|  |  |
| --- | --- |
|  | Institut Provincial de l’Enseignement technique Nivelles  **I.P.E.T.** |

Travail de fin d’études dans le cadre de l’option

INFORMATIQUE 7h/S

**ENSEIGNEMENT SECONDAIRE DE TRANSITION**

**Pokémon**

**Battle Royale**

Professeurs responsables :

Travail de fin d'études présenté par (**Montariol Adrien – Wery Alexandre)** dans le cadre de l’obtention du CESS

**B. Verfaille – I. Eloy**

**Année académique 2018-2019**

Table des matières

[**1.Introduction** 4](#_Toc9771329)

[**2.Interlocuteurs** 4](#_Toc9771330)

[**Directeurs de Projet :** 4](#_Toc9771331)

[**Personnes de référence :** 4](#_Toc9771332)

[**3.Répartition des tâches et planification** 5](#_Toc9771333)

[**Mars :** 6](#_Toc9771334)

[**Avril:** 7](#_Toc9771335)

[**Mai:** 8](#_Toc9771336)

[**4.Contexte** 9](#_Toc9771337)

[**4.1. Public visé** 9](#_Toc9771338)

[**4.2. But du jeu** 9](#_Toc9771339)

[**4.3. Description** 9](#_Toc9771340)

[**4.4 Règles** 11](#_Toc9771341)

[**Carte :** 11](#_Toc9771342)

[**Personnages :** 11](#_Toc9771343)

[**Contrôles :** 12](#_Toc9771344)

[**Déplacement :** 12](#_Toc9771345)

[**Combat :** 13](#_Toc9771346)

[**Progression :** 14](#_Toc9771347)

[**5.Supports utilisés** 16](#_Toc9771348)

[**6.Ressources apportées** 16](#_Toc9771349)

[**7.Structure** 17](#_Toc9771350)

[**Python :** 17](#_Toc9771351)

[**Web :** 18](#_Toc9771352)

[**8.Specification techniques** 21](#_Toc9771353)

[**Python** 21](#_Toc9771354)

[**Web** 22](#_Toc9771355)

[**9.Ebauche des prévisions des tests** 22](#_Toc9771356)

[**10.Bibliographie** 22](#_Toc9771357)

[**11. Analyse Bases de données et site (Mr Verfaille)** 23](#_Toc9771358)

[Schémas MCD/MLD pour l'application 23](#_Toc9771359)

[MCD 23](#_Toc9771360)

[MLD 24](#_Toc9771361)

[Schémas d'arborescence des pages de votre site 25](#_Toc9771362)

[Liste détaillée des objets, des évènements et/ou des fonctionnalités sur chaque page du site web 27](#_Toc9771363)

[Liste d’objets du site internet 27](#_Toc9771364)

[**12.Application : schémas et algorithmes** 28](#_Toc9771365)

[Schémas textuels de toutes les classes 29](#_Toc9771366)

[Schémas de structure globale du Programme 32](#_Toc9771367)

[13.Tables DB 33](#_Toc9771368)

[14.Chronologie des évènements 36](#_Toc9771369)

[Phase 1 : Analyse 36](#_Toc9771370)

[Phase 2 : Codage 36](#_Toc9771371)

[Phase3 : Tests 36](#_Toc9771372)

[15.Vues Finales 37](#_Toc9771373)

[Site 37](#_Toc9771374)

[Jeu 42](#_Toc9771375)

[Launcher 42](#_Toc9771376)

[Jeu 44](#_Toc9771377)

[16.Arborescences Finales 46](#_Toc9771378)

[Site Web 46](#_Toc9771379)

[Programme Python 47](#_Toc9771380)

# **1.Introduction**

Adrien Montariol

17 ans

6ème Informatique IPET

7152@site.ipet.be

Alexandre Wery

18 ans

6ème Informatique IPET

6228@site.ipet.be

Nous avons décidé de créer un jeu de type « Battle Royale » dans l’univers Pokémons. Nous avons choisi un jeu car en tant que joueurs, nous avons trouvé intéressant de comprendre les rouages du jeu vidéo, d’apprendre plus sur cet univers qui prends de plus en plus d’ampleur dans le monde aujourd’hui.

Ce TFE nous apportera de l’expérience dans la création de jeu vidéo. Le jeu que nous avons choisi nous forcera à être très rigoureux dans l’analyse. Le fait que nous créions un jeu va nous pousser à donner le meilleur de nous-même.

# **2.Interlocuteurs**

## **Directeurs de Projet :**

* Mme Eloy : [isabelleeloy@site.ipet.be](mailto:isabelleeloy@site.ipet.be)
* Mr Verfaille : [benjaminverfaille@site.ipet.be](mailto:benjaminverfaille@site.ipet.be)

## **Personnes de référence :**

* Arthur Wery : [6737@site.ipet.be](mailto:6737@site.ipet.be)
* Tiramisu Montariol : [7152@site.ipet.be](mailto:7152@site.ipet.be)

# **3.Répartition des tâches et planification**



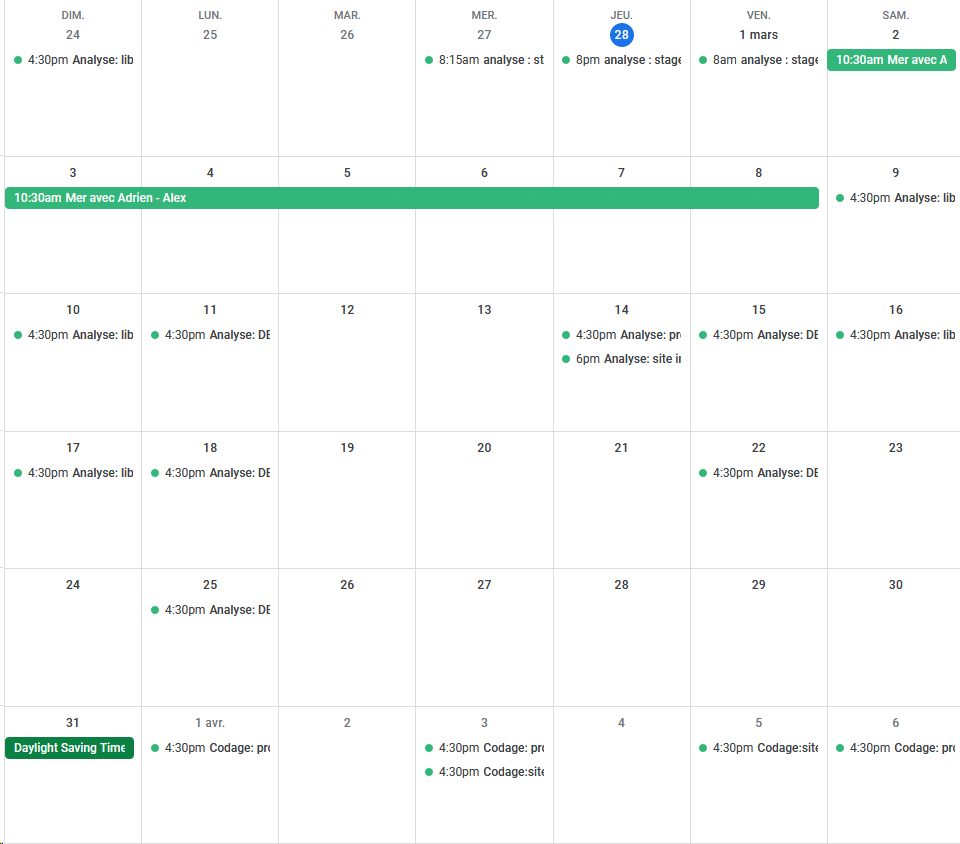
<https://calendar.google.com/calendar/r/month/2019/3/1?pli=1>

Agenda

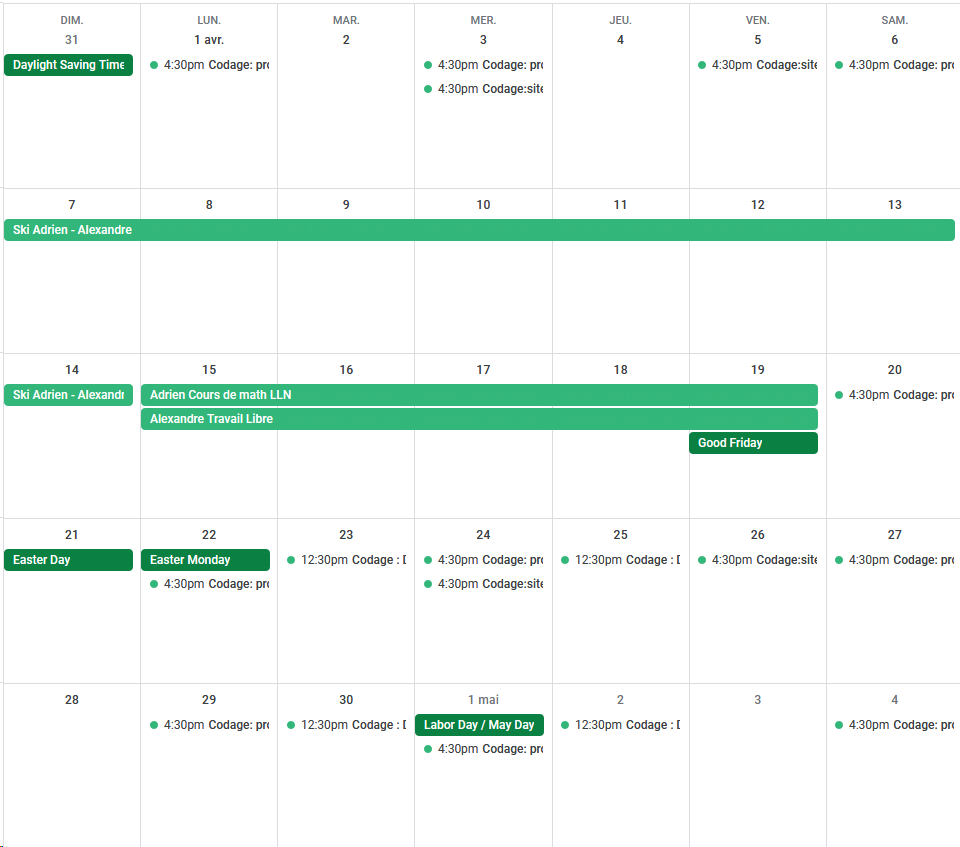
L’analyse sera faite a 2 de manière rigoureuse afin de pouvoir mieux construire notre programme et notre site par la suite.

