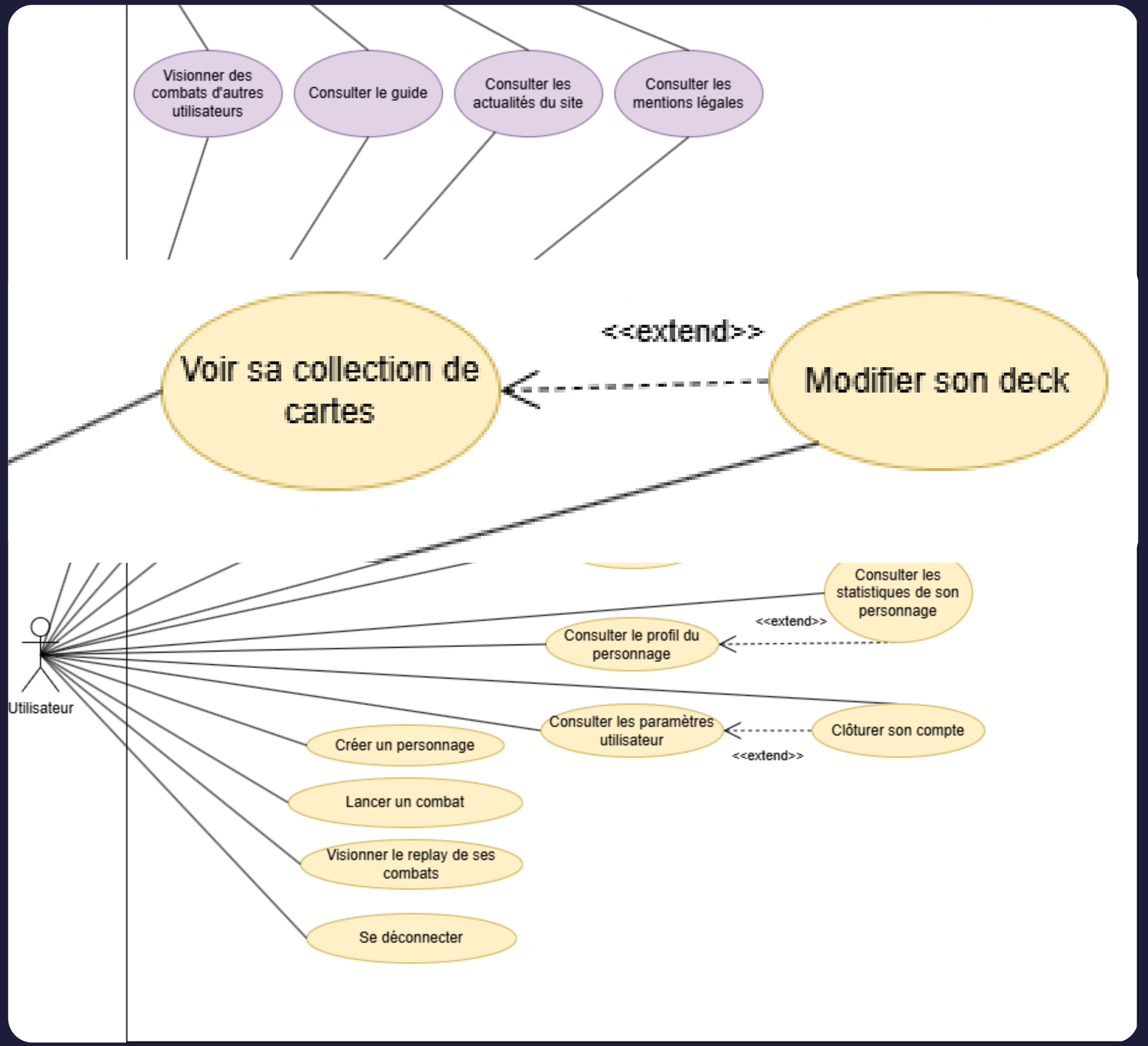
CONCEPTION TECHNIQUE (UML)

CAS D’UTILISATION


Ploybout
Visiteur
Créer un compte
Se connecter
Visionner des combats d'autres utilisateurs
Consulter le guide
Consulter les actualités du site
Consulter les mentions légales
Utilisateur
Voir sa collection de cartes
<<extend>>
Modifier son deck
Equiper son personnage
<<extend>>
Consulter les statistiques de son personnage
Consulter le profil du personnage
<<extend>>
Consulter les paramètres utilisateur
Clôturer son compte
<<extend>>
Créer un personnage
Lancer un combat
Visionner le replay de ses combats
Se déconnecter

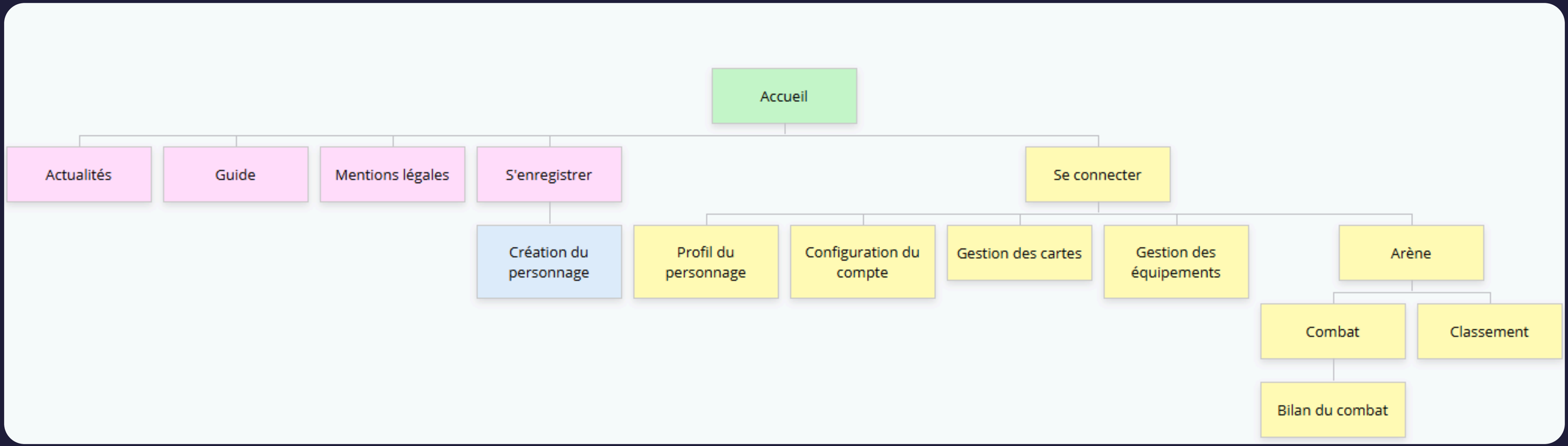

Ployboult
Créer un compte
Se connecter
Visiteur
Visionner des combats d'autres utilisateurs
Consulter le guide
Consulter les actualités du site
Consulter les mentions légales


Visionner des combats d'autres utilisateurs
Consulter le guide
Consulter les actualités du site
Consulter les mentions légales
Voir sa collection de cartes
<<extend>>
Modifier son deck
Equiper son personnage
<<extend>>
Consulter les statistiques de son personnage
Consulter le profil du personnage
<<extend>>
Utilisateur
Consulter les paramètres utilisateur
Clôturer son compte
<<extend>>
Créer un personnage
Lancer un combat
Visionner le replay de ses combats
Se déconnecter

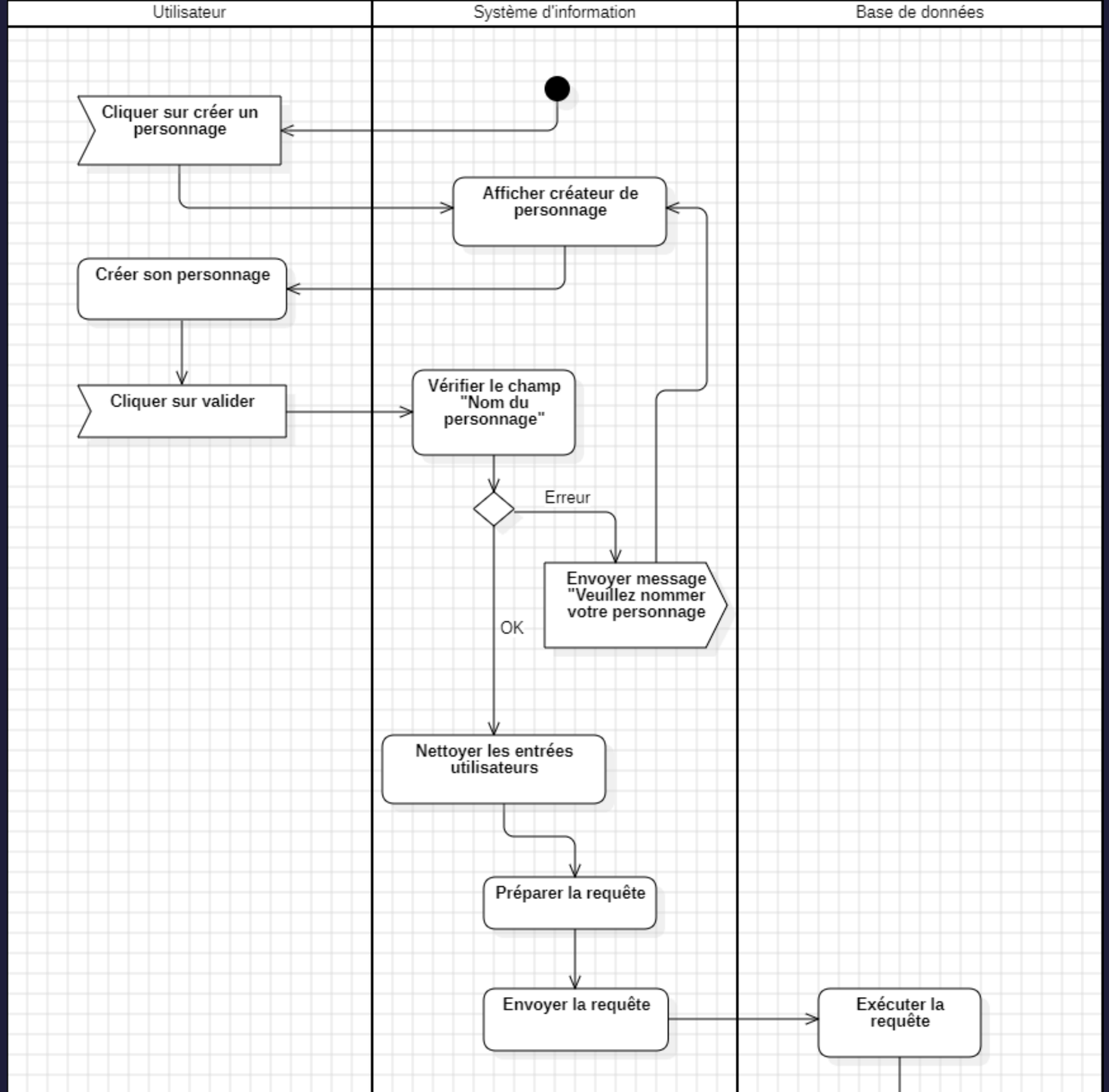

Visionner des combats d'autres utilisateurs
Consulter le guide
Consulter les actualités du site
Consulter les mentions légales
Voir sa collection de cartes
<<extend>>
Modifier son deck
Consulter les statistiques de son personnage
<<extend>>
Consulter le profil du personnage
Utilisateur
Consulter les paramètres utilisateur
Clôturer son compte
<<extend>>
Créer un personnage
Lancer un combat
Visionner le replay de ses combats
Se déconnecter