Nous travaillerons en parallèle sur la programmation web et la programmation du jeu en tant que telle.

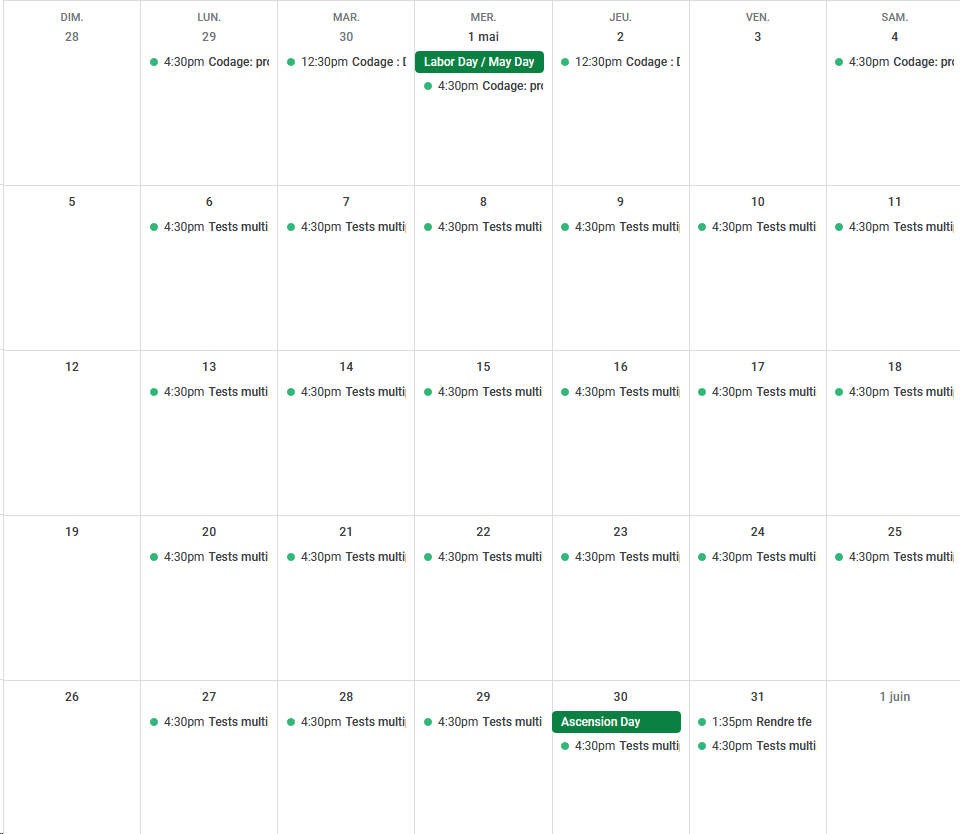
## **Mars :**



## **Avril:**



## **Mai:**



# **4.Contexte**

## **4.1. Public visé**

Notre jeu est un jeu tout public avec une prise en main très simple, il vise principalement un public de 6 à 16 ans mais peut être joué à tout âge.

## **4.2. But du jeu**

Le but du jeu est d'être le dernier joueur apte au combat. Pour être apte au combat le joueur doit posséder au moins un pokémon ayant plus de 0 point de vie.

## **4.3. Description**

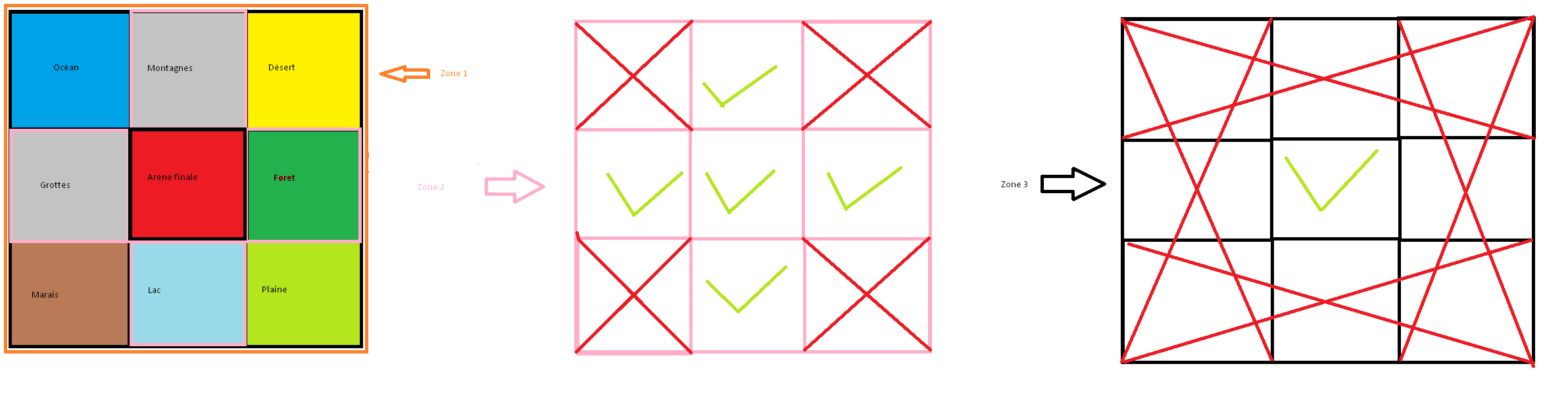
Le jeu que nous avons choisi est notre propre adaptation d’un style de jeu bien connu : “Battle royale”, mais dans l’univers Pokémon. Il s’agit donc d’un jeu d’aventure.

Le joueur physique incarne un dresseur. Les 7 autres joueurs sont des “bots”.

Un dresseur peut se déplacer et posséder des objets, des pokémons et de l'argent.

Il existe deux grandes familles de pokémons : les pokémons qui appartiendront à un dresseur et les pokémons sauvages. Seuls ces derniers peuvent être capturés par les joueurs.

Au lancement de la partie, le joueur choisit son premier pokémon. Une fois le choix terminé, une vue aérienne de la carte est affichée permettant au joueur de la visualiser.



Au départ, une safe-zone englobe les 9 zones qui composent la carte.

Le joueur physique (= dresseur) et les 7 “bots” sont placés aléatoirement sur une des 8 zones extérieures prédéfinies de la carte. Il ne peut y avoir plus d’un joueur par zone lors du lancement du jeu.

Les 8 joueurs se mettent en quête de pokémons afin de former une équipe composée d’un à six pokémons. Les pokémons sont “cachés” aléatoirement dans certaines parties de la zone où le joueur se promène.

Dans leurs quêtes, les joueurs peuvent rencontrer des “personnages non joueur” (= PNJ). Ces PNJ sont soit des dresseurs qui peuvent être provoqués en duel, soit des villageois qui peuvent offrir des objets.

Les équipes de pokémon de chaque joueur peuvent combattre les unes contre les autres ou contre des pokémons sauvages selon certaines règles. Elles peuvent aussi décider d’être passives.

Les combats se déroulent sous la forme d’un tour par tour.

Lors du combat, le dresseur peut choisir d'attaquer, d’utiliser des objets (ex : potion de soin), de fuir ou de changer de pokémon.

Le combat s'arrête quand le ou les pokémons adverses sont tous K.O. Si l’adversaire est un des 8 joueurs, il est éliminé.

A la fin du combat, le pokémon victorieux reçoit de l'expérience et le dresseur peut recevoir de l’argent ou des objets.



## **4.4 Règles**

### **Carte :**

La disposition de la carte est définie aléatoirement en suivant un paterne :

* 9 zones différentes entourée d’une safe zone.
* La safe zone rétrécit en fonction du temps de jeu.
* Chaque zone est reliée aux zones adjacentes par des tunnels.

### **Personnages :**

Les personnages sont spécifiques à chaque zone

* Joueur :
  + Il existe un joueur physique et 7 bots.
  + Le joueur est un dresseur.
  + Les dresseurs apparaissent selon les régions de manière aléatoire.
  + Le joueur a 10 secondes pour choisir parmi les 3 pokémons de départ : Carapuce, Salamèche et Bulbizarre.
* PNJ :
  + Villageois :
    - Peut donner de l’argent ou des objets.
    - Ne possède pas de pokémons et donc ne peut pas être provoqué en duel.
  + Dresseurs :
    - Possède 1 à 6 pokémons.
    - Une fois vaincu, peut donner de l’argent ou des objets.
* Pokémons :
  + Statut :
    - Pokémons sauvages :
      * Les pokémon sauvages ont une chance d'apparaître dans certaines zones.
      * Le niveau des pokémons sauvages augmente en fonction des niveaux des pokémons du joueur et du temps
    - Pokémons dressés :
      * Ils peuvent apprendre 1 à 4 capacités à la fois
      * Certains paliers de niveaux permettent d’apprendre une capacité
      * Chaque niveau gagné augmente les points de vies et
      * Les capacités des pokémons peuvent infliger des dégâts, soigner le pokémon en question, infliger des effets de confusion, paralysie, endormissement, ...

* + Types de pokémons :
    - Les pokémons ont tous 1 à 2 types
    - Certains pokémons sont plus ou moins forts contres d’autres selon leurs types

Tables des multiplicateurs de dégâts en fonction des types:

### **Contrôles :**

* Le joueur se déplace de manière verticale et horizontale grâce aux touches ZQSD ou aux flèches directionnelles
* Pour déclencher un combat, le joueur doit s’approcher d’un autre dresseur et presser la touche E
* Pour accéder au sac, le joueur presse la touche I
* Pour accéder au menu jeu, le joueur presse la touche Esc

### **Déplacement :**

* Le joueur se déplace de manière verticale et horizontale grâce aux touches ZQSD ou flèches directionnelles
* Le joueur ne peut pas traverser les obstacles physiques
* Capacités de déplacement :

Les capacités peuvent êtres apprises à certains pokémons :

* -Vol permet de voyager instantanément au centre d’une des zones après une canalisation de 10 secondes qui peut être interrompue par provocation en duel. Vol peut être apprise par les pokémons : Dracaufeu, Roucool, Roucoups, Roucarnage, Piafabec, Rapasdepic, Nosferapti, Nosferalto, Canarticho, Doduo, Dodrio, Ptéra, Artikodin, Electhor, Sulfura, Dracolosse, Mew
* Tunnel permet de revenir instantanément au dernier centre visité après une canalisation de 10 secondes qui peut être interrompue par une provocation en duel.
* Teleport permet de revenir instantanément au dernier centre visité après une canalisation de 10 secondes qui peut être interrompue par une provocation en duel

### **Combat :**

Le combat se déroule sous la forme d’un tour par tour

* Le dresseur ne peut effectuer qu’une action par tour :
  + Attaquer avec son pokémon
  + Utiliser un objet
  + Changer de pokémons
  + Fuir le combat.
* Le pokémon ayant la vitesse la plus élevée attaque en premier
* Une capacité peut :
  + Se canaliser sur plusieurs tours
  + Infliger des dégâts
  + Soigner son pokémon
  + Augmenter son attaque/réduire l’attaque de l’adversaire
  + Augmenter sa défense /réduire la défense de l’adversaire
  + Paralyser son adversaire
  + Rendre son adversaire confus
  + Endormir son adversaire
* Lors d’un combat contre un autre dresseur, le joueur peut utiliser ces objets :
  + -potion :
    - Normale : régénère 20 PV d'un pokémon
    - Super : régénère 50 PV d'un pokémon
    - Hyper : régénère 200 PV d'un pokémon
    - Max : régénère tous les PV d'un pokémon
  + -Rappel :
    - Normal : ranime un pokémon et lui restaure la moitié de ses PV
    - Max : ranime un pokémon et lui restaure tous ses PV.
* Lors d’un combat contre un pokémon sauvage, le joueur peut effectuer les mêmes actions que contre un dresseur mais peut aussi tenter de le capturer à l’aide de Balls
  + Poké Ball : Objet permettant la capture d’un pokémon sauvage
  + Super Ball : Objet permettant la capture d’un pokémon sauvage ayant un taux de réussite de capture plus élevé que la Poké ball
  + Hyper Ball : Objet permettant la capture d’un pokémon sauvage ayant un taux de réussite de capture plus élevé que la Super ball
  + Master Ball : Objet permettant la capture d’un pokémon sauvage ayant un taux de réussite de capture de 100%
* Lors d’une victoire de combat, le joueur peut potentiellement gagner :
  + De l’expérience afin de faire évoluer le pokémon ayant battu le pokémon de l'adversaire
  + Des objets comme des potions, des balls ou des rappels
  + De l’argent pour s’acheter des objets ou des capacités pouvant être apprises à certains pokémons compatibles



### **Progression :**

* Pokémons :
  + Pour chaque combat remporté le pokémon reçoit de l’expérience, au bout d’un nombre de point d'expérience propre à chaque pokémon, il gagne un niveau
  + Après avoir gagné un certain nombre de niveaux :
    - Il peut apprendre des attaques
    - Il peut évoluer en un pokémon plus puissant
    - Il gagne de la vie
    - Il gagne de la vitesse
    - Il gagne de la puissance (+attaque, +défense)
* Joueur :
  + Le joueur progresse en :
    - Capturant des pokémons
    - Battant des dresseurs
    - Faisant évoluer ses pokémons



# **5.Supports utilisés**

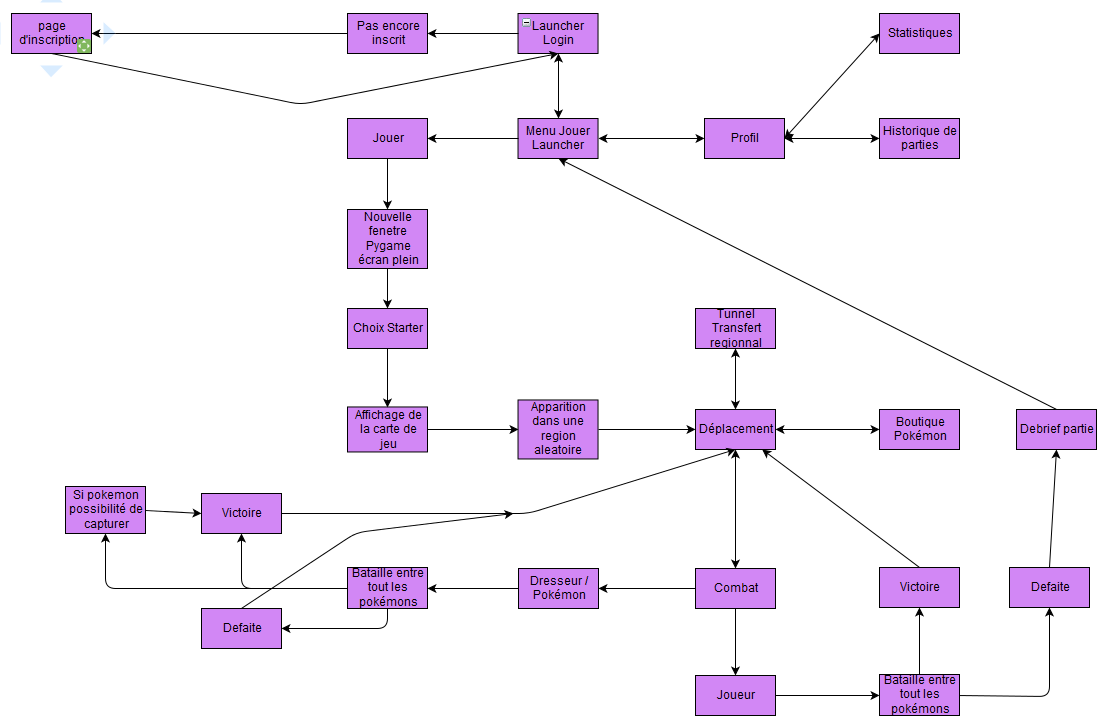
* Draw.io
* Visual Paradigm Online
* Tkinter
* Pygame
* MVC (Modèle – Vues – Contrôleur).
* MCD/MLD
* PHP/HTML/CSS
* MySQL
* PowerPoint
* Sublime Text 3
* MySQL Lite
* Wamp Server
* Google Docs - Microsoft Word

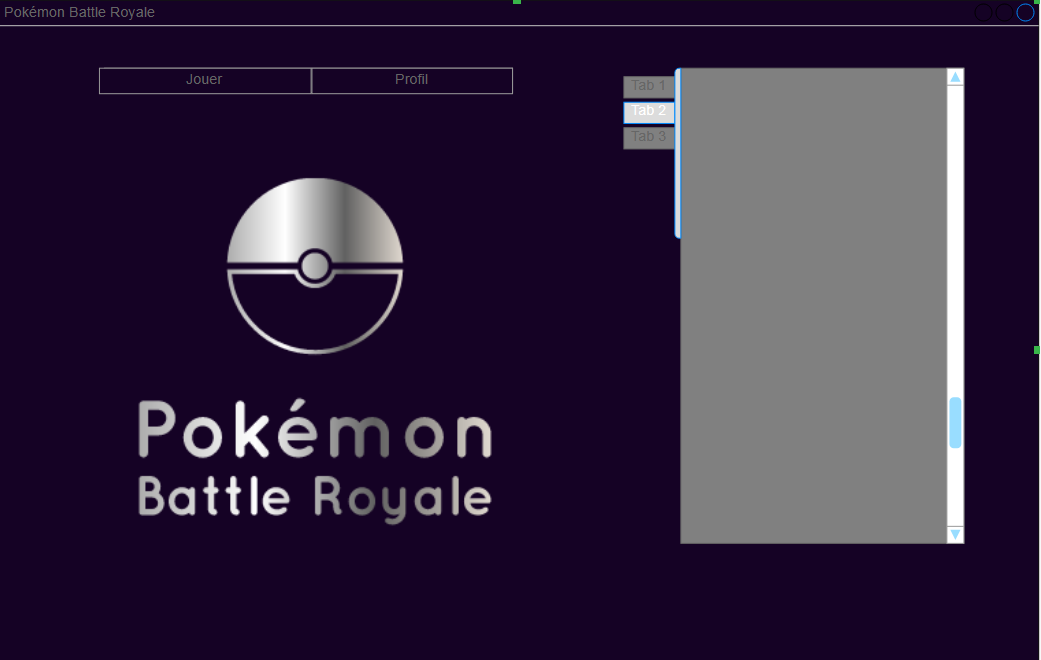
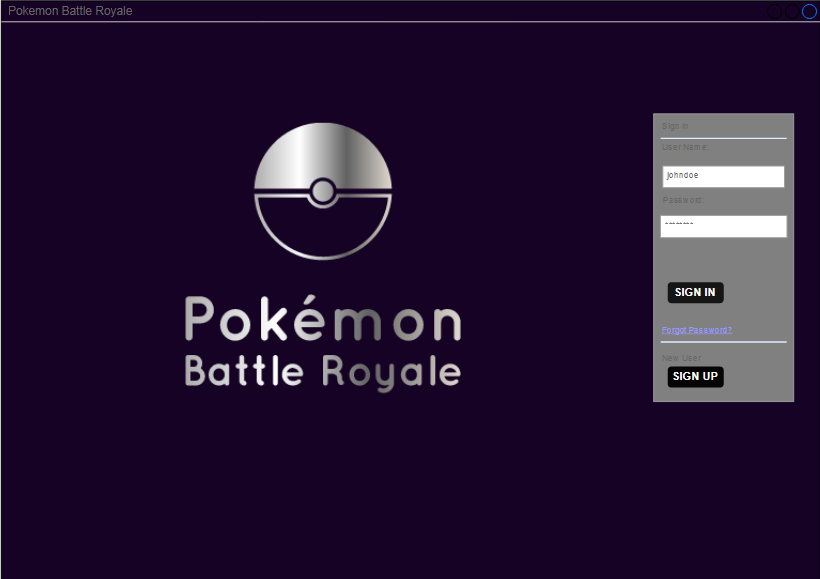
# **6.Ressources apportées**