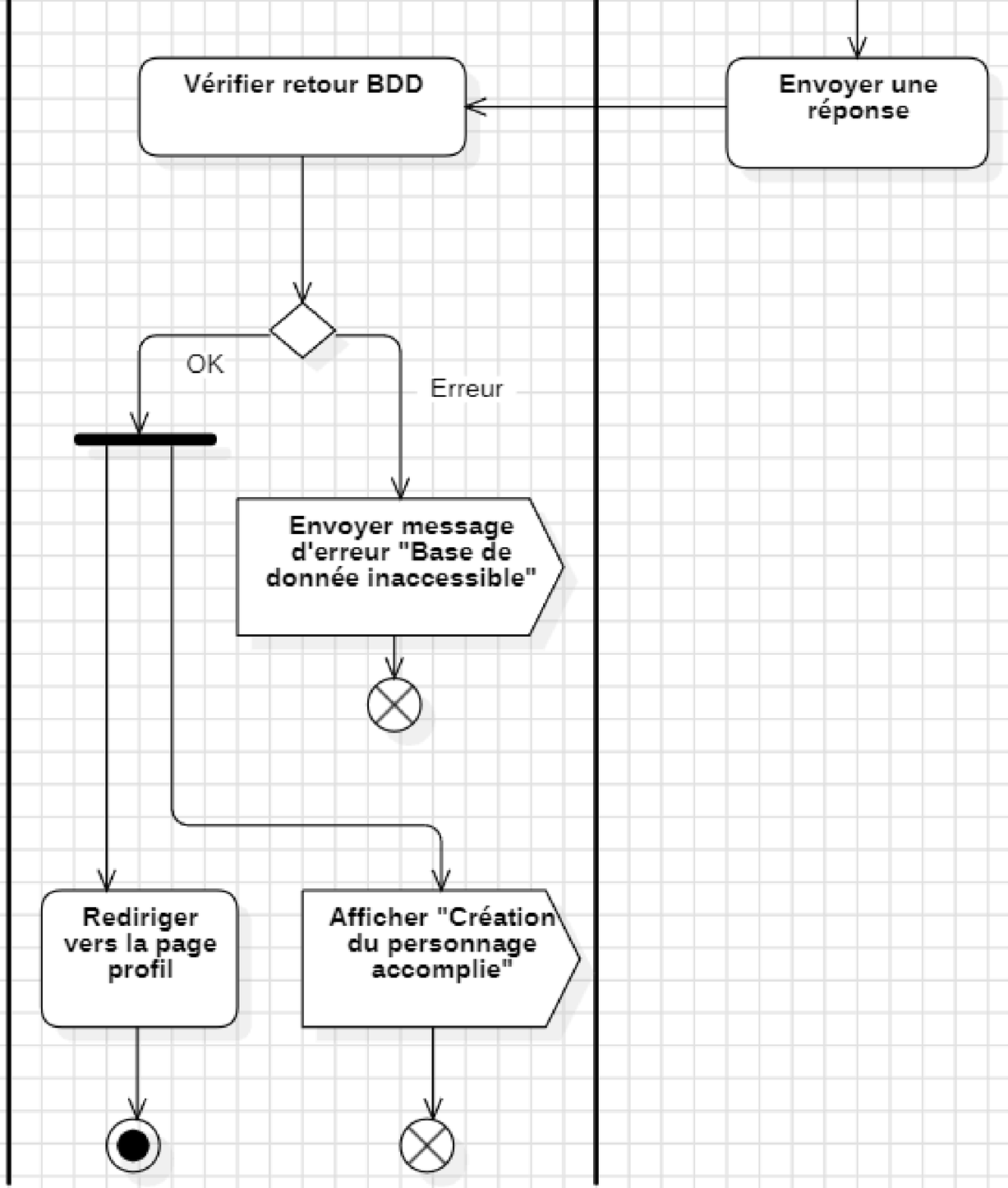
ARBORESCENCE

GlooMaps



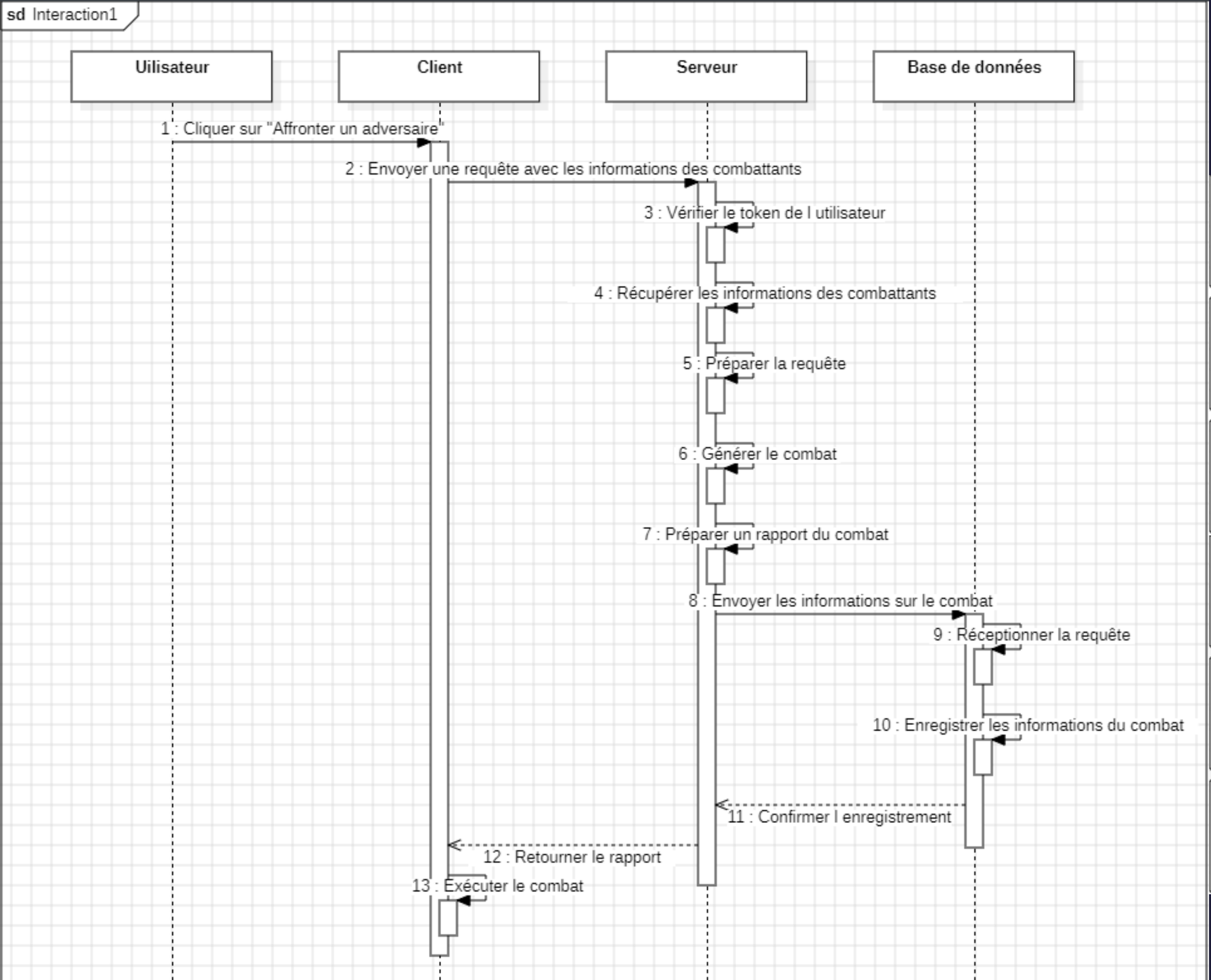
CRÉATION DE PERSONNAGE


Utilisateur
Système d'information
Base de données
Cliquer sur créer un personnage
Afficher créateur de personnage
Créer son personnage
Cliquer sur valider
Vérifier le champ "Nom du personnage"
Erreur
Envoyer message "Veuillez nommer votre personnage
OK
Nettoyer les entrées utilisateurs
Préparer la requête
Envoyer la requête
Exécuter la requête


Vérifier retour BDD
Envoyer une réponse
OK
Erreur
Envoyer message d'erreur "Base de donnée inaccessible"
Rediriger vers la page profil
Afficher "Création du personnage accomplie"

StarUML

DUEL



StarUML

(2) Scénario erreur : Le serveur ne répond pas.
2.1 Le SI envoie un message d'erreur au Client.
2.2 Le Client affiche un message d'erreur à l'Utilisateur "Le serveur ne répond pas".
Le scénario s'arrête ici.

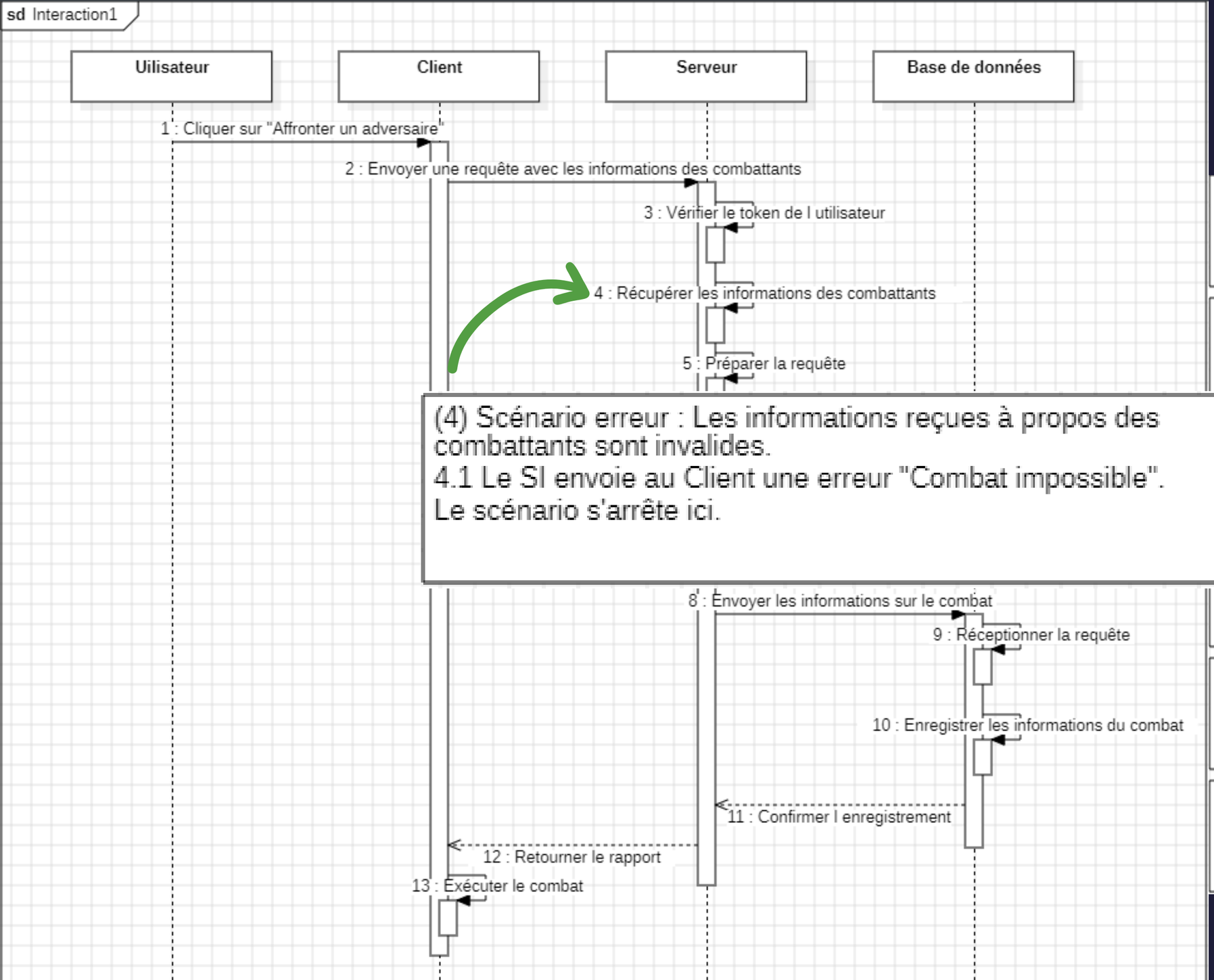
(3) Scénario erreur : Le token envoyé par le Client n'est pas valide.
3.1 Le SI envoie au Client une erreur d'authentification.
3.2 Le Client déconnecte l'utilisateur à la suite d'un message d'erreur l'invitant à se reconnecter.
Le scénario s'arrête ici.

(4) Scénario erreur : Les informations reçues à propos des combattants sont invalides.
4.1 Le SI envoie au Client une erreur "Combat impossible".
Le scénario s'arrête ici.

(6) Scénario erreur : Erreur lors de la génération du combat.
6.1 Le SI envoie au Client une erreur "Combat impossible".
Le scénario s'arrête ici.

(8) Scénario erreur : La base de données est hors service.
8.1 Le SI envoie un message d'erreur au Client.
8.2 Le Client affiche un message d'erreur à l'Utilisateur "Le serveur ne répond pas".
Le scénario s'arrête ici.

(11) Scénario erreur : Erreur dans la base de données lors de l'enregistrement des informations du combat.
11.1 Le SI envoie au Client une erreur de l'ajout du combat dans la base de donnée.
Le scénario s'arrête ici.



StarUML

(2) Scénario erreur : Le serveur ne répond pas.
2.1 Le SI envoie un message d'erreur au Client.
2.2 Le Client affiche un message d'erreur à l'Utilisateur "Le serveur ne répond pas".
Le scénario s'arrête ici.

(3) Scénario erreur : Le token envoyé par le Client n'est pas valide.
3.1 Le SI envoie au Client une erreur d'authentification.
3.2 Le Client déconnecte l'utilisateur à la suite d'un message d'erreur l'invitant à se reconnecter.
Le scénario s'arrête ici.

rreur : Les informations reçues à propos des
ont invalides.
ie au Client une erreur "Combat impossible".
arrête ici.

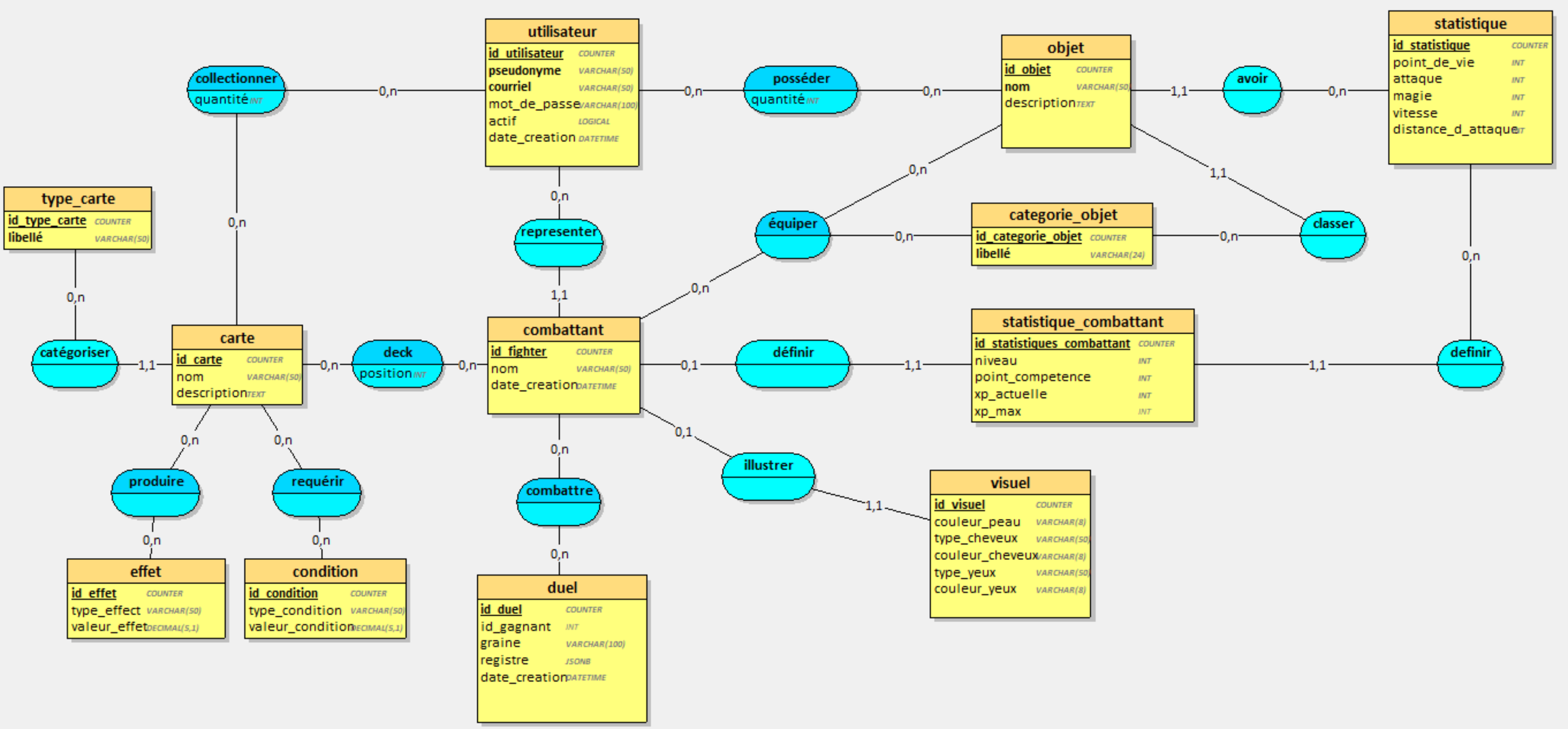
rreur : Erreur lors de la génération du combat.
ie au Client une erreur "Combat impossible".
arrête ici.

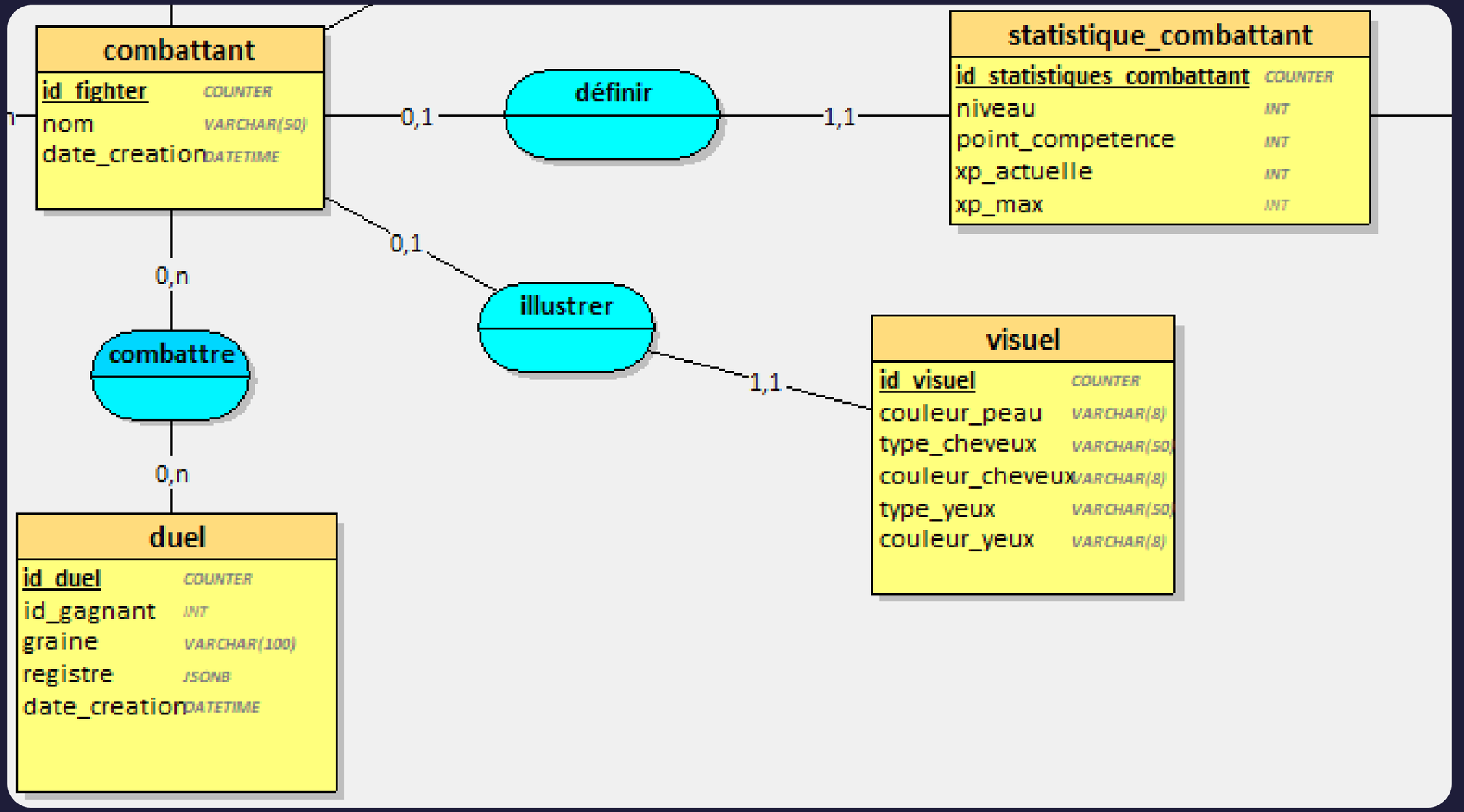
(8) Scénario erreur : La base de données est hors service.
8.1 Le SI envoie un message d'erreur au Client.
8.2 Le Client affiche un message d'erreur à l'Utilisateur "Le serveur ne répond pas".
Le scénario s'arrête ici.

(11) Scénario erreur : Erreur dans la base de données lors de l'enregistrement des informations du combat.
11.1 Le SI envoie au Client une erreur de l'ajout du combat dans la base de donnée.
Le scénario s'arrête ici.

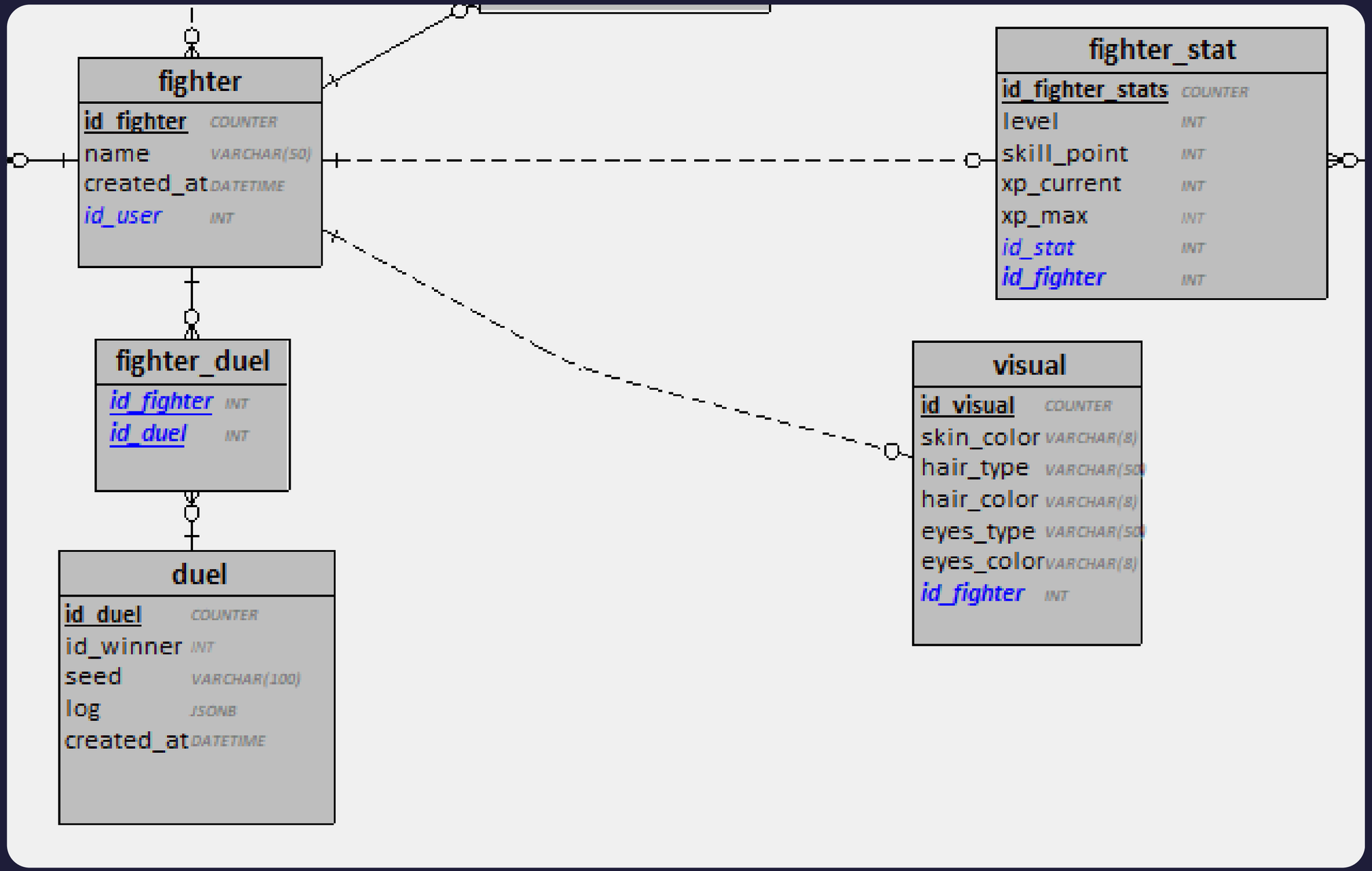
Looping

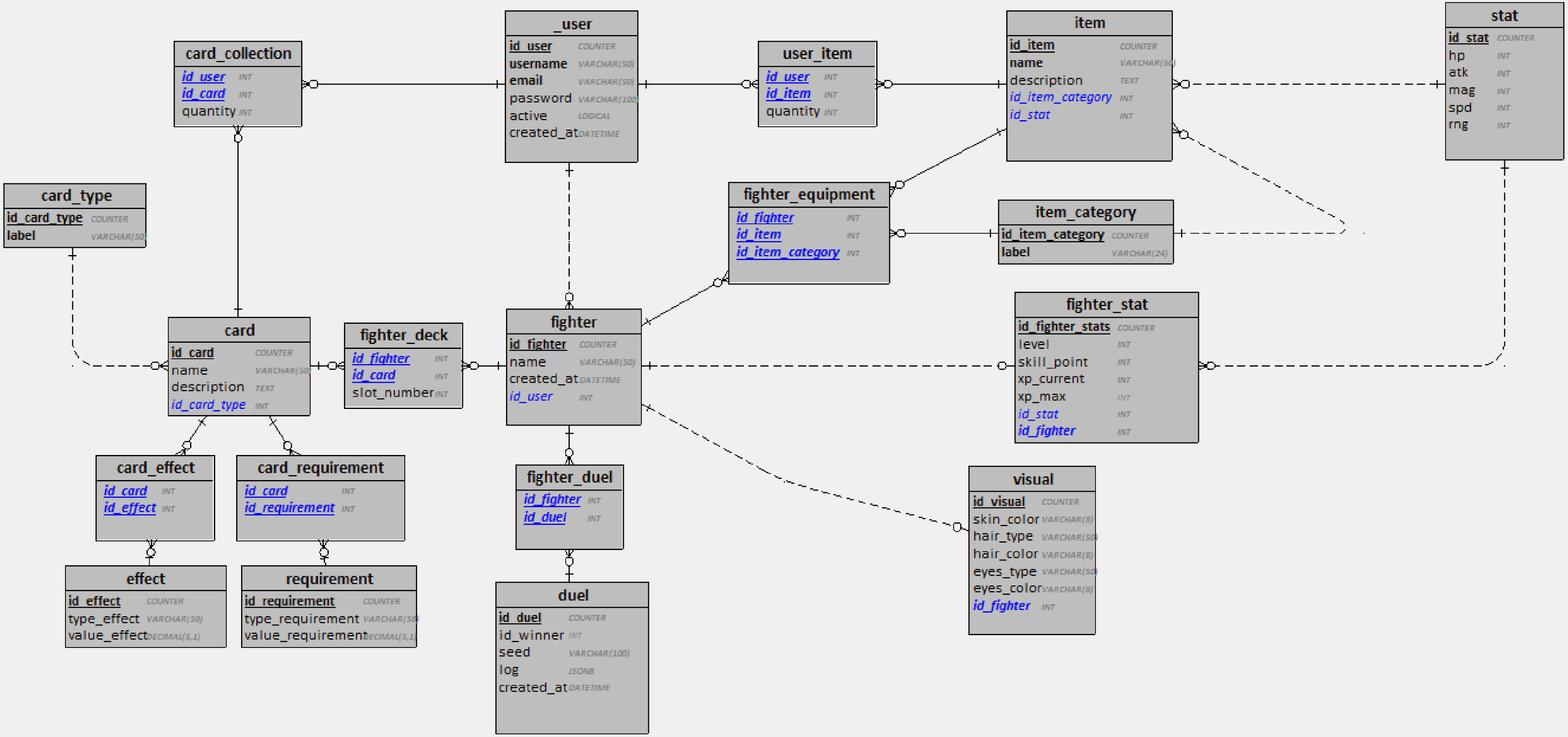
MODELE CONCEPTUEL DE DONNÉES (MCD)