* Première génération de Pokémons
* Sites comme W3Schools et openclassrooms et autres aides
* Bootstrap et autres Framework CSS et JS
* Présentation PowerPoint de Mr Lorie
* Cours de Labo Info, Labo Logique, Informatique et leurs supports de cours
* Dossier de présentation du TFE

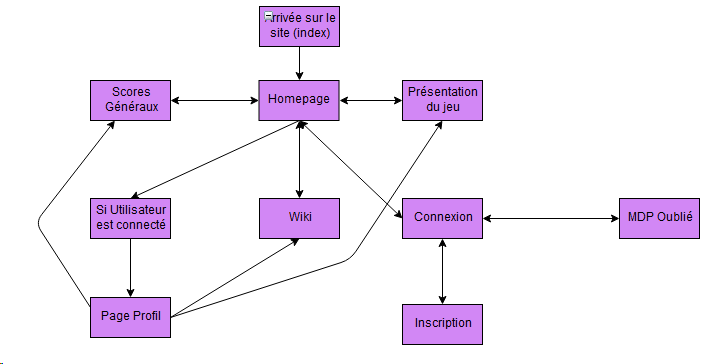
# **7.Structure**

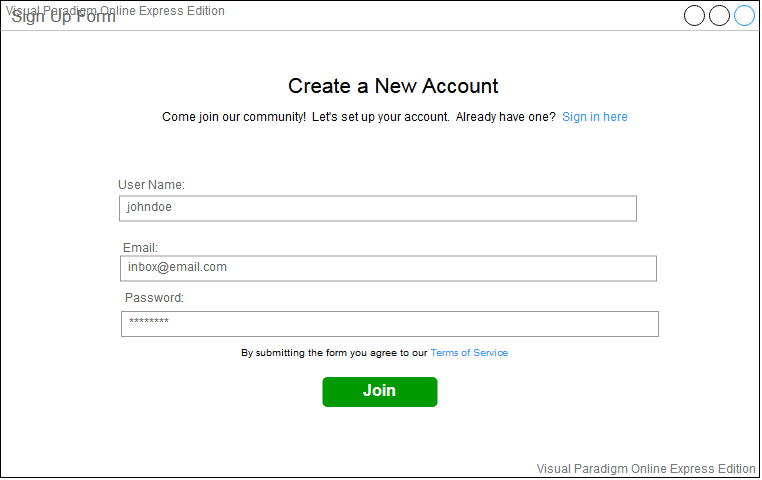
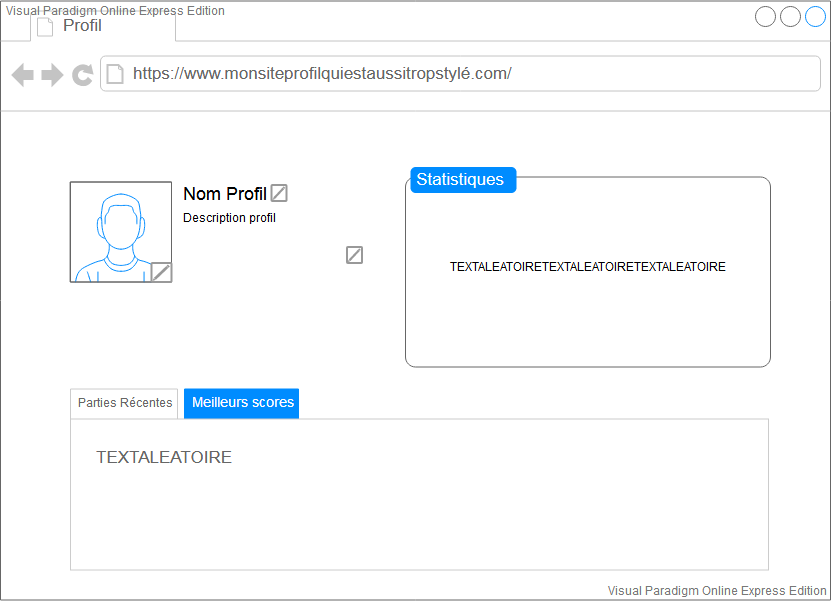
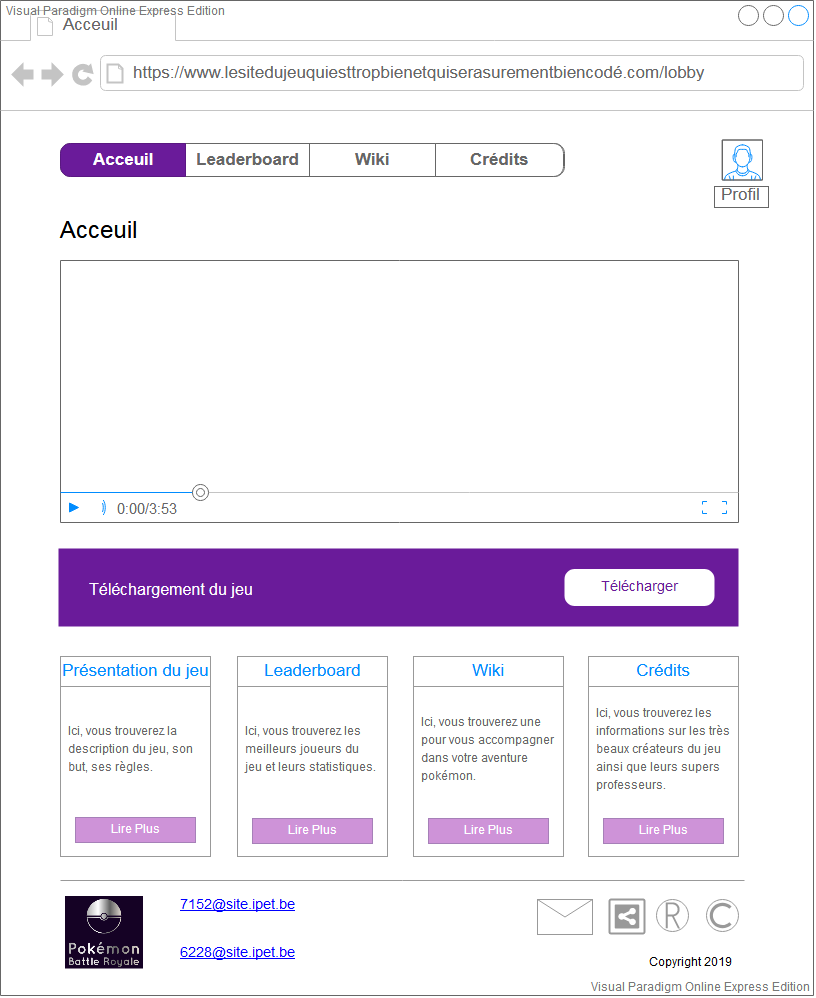
## **Python :**