utilisateur
id_utilisateur COUNTER
pseudonyme VARCHAR(50)
courriel VARCHAR(50)
mot_de_passe VARCHAR(100)
actif LOGICAL
date_creation DATETIME
collectionner
quantité INT
posséder
quantité INT
objet
id_objet COUNTER
nom VARCHAR(50)
description TEXT
avoir
statistique
id_statistique COUNTER
point_de_vie INT
attaque INT
magie INT
vitesse INT
distance_d_attaque INT
type_carte
id_type_carte COUNTER
libellé VARCHAR(50)
representer
équiper
categorie_objet
id_categorie_objet COUNTER
libellé VARCHAR(24)
classer
catégoriser
carte
id_carte COUNTER
nom VARCHAR(50)
description TEXT
deck
position INT
combattant
id_fighter COUNTER
nom VARCHAR(50)
date_creation DATETIME
définir
statistique_combattant
id_statistiques_combattant COUNTER
niveau INT
point_competence INT
xp_actuelle INT
xp_max INT
definir
produire
requérir
combattre
illustrer
visuel
id_visuel COUNTER
couleur_peau VARCHAR(8)
type_cheveux VARCHAR(50)
couleur_cheveux VARCHAR(8)
type_yeux VARCHAR(50)
couleur_yeux VARCHAR(8)
effet
id_effet COUNTER
type_effect VARCHAR(50)
valeur_effet DECIMAL(5,1)
condition
id_condition COUNTER
type_condition VARCHAR(50)
valeur_condition DECIMAL(5,1)
duel
id_duel COUNTER
id_gagnant INT
graine VARCHAR(100)
registre JSONB
date_creation DATETIME


combattant
id_fighter COUNTER
nom VARCHAR(50)
date_creation DATETIME
définir
0,1
1,1
statistique_combattant
id_statistiques_combattant COUNTER
niveau INT
point_competence INT
xp_actuelle INT
xp_max INT
0,n
0,1
illustrer
1,1
visuel
id_visuel COUNTER
couleur_peau VARCHAR(8)
type_cheveux VARCHAR(50)
couleur_cheveux VARCHAR(8)
type_yeux VARCHAR(50)
couleur_yeux VARCHAR(8)
combattre
0,n
duel
id_duel COUNTER
id_gagnant INT
graine VARCHAR(100)
registre JSONB
date_creation DATETIME

MODELE LOGIQUE DE DONNÉES (MLD)


fighter
id_fighter COUNTER
name VARCHAR(50)
created_at DATETIME
id_user INT
fighter_stat
id_fighter_stats COUNTER
level INT
skill_point INT
xp_current INT
xp_max INT
id_stat INT
id_fighter INT
fighter_duel
id_fighter INT
id_duel INT
visual
id_visual COUNTER
skin_color VARCHAR(8)
hair_type VARCHAR(50)
hair_color VARCHAR(8)
eyes_type VARCHAR(50)
eyes_color VARCHAR(8)
id_fighter INT
duel
id_duel COUNTER
id_winner INT
seed VARCHAR(100)
log JSONB
created_at DATETIME

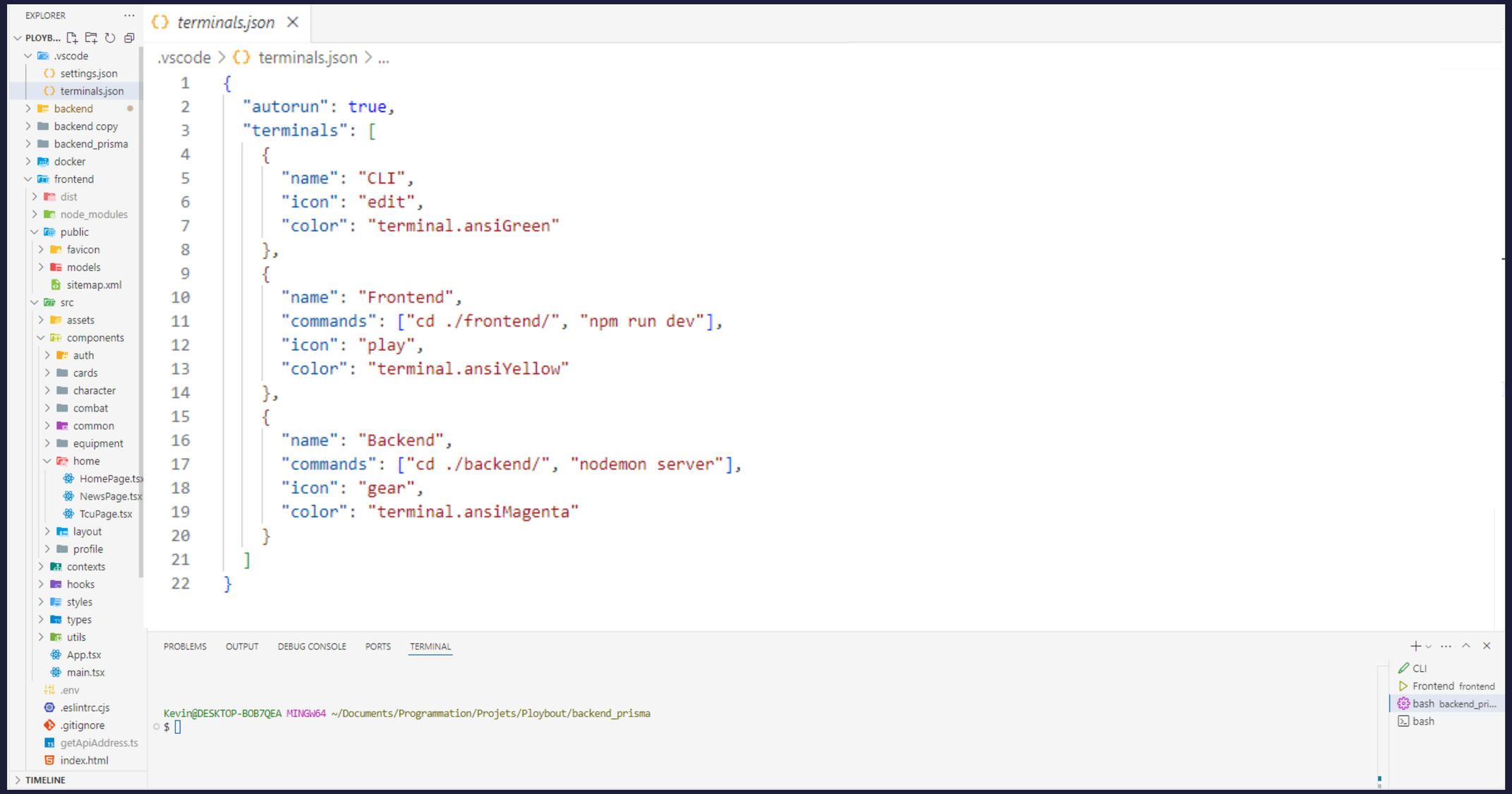

card_collection
id_user INT
id_card INT
quantity INT
_user
id_user COUNTER
username VARCHAR(50)
email VARCHAR(50)
password VARCHAR(100)
active LOGICAL
created_at DATETIME
user_item
id_user INT
id_item INT
quantity INT
item
id_item COUNTER
name VARCHAR(50)
description TEXT
id_item_category INT
id_stat INT
stat
id_stat COUNTER
hp INT
atk INT
mag INT
spd INT
rng INT
card_type
id_card_type COUNTER
label VARCHAR(50)
fighter_equipment
id_fighter INT
id_item INT
id_item_category INT
item_category
id_item_category COUNTER
label VARCHAR(24)
card
id_card COUNTER
name VARCHAR(50)
description TEXT
id_card_type INT
fighter_deck
id_fighter INT
id_card INT
slot_number INT
fighter
id_fighter COUNTER
name VARCHAR(50)
created_at DATETIME
id_user INT
fighter_stat
id_fighter_stats COUNTER
level INT
skill_point INT
xp_current INT
xp_max INT
id_stat INT
id_fighter INT
card_effect
id_card INT
id_effect INT
card_requirement
id_card INT
id_requirement INT
fighter_duel
id_fighter INT
id_duel INT
visual
id_visual COUNTER
skin_color VARCHAR(8)
hair_type VARCHAR(50)
hair_color VARCHAR(8)
eyes_type VARCHAR(50)
eyes_color VARCHAR(8)
id_fighter INT
effect
id_effect COUNTER
type_effect VARCHAR(50)
value_effect DECIMAL(5,1)
requirement
id_requirement COUNTER
type_requirement VARCHAR(50)
value_requirement DECIMAL(5,1)
duel
id_duel COUNTER
id_winner INT
seed VARCHAR(100)
log JSONB
created_at DATETIME