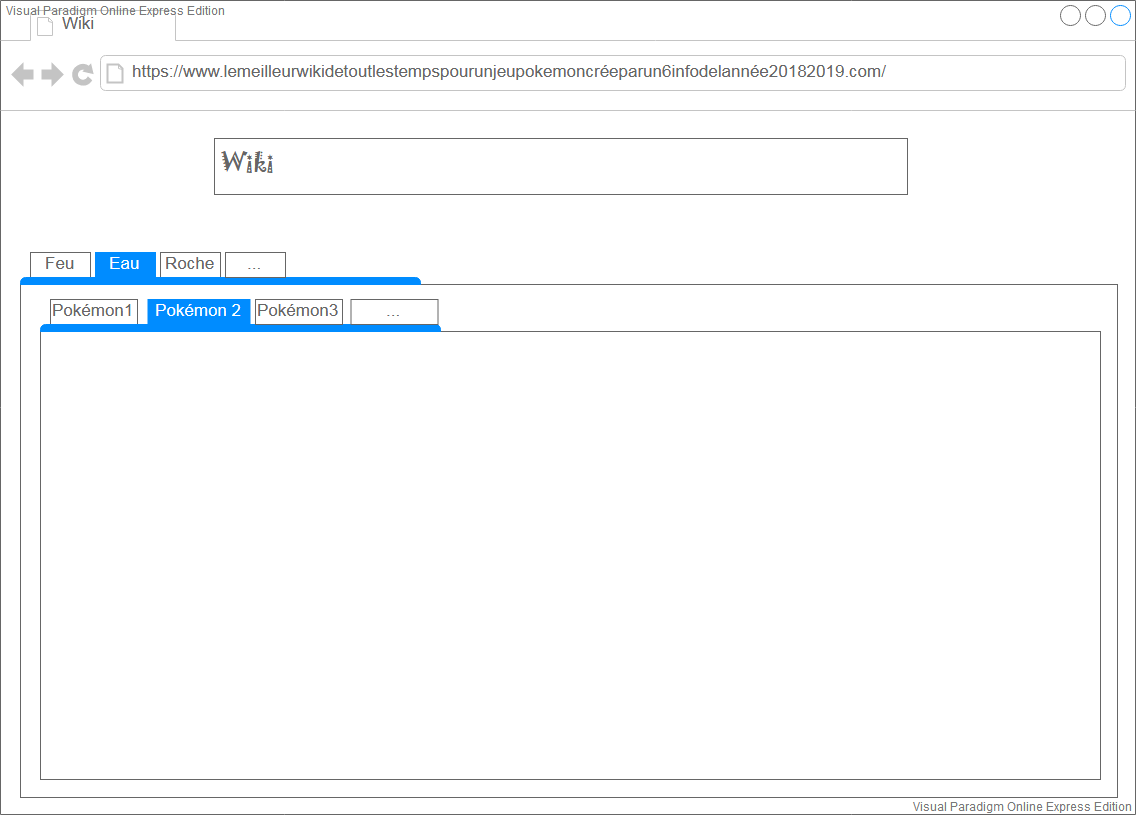




## **Web :**







# **8.Specification techniques**

## **Python**

* Lors du lancement du jeu, ouverture du launcher et demande de connexion
* Possibilité d’inscription
* Page d’accueil Tkinter :
  + Bouton jouer
  + Bouton aide
* Jouer :
  + Lancement de la fenêtre Pygame
  + Demande du choix de starter
  + Affichage de la carte du jeu
  + Arrivée dans le jeu en tant que telle
  + Déplacement libre dans la région
  + Combats si rencontre avec un personnage hostile
  + Déplacement de régions en régions
  + Rétrécissement de la safe zone si le compteur arrive a 0 (répète 2 fois)
  + Si safe zone arrive au niveau 3 :
    - Combat entre tous les joueurs restants
  + Détermination d’un vainqueur
* Aide :
  + Règles du jeu
  + Statistiques basiques
* Possibilité de quitter le jeu à tout moment
* Pas de sauvegarde locale du jeu

## **Web**

* Arrivée sur la page d’accueil
* Possibilité d’accéder à certaines pages du site sans être inscrit
* Demande de connexion
* Si pas inscrit, possibilité de s’inscrire
* Possibilité de récupérer le mot de passe
* Interaction avec la base de données pour le tableau des scores, pour les logins et mots de passes

# **9.Ebauche des prévisions des tests**

Prévision des cas pour l’application :

* Plantage lié à l’encodage des données utilisateur
* Déplacement dans l’eau sans Pokémons adaptés
* Déplacement à travers les obstacles qui lui sont interdits
* Changement des pokémons KO par combat
* Élimination après une défaite

Prévision des cas pour le site :

* Plantage lié à l’encodage des données utilisateur
* Accès aux pages privées sans être connecté
* Modification non autorisée des bases de données

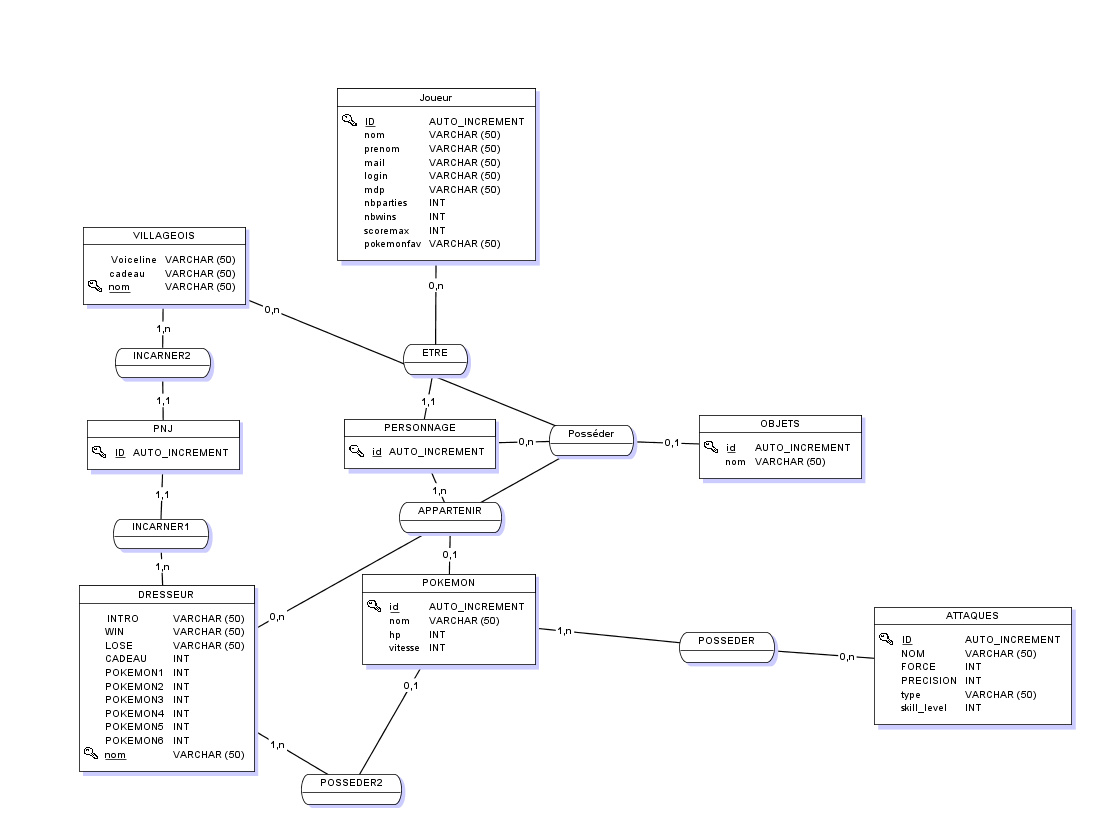
# **10.Bibliographie**

* Maintes sources provenant de google images
* Autres images créées nous-mêmes

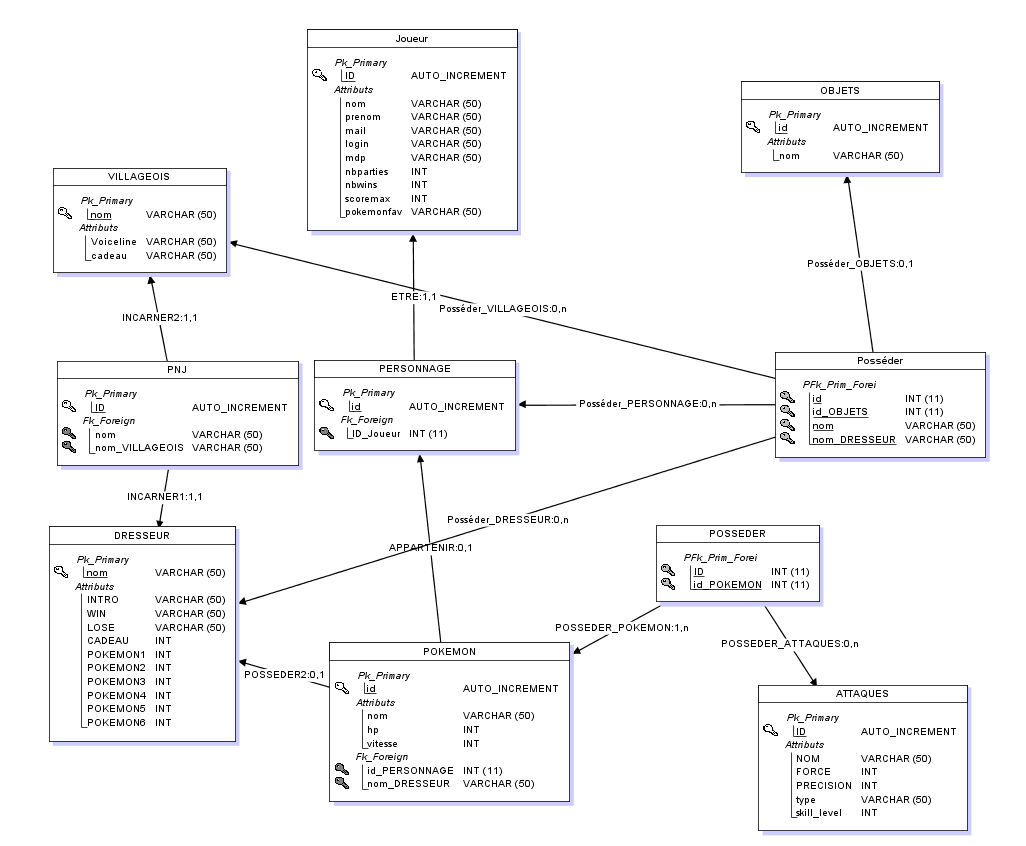
# **11. Analyse Bases de données et site (Mr Verfaille)**