06

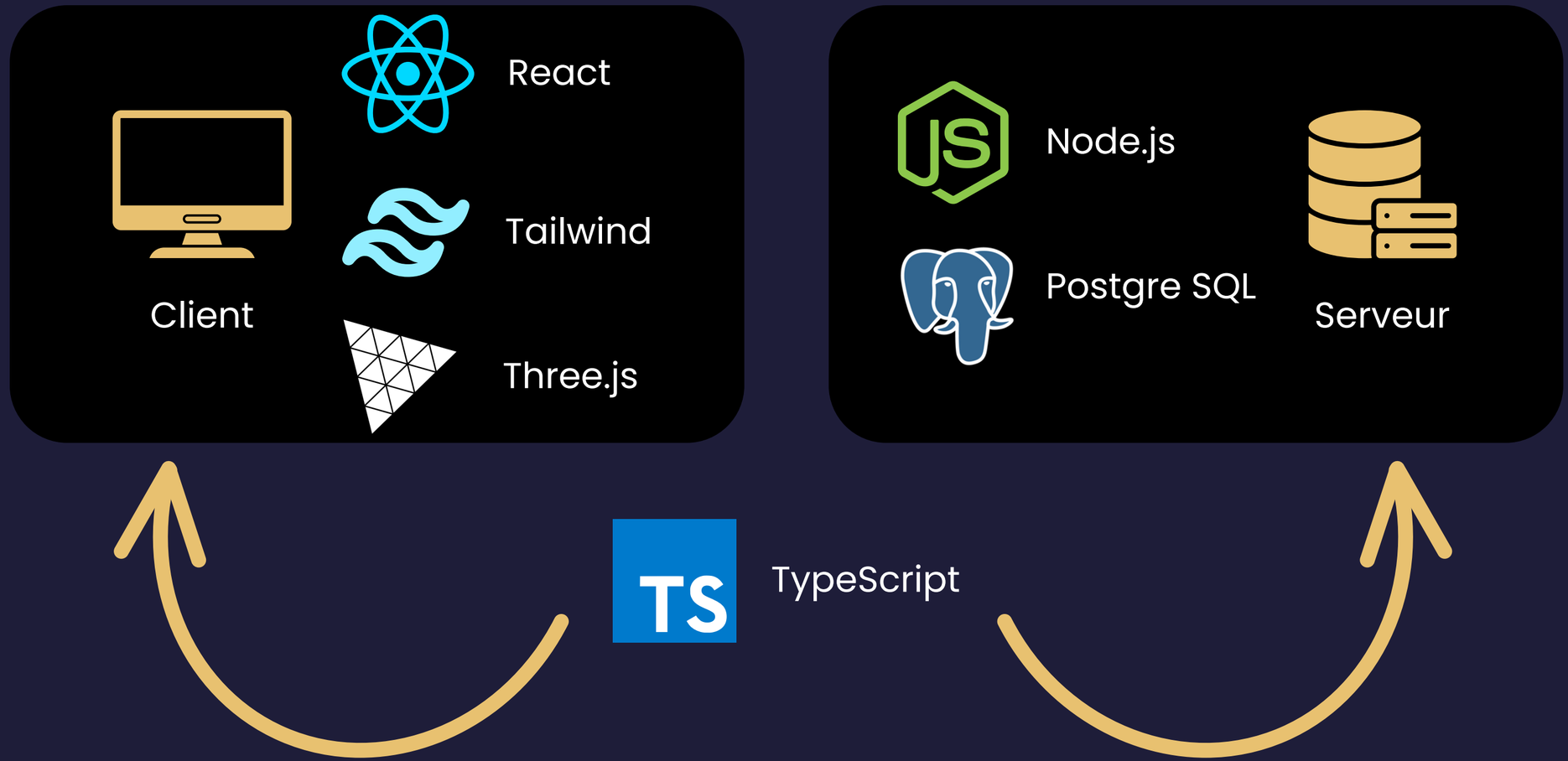
MISE EN OEUVRE

ENVIRONNEMENT

Visual Studio Code



TECHNOLOGIES DE DÉVELOPPEMENT



HEBERGEMENT

Déploiement

Test

Client

Serveur & base de données

Base de données

GitHub Pages

render.com

mkdb.sh

0 €

0 € ** → 20 €

0 €

** par mois

NOM DE DOMAINE

ploybout.com

* 10,40€

*par an, namecheap.com

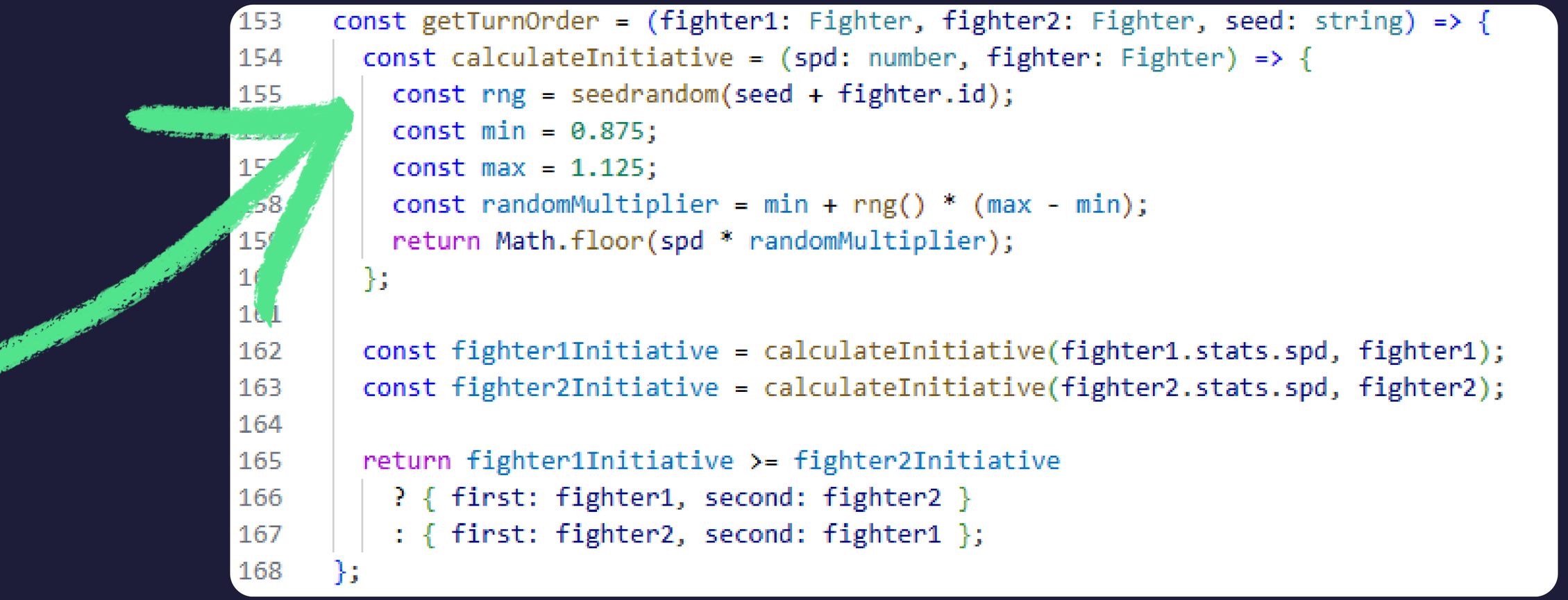
COMBAT (SERVEUR)

```
11  export const saveCombatResult = async (
12    req: Request<{}, {}, { fighter1: Fighter; fighter2: Fighter }>,
13    res: Response
14  ): Promise<void> => {
15    const { fighter1, fighter2 } = req.body;
16    if (!fighter1 || !fighter2) {
17      res.status(400).json({ error: "Invalid fighter data" });
18      return;
19    }
20
21    const seed = generateRandomSeed();
22
23    const { fighter1Id, fighter2Id, winner, combatLog } = await executeCombat(
24      fighter1,
25      fighter2,
26      seed
27    );
```

```ts
export const executeCombat = async (
  fighter1: Fighter,
  fighter2: Fighter,
  seed: string
) => {
  let updatedFighter1 = updateFighter(fighter1);
  let updatedFighter2 = updateFighter(fighter2);

  let fighter1Health = updatedFighter1.stats.hp;
  let fighter2Health = updatedFighter2.stats.hp;
  const fighter1Energy: string[] = [];
  const fighter2Energy: string[] = [];

  const maxTurns = 100;
  const combatLog = [];
  let turn = 0;

  if (!Array.isArray(fighter1.deck) || !Array.isArray(fighter2.deck)) {
    throw new Error("Invalid deck data");
  }
```

```ts
export interface Fighter {
  id: number;
  name: string;
  userId: number;
  visual: Visual;
  stats: Stat;
  deck: Card[];
  equipment: Equipment;
}
```

```
47    let state: State = {
48      fighter1Health,
49      fighter2Health,
50      fighter1Energy,
51      fighter2Energy,
52      fighter1Range,
53      fighter2Range,
54      fighter1Position: -3,
55      fighter2Position: 3,
56    };
57    while (
58      turn < maxTurns &&
59      state.fighter1Health > 0 &&
60      state.fighter2Health > 0
61    ) {
62      const currentSeed = `${seed}-${turn}`;
63
64      const { first, second } = getTurnOrder(
65        updatedFighter1,
66        updatedFighter2,
67        currentSeed
68      );
```



```
153  const getTurnOrder = (fighter1: Fighter, fighter2: Fighter, seed: string) => {
154    const calculateInitiative = (spd: number, fighter: Fighter) => {
155      const rng = seedrandom(seed + fighter.id);
156      const min = 0.875;
157      const max = 1.125;
158      const randomMultiplier = min + rng() * (max - min);
159      return Math.floor(spd * randomMultiplier);
160    };
161
162    const fighter1Initiative = calculateInitiative(fighter1.stats.spd, fighter1);
163    const fighter2Initiative = calculateInitiative(fighter2.stats.spd, fighter2);
164
165    return fighter1Initiative >= fighter2Initiative
166      ? { first: fighter1, second: fighter2 }
167      : { first: fighter2, second: fighter1 };
168  };
```

```
64    const { first, second } = getTurnOrder(
65      updatedFighter1,
66      updatedFighter2,
67      currentSeed
68    );
69
70    const players = [
71      {
72        fighter: first,
73        isPlayer: first === updatedFighter1,
74        deck: first.deck.sort((a, b) => (a.slot ?? 0) - (b.slot ?? 0)),
75      },
76      {
77        fighter: second,
78        isPlayer: second === updatedFighter1,
79        deck: second.deck.sort((a, b) => (a.slot ?? 0) - (b.slot ?? 0)),
80      },
81    ];
```

```
83    for (const { fighter, isPlayer, deck } of players) {
84      let currentCard = await getValidCard(deck, isPlayer, state);
85      if (currentCard) {
86        for (const effect of currentCard.effects || []) {
87          state = await applyCardEffects(
88            effect,
89            isPlayer,
90            state,
91            currentSeed,
92            fighter
93          );
94        }
```

```
95          combatLog.push({
96            currentFighter: fighter.id,
97            card: currentCard,
98            fighter1: {
99              health: state.fighter1Health,
100             energy: state.fighter1Energy,
101             position: state.fighter1Position,
102           },
103           fighter2: {
104             health: state.fighter2Health,
105             energy: state.fighter2Energy,
106             position: state.fighter2Position,
107           },
108           turn: turn,
109           seed: currentSeed,
110         });
111       }
112       if (state.fighter1Health <= 0 || state.fighter2Health <= 0) {
113         break;
114       }
115     }
116
117     turn++;
118   }
```

```
120     const fighter1Id = fighter1.id;
121     const fighter2Id = fighter2.id;
122     let winner: number | null;
123
124     if (state.fighter1Health > state.fighter2Health) {
125       winner = fighter1.id;
126     } else if (state.fighter1Health < state.fighter2Health) {
127       winner = fighter2.id;
128     } else {
129       winner = null;
130     }
131
132     return { fighter1Id, fighter2Id, winner, combatLog };
133   };
```

```
28    const query = `
29      INSERT INTO combats (
30        fighter1_id, fighter2_id, fighter1_name, fighter2_name,
31        winner_id, seed, combat_log, created_at
32      ) VALUES (
33        $1, $2, $3, $4, $5, $6, $7, NOW()
34      )
35      RETURNING *;
36    `;
37
38    const values = [
39      fighter1Id,
40      fighter2Id,
41      fighter1.name,
42      fighter2.name,
43      winner,
44      seed,
45      JSON.stringify(combatLog),
46    ];
47    try {
48      const result = await db.query(query, values);
49      res.status(201).json({
50        message: "Combat result saved successfully",
51        combat: result.rows[0],
52      });
53    } catch (error) {
54      console.error("Error saving combat result:", error);
55      res.status(500).json({ error: "Internal server error" });
56    }
57  }.
```

```
23    const { fighter1Id, fighter2Id, winner, combatLog } = await executeCombat(
24      fighter1,
25      fighter2,
26      seed
27    );
```

07

STRATÉGIE ET PERFORMANCES

Optimiser le SEO

Augmenté la visibilité

- Privilégier les pages publiques
- Balise méta dynamique
- Contenu sémantique (HTML)

Augmenter le trafic

- Utiliser des mots-clés pertinents
- Contenu associé

Navigation optimisée

- Mobile friendly
- Interactivité
- Améliorer les performances

HELMET

```tsx
import { Link } from "react-router-dom";
import { useAuth } from "../../hooks/useAuth";
import { Helmet } from "react-helmet-async";

export default function Home() {
  const { isLoggedIn } = useAuth();
  return (
    <>
      <Helmet>
        <title>Accueil | Ploybout</title>
        <meta
          name="description"
          content="Welcome to Ploybout, a strategic deck-building autobattler RPG! Build your fighter, customize your deck, and challenge players in epic arena battles."
        />
        <meta
          name="keywords"
          content="Ploybout, autobattler, RPG, deck-building, strategy game, online battles, card game, tactical combat, multiplayer, arena, turn-based, PvP, collectible cards, fantasy, competitive gaming"
        />
        <meta
          property="og:title"
          content="Ploybout - The Ultimate Deck-Building Arena Game"
        />
        <meta
          property="og:description"
          content="Create and customize your fighter, build the perfect deck, and challenge opponents in strategic turn-based battles!"
        />
      </Helmet>

      <div className="flex-1 flex ">
        <main className="flex flex-col items-center border-2 border-gray-200 border-opacity-10 w-full">
          <div className="flex items-center p-4 bg-gradient-to-r from-blue-500 to-purple-800 to-50% w-full h-32">
            <div className="flex flex-col w-1/2 items-center justify-center">
              <p className="text-white text-lg">
                Welcome to <em className=" not-italic text-2xl">Ploybout</em>
              </p>
              <p className="text-white opacity-50 text-md">
                Participate at joyful fights!
              </p>
            </div>
            <div className="flex flex-col w-1/2  items-center justify-center"></div>
          </div>
          {!isLoggedIn && (
            <button className="relative bottom-5 text-white bg-orange-500 w-24 py-2 text-center rounded-lg">
              <Link to="/login">Sign in!</Link>
            </button>
          )}
        </main>
      </div>
    </>
```

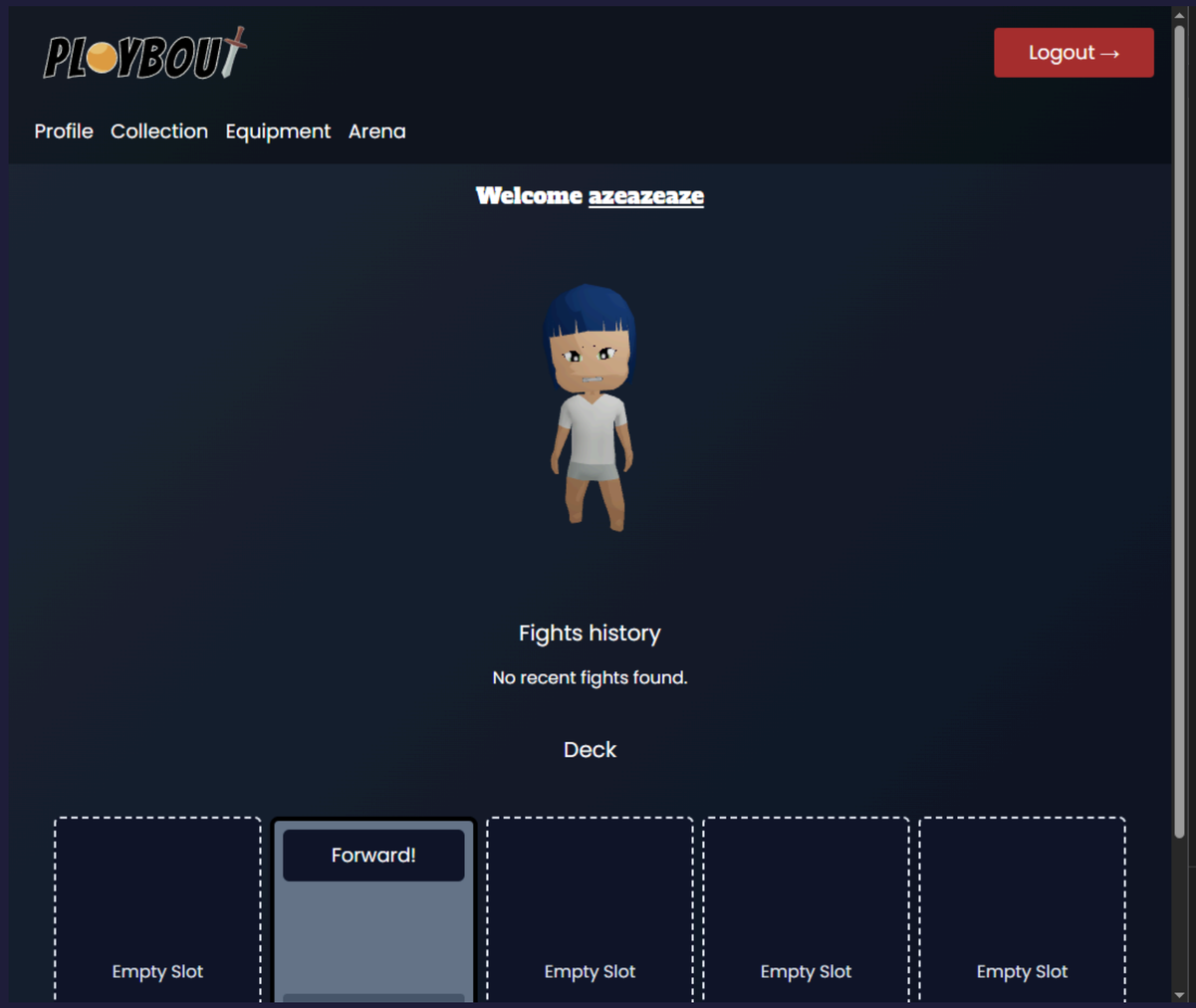
COMPATIBILITÉ DE WEBGL

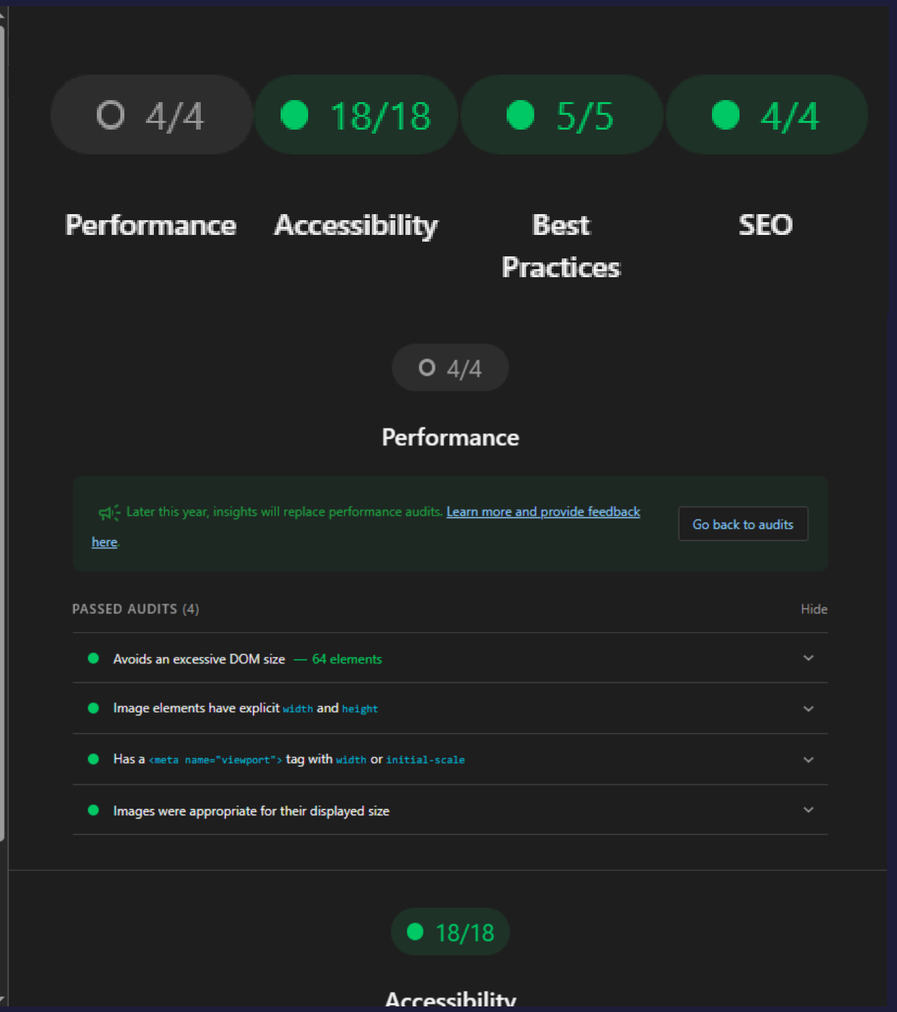
Desktop browser compatibility table

	Mozilla Firefox 42	Google Chrome 46	Apple Safari 9.0	MS Internet Explorer 11	MS Edge 13
WebGL Support	**Yes** GPU blacklists apply. WebGL may be unsupported for specific older graphics cards. Details available on the Mozilla wiki page on Blocklisting/Blocked Graphics Drivers and the Khronos wiki page on Blacklists and Whitelists.	**Yes** GPU blacklists apply. WebGL may be unsupported for specific older graphics cards. Details available on the Mozilla wiki page on Blocklisting/Blocked Graphics Drivers and the Khronos wiki page on Blacklists and Whitelists.	**Yes** Safari 8 and higher	**Yes** IE 11 and higher	**Yes**

Source : doc.unity3d.com

ANALYSE DE PERFORMANCE





TEMPS DE RÉPONSE DE L'API

```
92  useEffect(() => {
93    const fetchBattleResult = async () => {
94      if (!playerFighter || !opponentFighter) {
95        return;
96      }
97      try {
98        const start = performance.now();
99
100       const response = await axios.post(
101         `${import.meta.env.VITE_API_BASE_URL}/api/combat-result/`,
102         { fighter1: playerFighter, fighter2: opponentFighter },
103         {
104           headers: {
105             Authorization: `Bearer ${localStorage.getItem("token")}`,
106           },
107         }
108       );
109       const end = performance.now();
110       console.info(`Temps de réponse serveur : ${end - start}`);
111
112       setBattleLog(response.data.combat.combat_log);
113       setBattleResult(
114         response.data.combat.winner_id === null
115           ? null
116           : response.data.combat.winner_id === player.id
117       );
```

```
Temps de réponse serveur : 132.30000001192093
```

```
Temps de réponse serveur : 163.79999995231628
```

08 TESTS ET SÉCURITÉ

Connexion d’un utilisateur

Objectif :

Vérifier que la fonctionnalité de connexion fonctionne correctement.

Préconditions :

- L'application est connectée à la base de données.
- Un utilisateur valide est déjà enregistré.

Étapes :

1. Accéder à l'interface de connexion.
2. Saisir un nom d'utilisateur et un mot de passe valides.
3. Soumettre le formulaire de connexion.

Résultats attendus :

- L'utilisateur est authentifié avec succès.
- L'utilisateur est redirigé vers son tableau de bord.

Statut :

❌ Non validé

Rapport de bugs :

Bugs

Notion

frontend > cypress > e2e > login.cy.ts > ...

```
describe("Se connecter à un compte", () => {
  it("Devrait afficher la page de connexion", () => {
    cy.visit("http://localhost:5173/login");
  });
  it("Devrait pouvoir se connecter", () => {
    cy.visit("http://localhost:5173/login");

    cy.get('input[name="username"]').type("azeazeaze");
    cy.get('input[name="password"]').type("azeazeaze");

    cy.get('button[type="submit"]').click();

    cy.url().should("include", "/profile");
    cy.contains("azeazeaze").should("be.visible");
  });
});

```

```yaml
name: Front - Back

on:
  push:
    branches:
      - main
  pull_request:
    branches:
      - main

jobs:
  front-back-cypress:
    runs-on: ubuntu-latest

    steps:
      - name: Pull le repo du frontend
        uses: actions/checkout@v4
        with:
          path: frontend

      - name: Pull le repo du backend
        uses: actions/checkout@v4
        with:
          repository: Cemus/fil-rouge-back
          path: backend

      - name: Installation de Node
        uses: actions/setup-node@v4
        with:
          node-version: 18
          cache: "npm"

      - name: Installation des dépendances frontend
        run: npm install

      - name: Installation des dépendances backend
        run: |
          cd backend
          npm install

      - name: Lancement du backend
        run: |
          cd backend
          npm start &
        env:
          PORT: 3000
          DATABASE_URL: ${{ secrets.DATABASE_URL }}
          JWT_SECRET: ${{ secrets.JWT_SECRET }}
          DB_USER: ${{ secrets.DB_USER }}
          DB_PASSWORD: ${{ secrets.DB_PASSWORD }}

      - name: Attendre le back...
        run: npx wait-on http://localhost:3000/health --timeout 30000

      - name: Lancement du frontend
        run: npm run dev &

      - name: Attendre le front...
        run: npx wait-on http://localhost:5173

      - name: Runner les tests
        uses: cypress-io/github-action@v6
        with:
          wait-on: http://localhost:5173
          browser: chrome

      - name: Upload des screens
        if: failure()
        uses: actions/upload-artifact@v4
        with:
          name: cypress-artifacts
          path: |
            cypress/videos
            cypress/screenshots
```

```
  Running:  login.cy.ts
Still waiting to connect to Chrome, retrying in 1 second (attempt 18/62)
Still waiting to connect to Chrome, retrying in 1 second (attempt 19/62)
Still waiting to connect to Chrome, retrying in 1 second (attempt 20/62)
Still waiting to connect to Chrome, retrying in 1 second (attempt 21/62)
Still waiting to connect to Chrome, retrying in 1 second (attempt 22/62)
Still waiting to connect to Chrome, retrying in 1 second (attempt 23/62)


  Se connecter à un compte
    ✓ Devrait afficher la page de connexion (1593ms)
    ✓ Devrait pouvoir se connecter (3112ms)
  2 passing (6s)
  (Results)
  ┌──────────────────────────────────────────────────────────────────────────────────────────────┐
  │ Tests:        2                                                                              │
  │ Passing:      2                                                                              │
  │ Failing:      0                                                                              │
  │ Pending:      0                                                                              │
  │ Skipped:      0                                                                              │
  │ Screenshots:  0                                                                              │
  │ Video:        false                                                                          │
  │ Duration:     6 seconds                                                                      │
  │ Spec Ran:     login.cy.ts                                                                    │
  └──────────────────────────────────────────────────────────────────────────────────────────────┘
```

Connexion d’un utilisateur

Objectif :

Vérifier que la fonctionnalité de connexion fonctionne correctement.

Préconditions :

- L'application est connectée à la base de données.
- Un utilisateur valide est déjà enregistré.

Étapes :

1. Accéder à l'interface de connexion.
2. Saisir un nom d'utilisateur et un mot de passe valides.
3. Soumettre le formulaire de connexion.

Résultats attendus :

- L'utilisateur est authentifié avec succès.
- L'utilisateur est redirigé vers son tableau de bord.

Statut :

✅ Validé

Rapport de bugs :

Bugs

JSON WEB TOKEN

Serveur

```
// User routes
router.post("/register", register);
router.post("/login", login);
router.get("/profile", authMiddleware, getUser);
router.get("/players", getAllUsers);

//Fighter routes
router.post("/seek-fighters", authMiddleware, seekFighters);
router.post("/create-fighter", authMiddleware, createFighter);

//Card Routes
router.get("/all-cards", authMiddleware, getAllCards);
router.get("/owned-cards", authMiddleware, getOwnedCards);
router.post("/update-cards", authMiddleware, postEquippedCard);
router.get("/equipped-cards/:fighter_id", authMiddleware, getEquippedCard);
```

Serveur

```
export const register = async (req: Request, res: Response) => {
  const { username, password } = req.body;
  const client = await db.connect();

  try {
    const hashedPassword = await bcrypt.hash(password, 10);

    await client.query("BEGIN");

    const usernameCheck = await client.query(
      `SELECT username FROM users WHERE username = $1`,
      [username]
    );

    if (usernameCheck.rows.length > 0) {
      throw new Error("Username already exists");
    }

    const userResult = await client.query(
      `INSERT INTO users (username, password, created_at, updated_at)
       VALUES ($1, $2, NOW(), NOW()) RETURNING id`,
      [username, hashedPassword]
    );

    const newUserId = userResult.rows[0].id;

    await createInitialCardCollection({ id: newUserId }, client);
    await createInitialEquipmentCollection({ id: newUserId }, client);

    await client.query("COMMIT");

    const token = jwt.sign(
      { id: newUserId },
      process.env.JWT_SECRET as string,
      { expiresIn: "1h" }
    );

    res.status(201).json({ token });
  } catch (error: any) {
    await client.query("ROLLBACK");
    console.error("Error during register:", error);
```