## Schémas MCD/MLD pour l'application

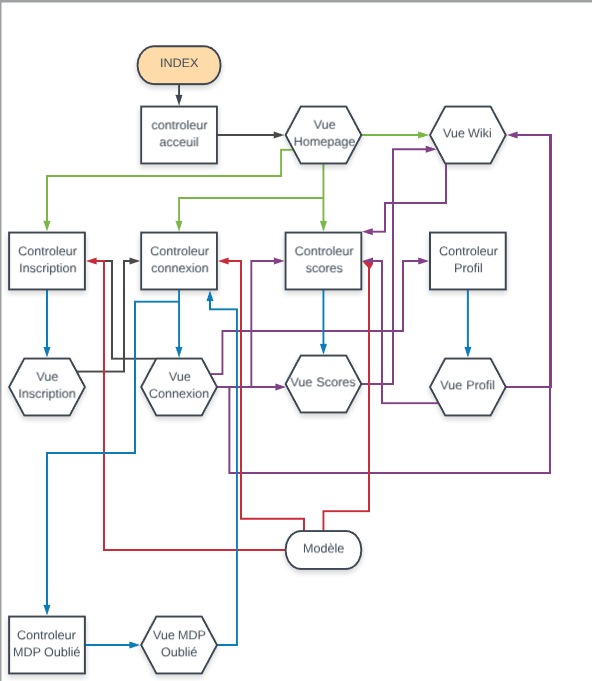
### MCD

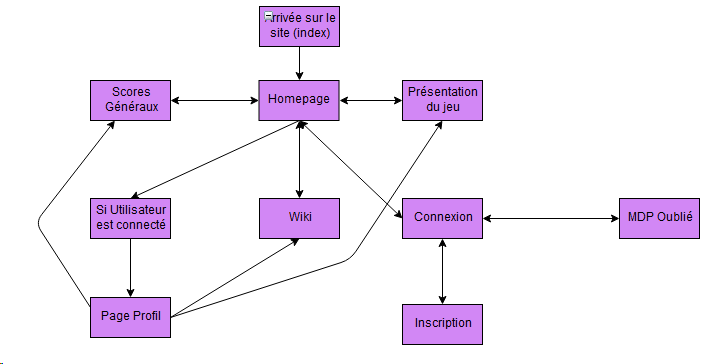


### MLD



## Schémas d'arborescence des pages de votre site



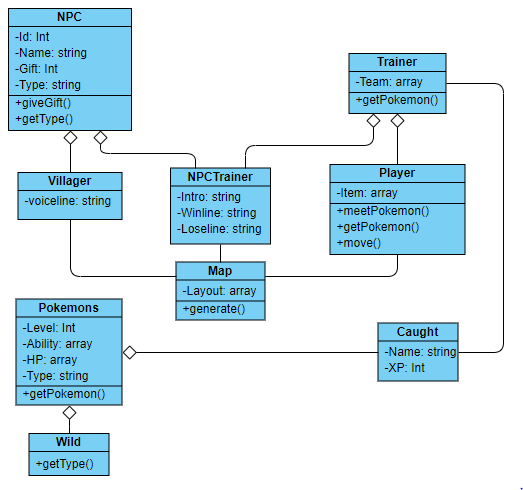


## Liste détaillée des objets, des évènements et/ou des fonctionnalités sur chaque page du site web

### Liste d’objets du site internet

* Page d'accueil
  + Boutons de navigation pour accéder aux différentes pages
  + vidéo de présentation du jeu
  + label présentant un court résumé des pages et y comprenant un lien
  + Bouton permettant de se connecter ou se déconnecter
* Page Profil
  + Boutons de navigation pour accéder aux différentes pages
  + Photo de profil
  + statistiques personnelles
  + parties récentes
* Inscription
  + Boutons de navigation pour accéder aux différentes pages
  + label pour les différents champs
  + bouton s’inscrire
* Connexion
  + Boutons de navigation pour accéder aux différentes pages
  + label d’entrée de login et de mot de passe
  + bouton connexion
* Stats
  + Boutons de navigation pour accéder aux différentes pages
  + labels avec les différents “High Scores” des joueurs
* Wiki
  + Boutons de navigation pour accéder aux différentes pages
  + barre de recherche dans le wiki
  + navbar pour le wiki
  + labels vers les différents articles du wiki

# **12.Application : schémas et algorithmes**

UML de classes Interface et Application

# 

## Schémas textuels de toutes les classes

Classe

* nom : PNJ
* role : classe mere permettant d'instancier des PNJs

Attributs:

* identifiant : entier pour différencier les différents PNJs
* nom : string pour stocker le nom du PNJ instancié
* cadeau : entier pour stocker l'ID du cadeau donné au joueur

Méthodes :

* donnerCadeau : donner le cadeau de getCadeau au joueur (l'ajouter dans le sac)

---------------------------------------------------------------------------------------------------

Classe

* Nom : DresseurPNJ
* Role: classe fille hérite de PNJ et de Dresseur permettant d'instancier des PNJs dresseurs

Attributs:

* intro : string pour stocker la "voiceline" d'intro du dresseur PNJ
* winline : string pour stocker la "voiceline" de victoire du dresseur PNJ
* loseline : string pour stocker la "voiceline" de défaite du dresseur PNJ

Méthodes :

* /

---------------------------------------------------------------------------------------------------

Classe

* Nom: Dresseur
* Rôle: permet de créer des dresseurs

Attributs

* Equipe: Tableau qui comprend 1 à 6 pokémons qui composent l’équipe

Méthodes :

* getPokemon : renvoie un pokémon récupéré de la base de données et ses attributs

---------------------------------------------------------------------------------------------------

Classe

* Nom: Joueur
* Rôle: permet de stocker toutes les informations concernant l’avancée du joueur dans une partie
* Héritage de la classe mère Dresseur

Attributs

* Objets: Dictionnaire comprenant chaque objet ainsi que sa quantitée

Méthodes:

* rencontrer Pokémons : méthode s'enclenchent dans certaines zones de la carte ayant 1 chance sur 10 par case de jeu d'appeler la méthode getPokemon

---------------------------------------------------------------------------------------------------

Classe

* Nom: Pokémons
* Rôle: permet de créer des pokémons

Attributs

* Niveau: Entier allant de 1 à 100 permettant de déterminer le niveau du pokémon
* Capacités: Liste comprenant 1 à 4 capacités différentes
* vie : Liste pour stocker la vie actuelle du pokémon
* type : string pour stocker le type du pokémon instancié

Methodes:

* getPokemon : getPokemon : renvoie un pokémon récupéré de la base de données et ses attributs

---------------------------------------------------------------------------------------------------

Classe

* Nom: Sauvage
* Rôle: permet de créer des pokémons sauvages
* Héritage de la classe mère Pokémons

Attributs:

* /

Méthodes:

* /

---------------------------------------------------------------------------------------------------

Classe

* Nom: Dressé
* Rôle: permet de créer des pokémons dressés

Attributs

* Nom: Chaîne de caractères contenant le nom du pokémon si le joueur a décidé de le renommer
* Expérience: Entier contenant le nombre de points d'expérience du pokémon

Méthodes :

* /

---------------------------------------------------------------------------------------------------

Classe

* Nom: Carte
* Rôle: permet de créer la carte

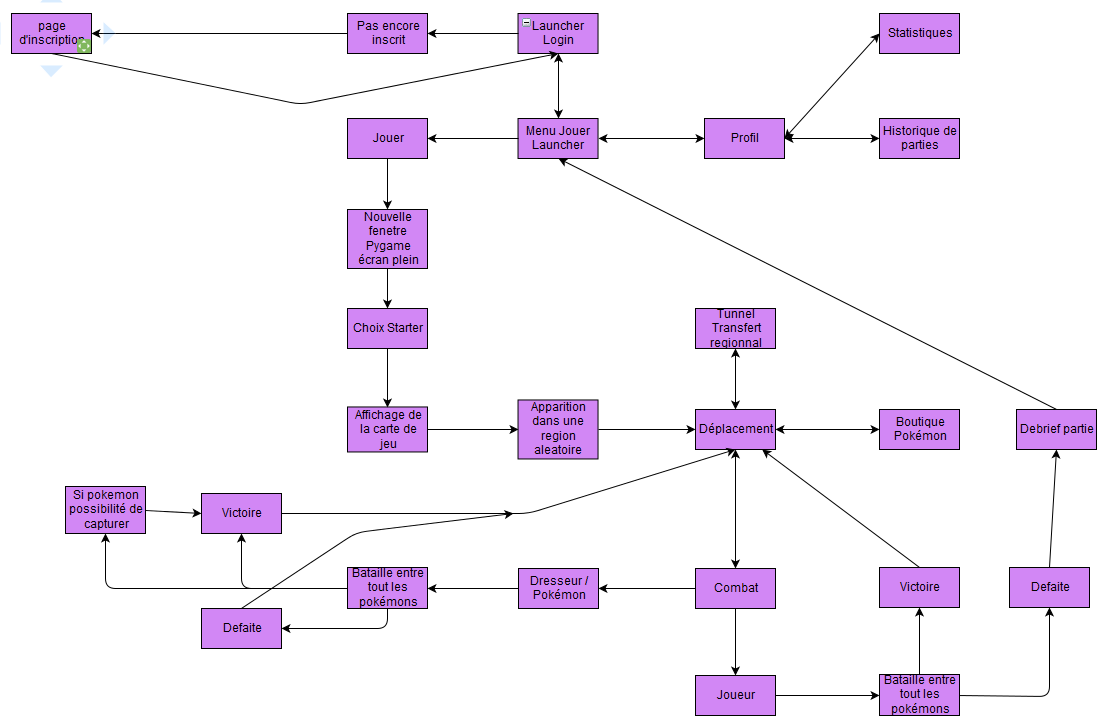
Attributs:

* layout : liste de liste [[x,x,x],[x,x,x],[x,x,x]] représentant la carte du jeu
* zone[1->9] : variable permettant de stocker la zone de jeu

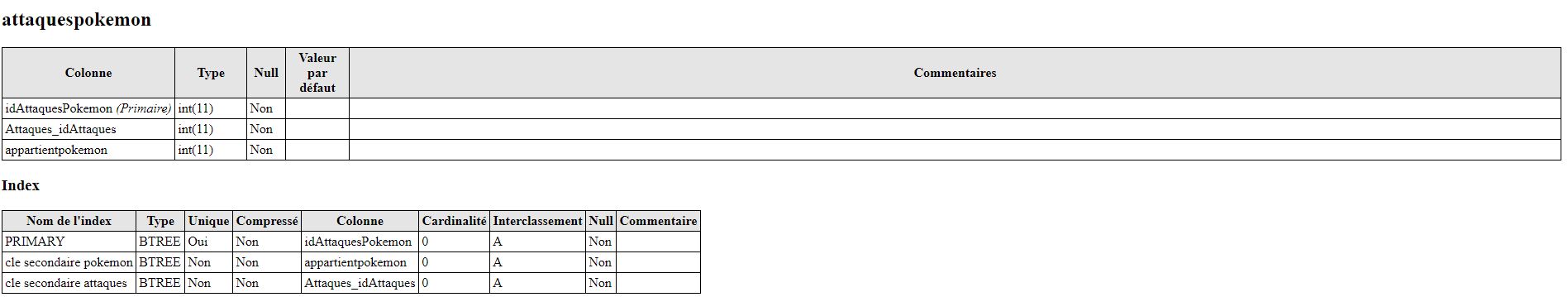
Méthodes :

* /

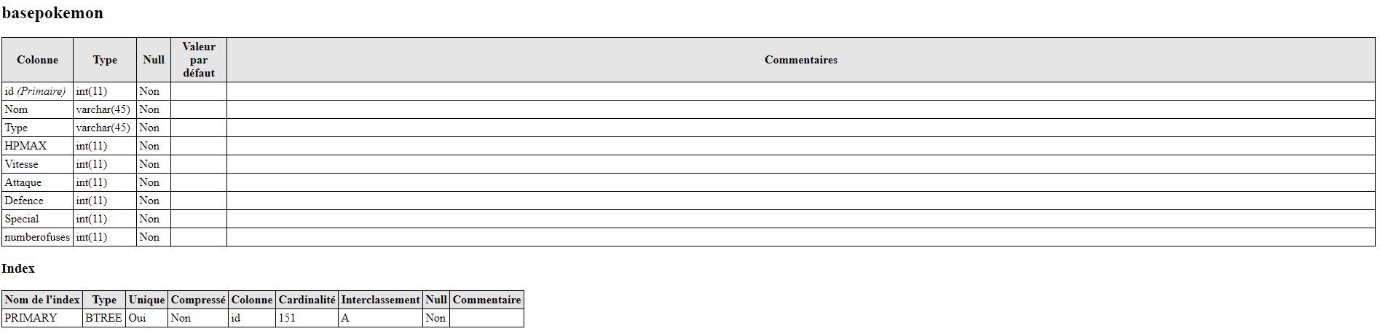
## Schémas de structure globale du Programme



# 13.Tables DB



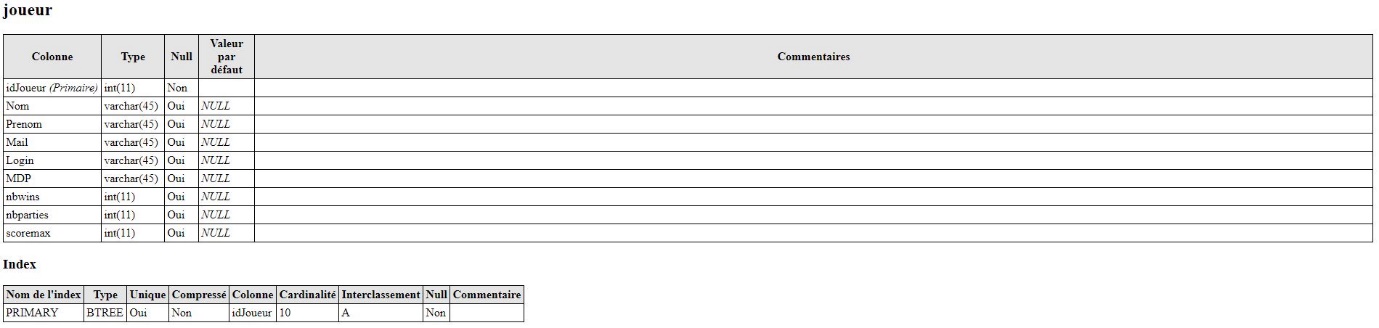
La table attaquespokemon, est utilisée pour stocker les attaques d’un Pokémon spécifique. On y retrouve les champs de l’identifiant, une liaison à la table Attaques et une liaison à la table Pokémon.



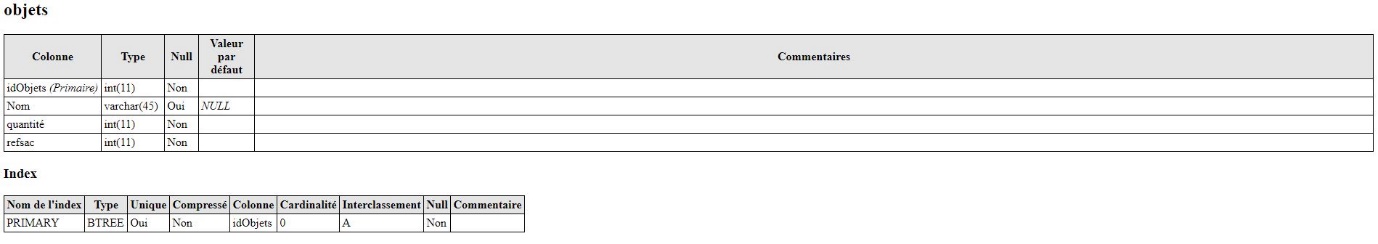
La table basepokemon est une table utilisée pour stocker les « Template » des Pokémons. On y retrouve les attributs principaux des Pokémons comme le nom, le type, les Hp, la vitesse, l’attaque, la défense et le nombre de fois que le Pokémon a été utilisé, ce qui est utile pour nos statistiques.



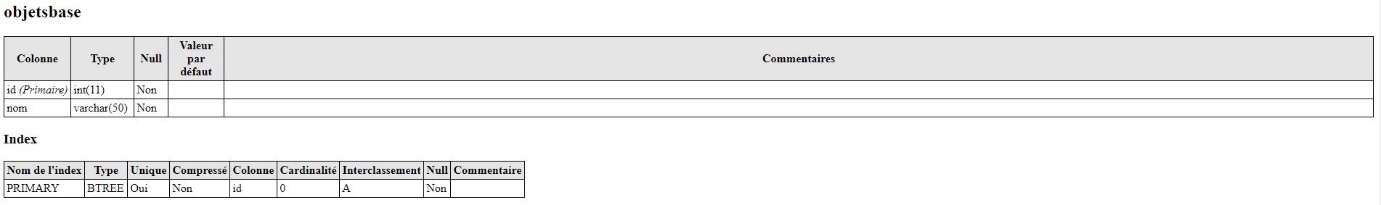
La table équipe est une table utile pour faire le lien entre les différents Pokémons d’un joueur et la table joueur/pnj. Elle ne contient qu’un identifiant permettant de différencier les différentes équipes.



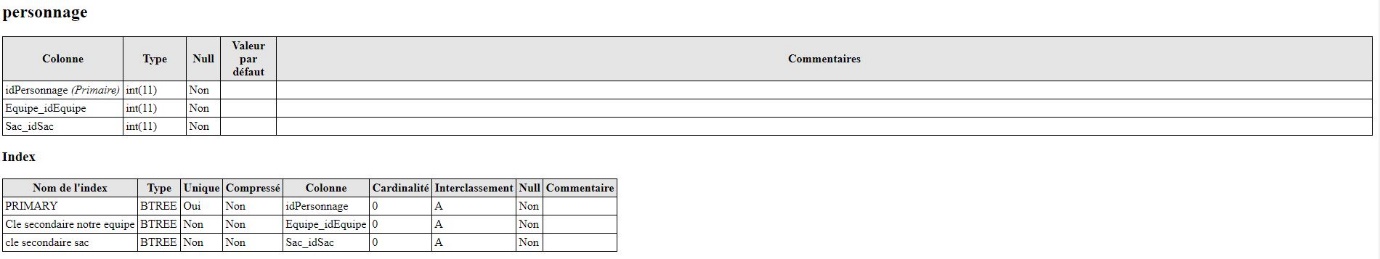
La table joueur est la table utile pour stocker les caractéristiques du joueur telles que le nom, le prénom, l’adresse mail, le login, le mot de passe crypté et quelques statistiques.



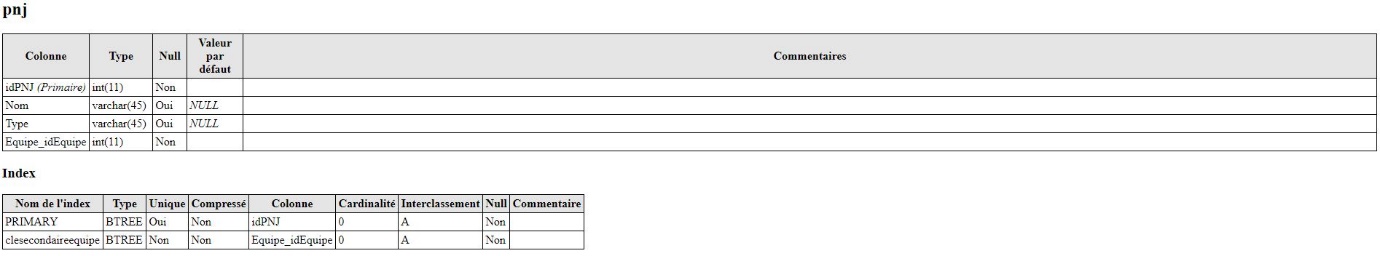
La table objets est une table utilisée pour stocker les différents objets présents dans le sac du joueur. On y retrouve le nom de l’objet, la quantité et une référence au sac.