Client

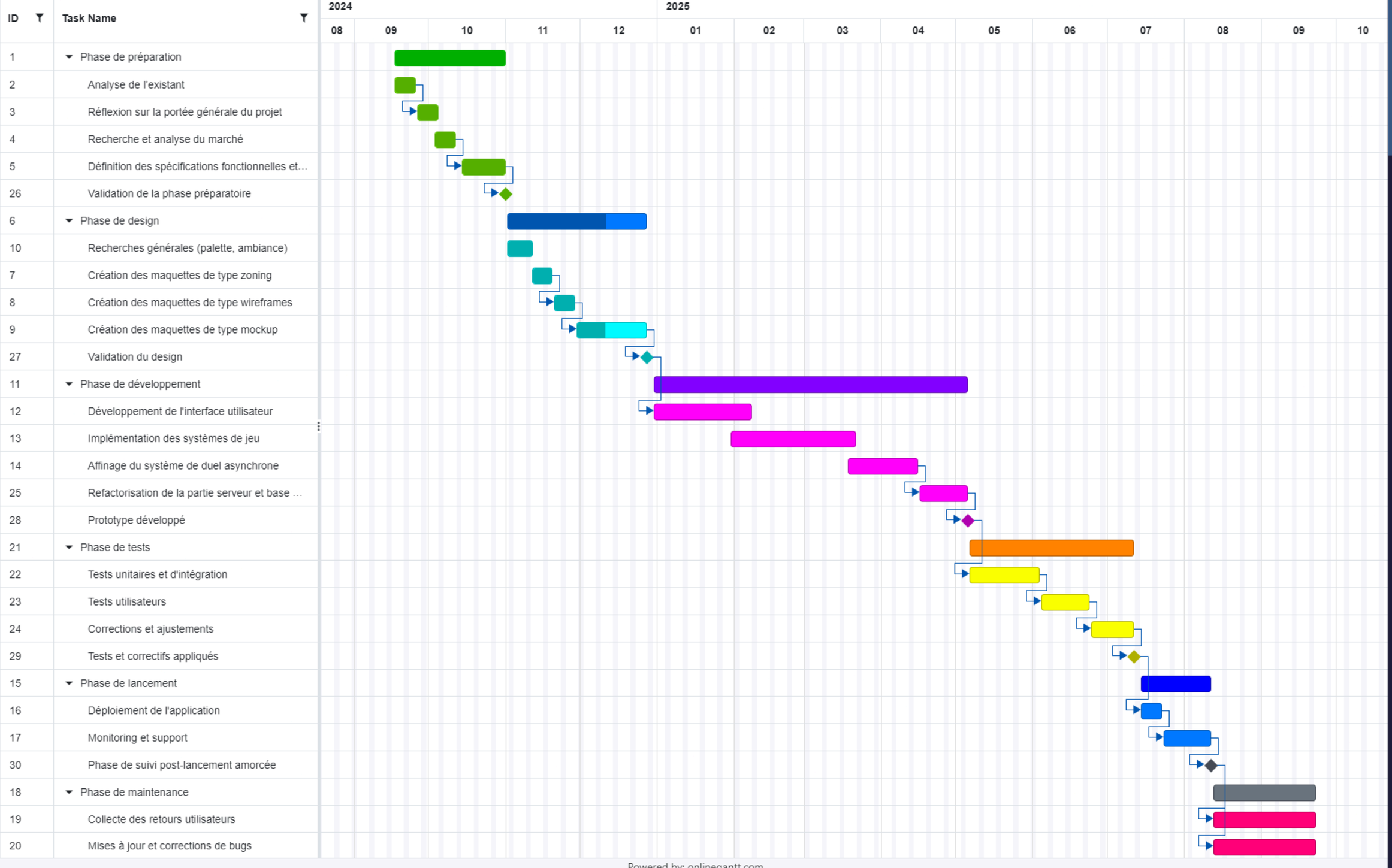
```
useEffect(() => {
  const fetchBattleResult = async () => {
    if (!playerFighter || !opponentFighter) {
      return;
    }

    try {
      const response = await axios.post(
        "/api/combat-result/",
        { fighter1: playerFighter, fighter2: opponentFighter },
        {
          headers: {
            Authorization: `Bearer ${localStorage.getItem("token")}`,
          },
        }
      );
      setBattleLog(response.data.combat.combat_log);
    } catch (error) {
      console.error(
        "Error during the battle sequence",
        error
      );
      const e = error as AxiosError;
      const errorMessage = (e.response?.data as { error?: string }).error;
      console.error(errorMessage);
      if (e.response?.status === 401) {
        stableExitSession();
      }
    }
  };

  fetchBattleResult();
}, [stableExitSession]);
```

Serveur

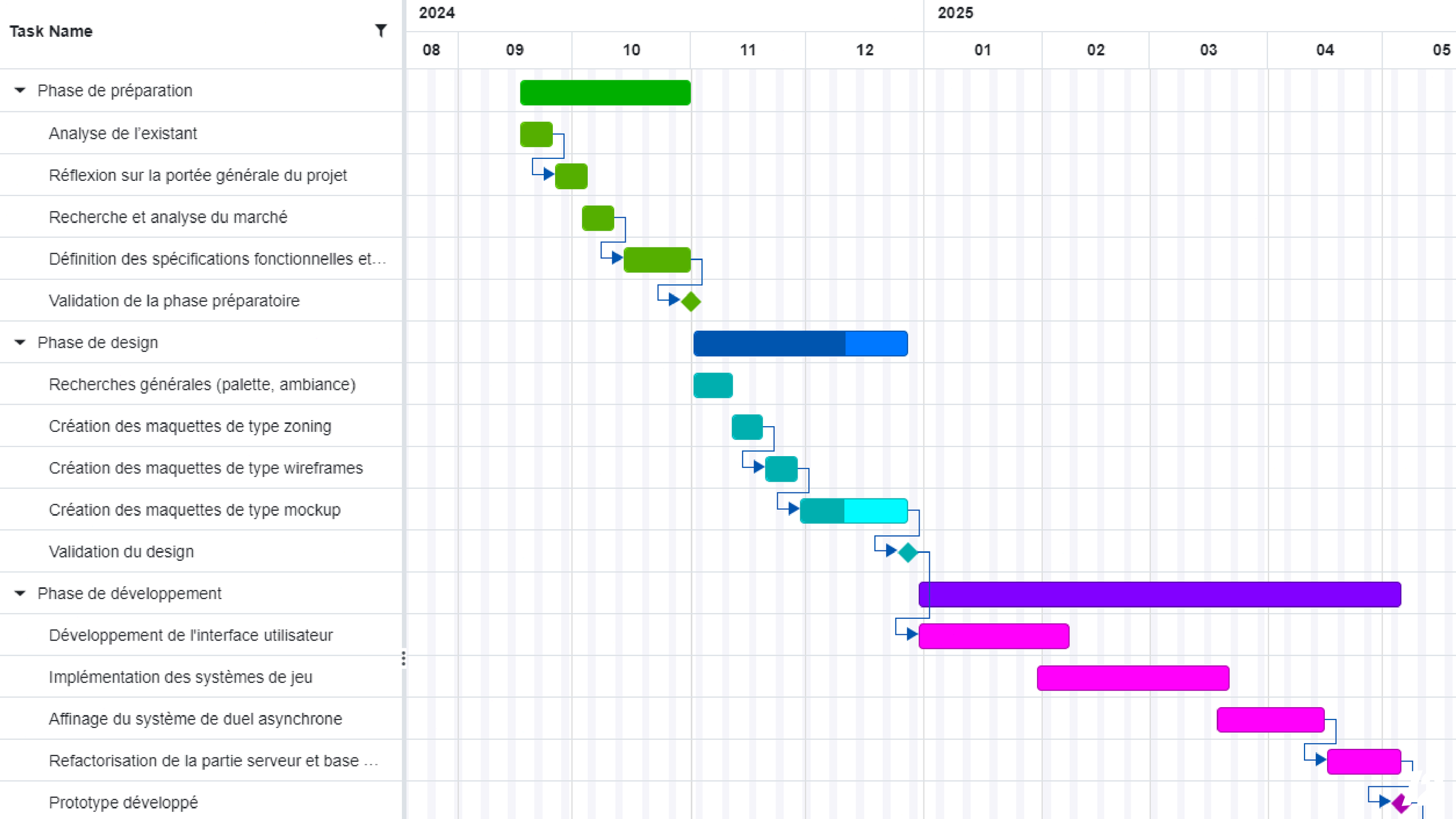
```typescript
interface TokenPayload {
  id: number;
  iat: number;
  exp: number;
}

export default function authMiddleware(
  req: Request,
  res: Response,
  next: NextFunction
): void {
  const authHeader = req.headers.authorization;

  if (!authHeader) {
    res.status(401).json({ error: "No token provided" });
    return;
  }

  const token = authHeader.split(" ")[1];

  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET!) as TokenPayload;
    (req as Request & { user?: { id: number } }).user = { id: decoded.id };
    next();
  } catch (error) {
    res.status(401).json({ error: "Invalid token" });
  }
}
```

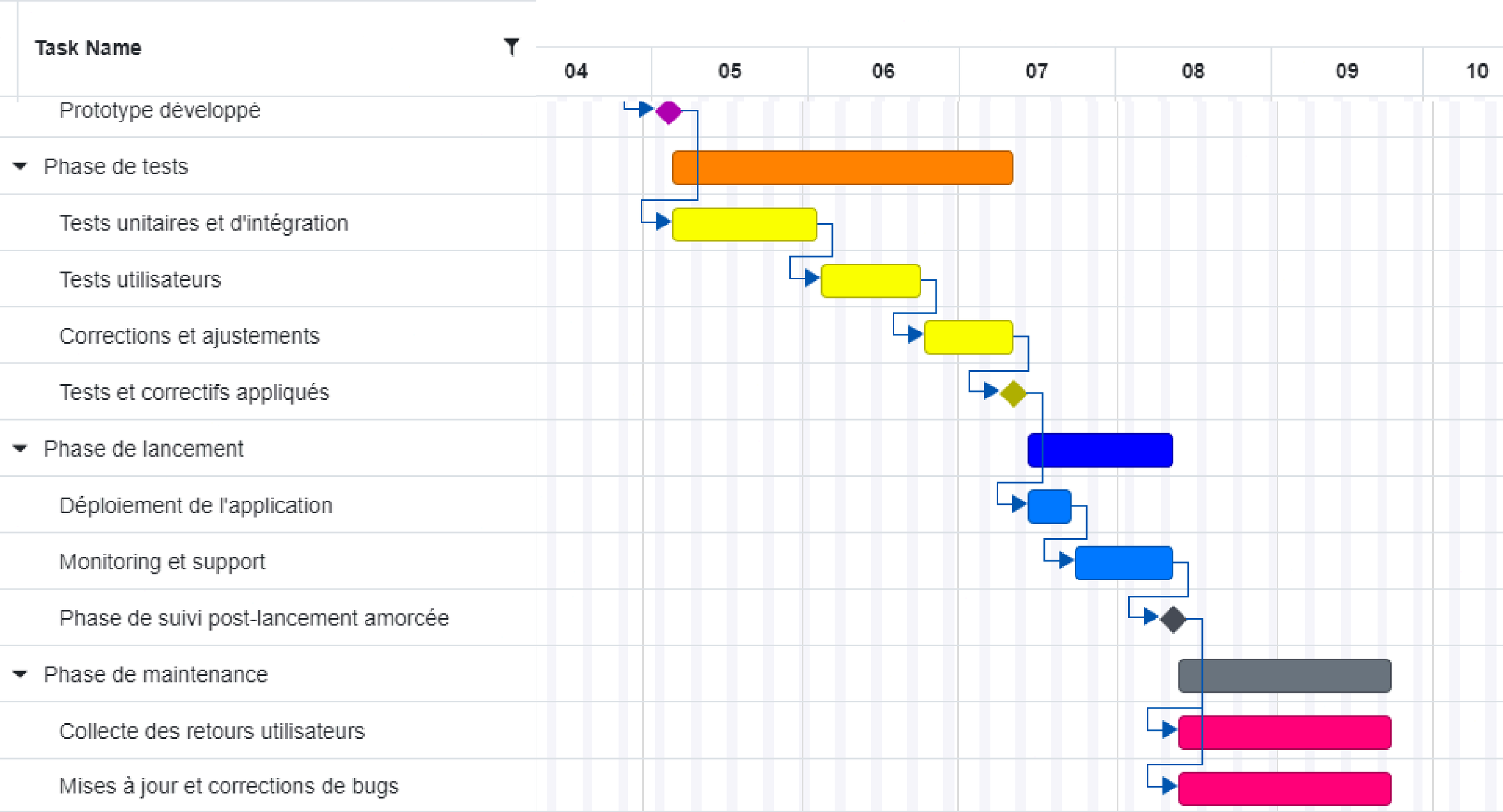
09

GESTION DE PROJET

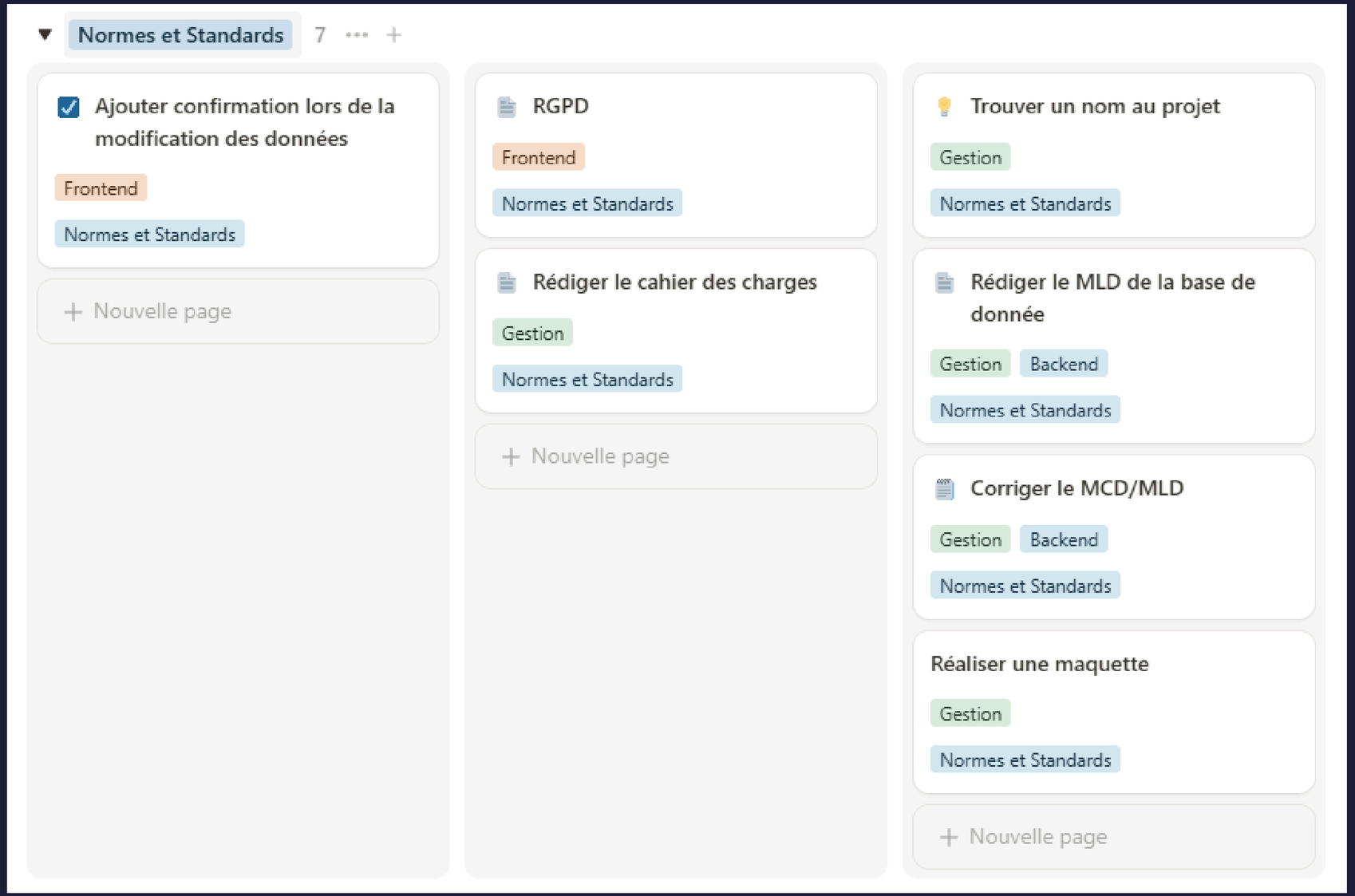


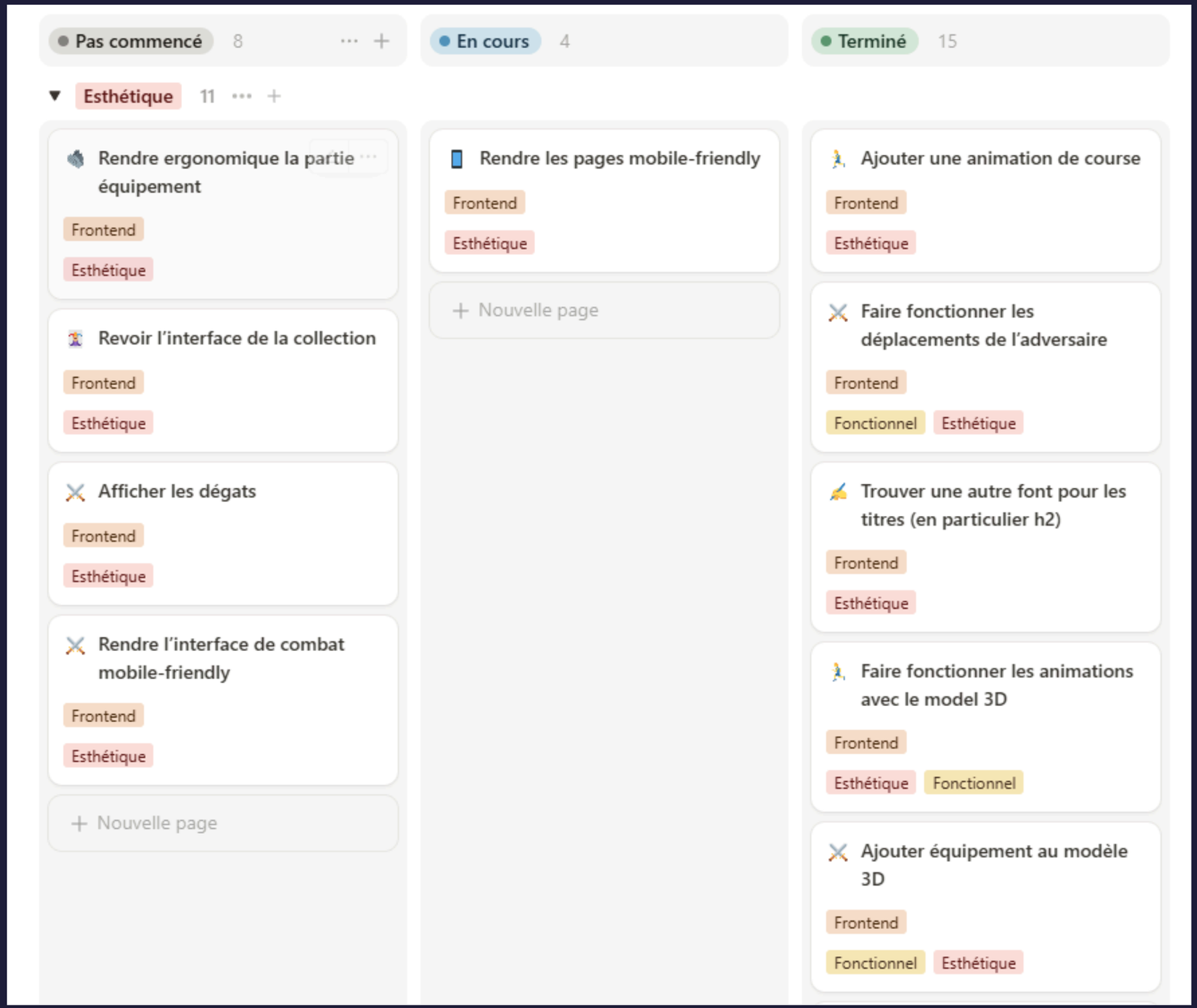
gantt-online


Task Name
2024
2025
08
09
10
11
12
01
02
03
04
05
Phase de préparation
Analyse de l'existant
Réflexion sur la portée générale du projet
Recherche et analyse du marché
Définition des spécifications fonctionnelles et...
Validation de la phase préparatoire
Phase de design
Recherches générales (palette, ambiance)
Création des maquettes de type zoning
Création des maquettes de type wireframes
Création des maquettes de type mockup
Validation du design
Phase de développement
Développement de l'interface utilisateur
Implémentation des systèmes de jeu
Affinage du système de duel asynchrone
Refactorisation de la partie serveur et base ...
Prototype développé


Task Name
04
05
06
07
08
09
10
Prototype développé
Phase de tests
Tests unitaires et d'intégration
Tests utilisateurs
Corrections et ajustements
Tests et correctifs appliqués
Phase de lancement
Déploiement de l'application
Monitoring et support
Phase de suivi post-lancement amorcée
Phase de maintenance
Collecte des retours utilisateurs
Mises à jour et corrections de bugs

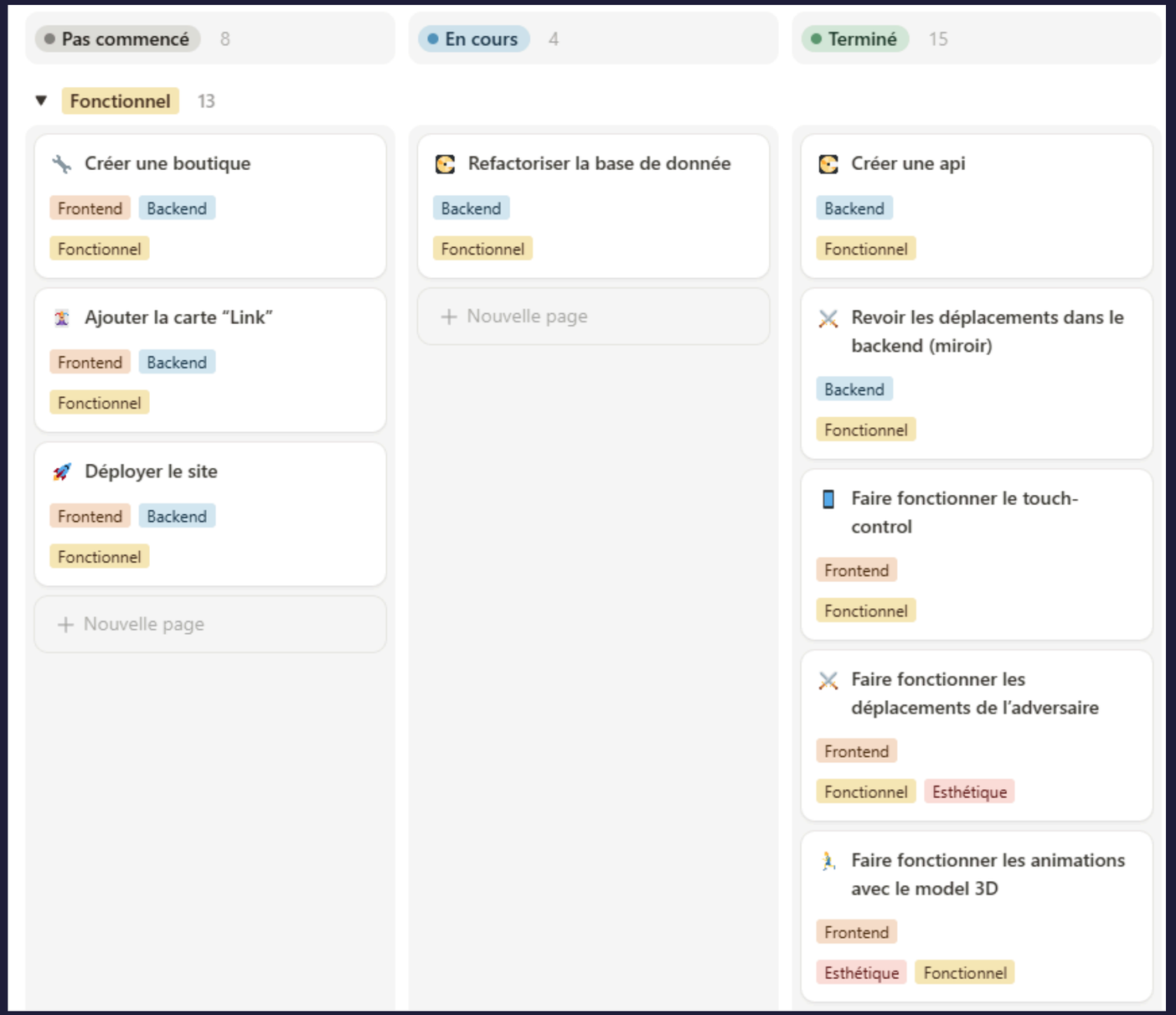
Notion


Normes et Standards 7
Ajouter confirmation lors de la modification des données
Frontend
Normes et Standards
Nouvelle page
RGPD
Frontend
Normes et Standards
Rédiger le cahier des charges
Gestion
Normes et Standards
Nouvelle page
Trouver un nom au projet
Gestion
Normes et Standards
Rédiger le MLD de la base de donnée
Gestion
Backend
Normes et Standards
Corriger le MCD/MLD
Gestion
Backend
Normes et Standards
Réaliser une maquette
Gestion
Normes et Standards
Nouvelle page


Pas commencé 8
En cours 4
Terminé 15
Esthétique 11
Rendre ergonomique la partie équipement
Frontend
Esthétique
Revoir l'interface de la collection
Frontend
Esthétique
Afficher les dégats
Frontend
Esthétique
Rendre l'interface de combat mobile-friendly
Frontend
Esthétique
Nouvelle page
Rendre les pages mobile-friendly
Frontend
Esthétique
Nouvelle page
Ajouter une animation de course
Frontend
Esthétique
Faire fonctionner les déplacements de l'adversaire
Frontend
Fonctionnel
Esthétique
Trouver une autre font pour les titres (en particulier h2)
Frontend
Esthétique
Faire fonctionner les animations avec le model 3D
Frontend
Esthétique
Fonctionnel
Ajouter équipement au modèle 3D
Frontend
Fonctionnel
Esthétique

Notion

Notion


Pas commencé 8
En cours 4
Terminé 15
Fonctionnel 13
Créer une boutique
Frontend Backend
Fonctionnel
Ajouter la carte “Link”
Frontend Backend
Fonctionnel
Déployer le site
Frontend Backend
Fonctionnel
Nouvelle page
Refactoriser la base de donnée
Backend
Fonctionnel
Nouvelle page
Créer une api
Backend
Fonctionnel
Revoir les déplacements dans le backend (miroir)
Backend
Fonctionnel
Faire fonctionner le touch-control
Frontend
Fonctionnel
Faire fonctionner les déplacements de l’adversaire
Frontend
Fonctionnel Esthétique
Faire fonctionner les animations avec le model 3D
Frontend
Esthétique Fonctionnel

10 CONCLUSION

PERSPECTIVE

- Finaliser le visuel
- Optimiser l’expérience utilisateur
- Mettre en place le système de progression
- Améliorer la sécurité
- Interface administrateur

Merci