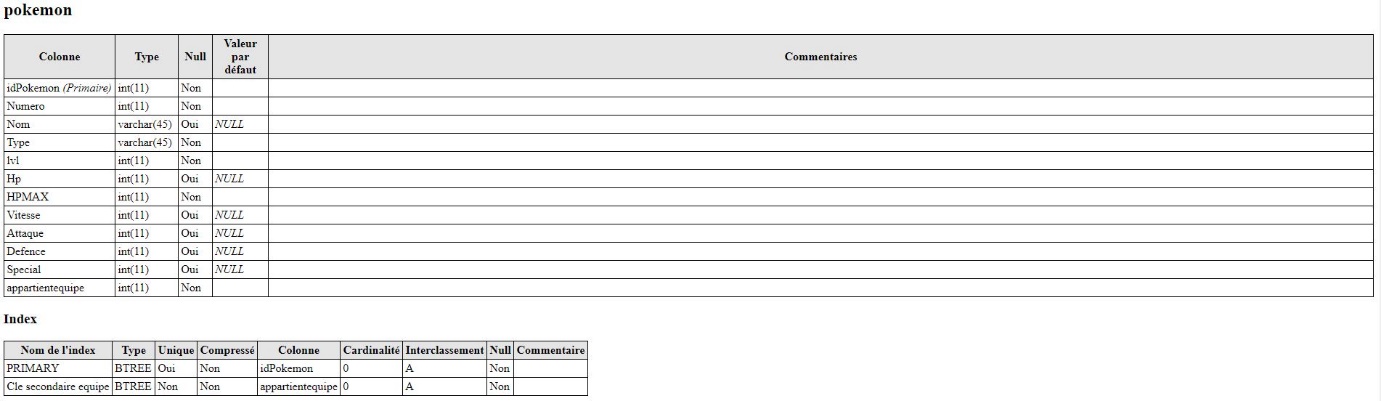
Objetsbase est une table « Template » pour objets. On y retrouve les différentes caractéristiques de base des objets.



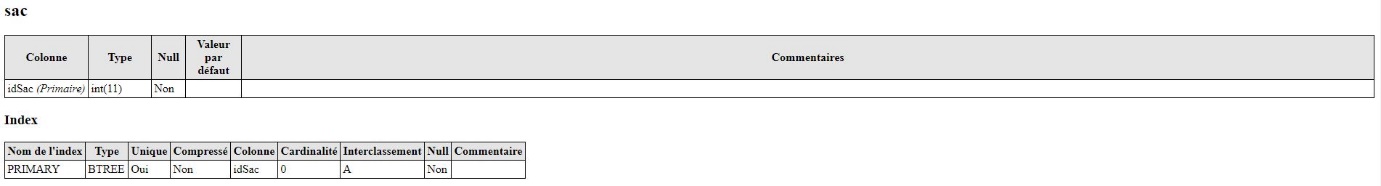
Personnage est une table utile pour représenter le joueur au sein du jeu. Elle est liée au sac et a l’équipe. Elle possède un id.



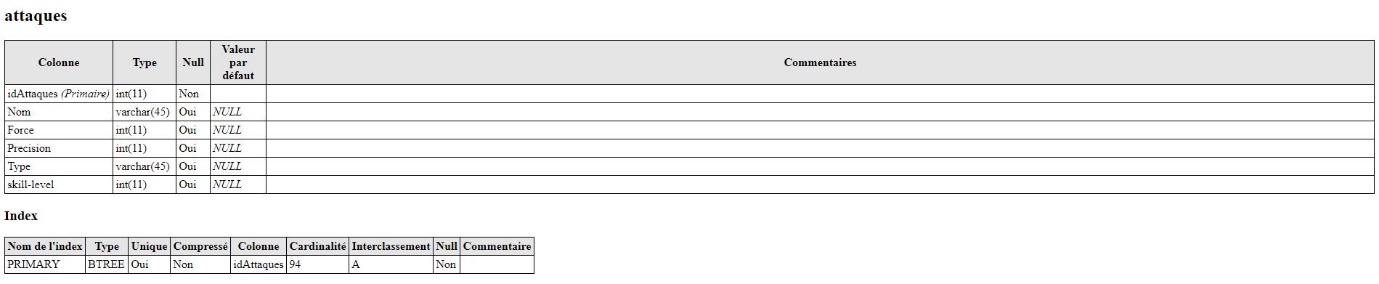
Pnj est une table utile pour représenter les bots dans le jeu. Ils possèdent un nom, un type et possèdent une équipe.



Pokémon est la table utilisée pour stocker les différents Pokémons du joueur. Ils possèdent un id, un numéro pour se référer au Template de ce Pokémon, un nom, un type, des HP, des hpmax, une vitesse, des statistiques d’attaque, de défense, et une référence à l’équipe à laquelle il appartiens.



Sac est une table qui, comme équipe est utilisée pour faire le lien entre le joueur et les objets. Elle ne possède qu’un identifiant.



Attaques est une table utile pour stocker les différentes attaques pouvant être utilisées et apprises par les Pokémons adéquats.

# 14.Chronologie des évènements

## Phase 1 : Analyse

Après avoir décidé de notre sujet de TFE, nous avons commencé à réfléchir sur les aspects concrets de notre jeu. Nous avons commencé par définir les règles de notre jeu, c’est-à-dire les différents éléments que l’utilisateur pourra et ne pourra pas faire dans le jeu. Nous avons ensuite déterminé le contexte du jeu et à qui le jeu s’adresse. Nous avons décidé de faire, dans l’esprit des jeux Pokémons officiels, un jeu qui s’adresse à toute la famille. Ensuite, nous avons commencé à élaborer notre cahier des charges. Nous avons alors mis par écrit, lors du stage de février, toutes nos idées. Nous avons aussi, durant ce stage, reçu de l’aide de la part de nos professeurs concernant leurs parties respectives du projet. Nous avions déjà soulevé quelques problèmes concernant les règles que nous avions établi. Après ce stage, nous avons commencé à créer les vues de notre jeu et de notre site. Vu que nous avions décidé de faire un jeu de type Battle Royale, nous avions aussi dû adapter les règles du jeu de base. Nous avons ensuite créé nos schémas UML et modèles textuels de classe et pour le programme et les schémas MCD, MLD.

## Phase 2 : Codage

Après avoir fini une grosse partie de notre analyse, nous avons commencé à coder.

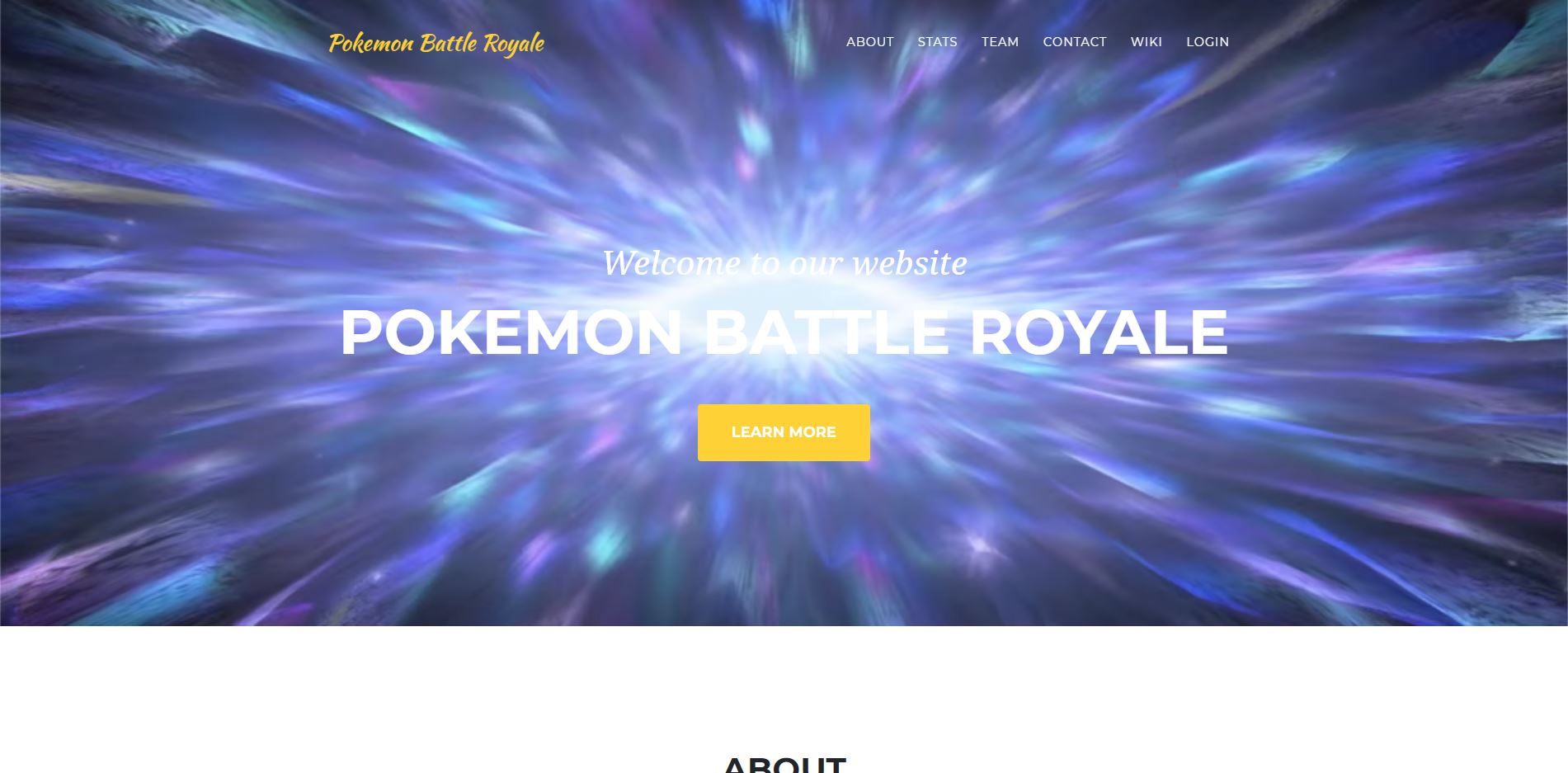
Nous avons commencé par le launcher du jeu et par le visuel du site. Nous avions, pour le launcher, commencé en codant en procédural du coup nous avons dû traduire ce que nous avions fait en Programmation Orientée Objets. Nous avons ensuite fait évoluer le site et le programme en parallèle. C’est lors du mois de mai que nous avons vraiment vu notre projet évoluer. En effet, lorsqu’on a rassemblé les différentes parties codées, on a vraiment vu le jeu tel qu’il était censé être.

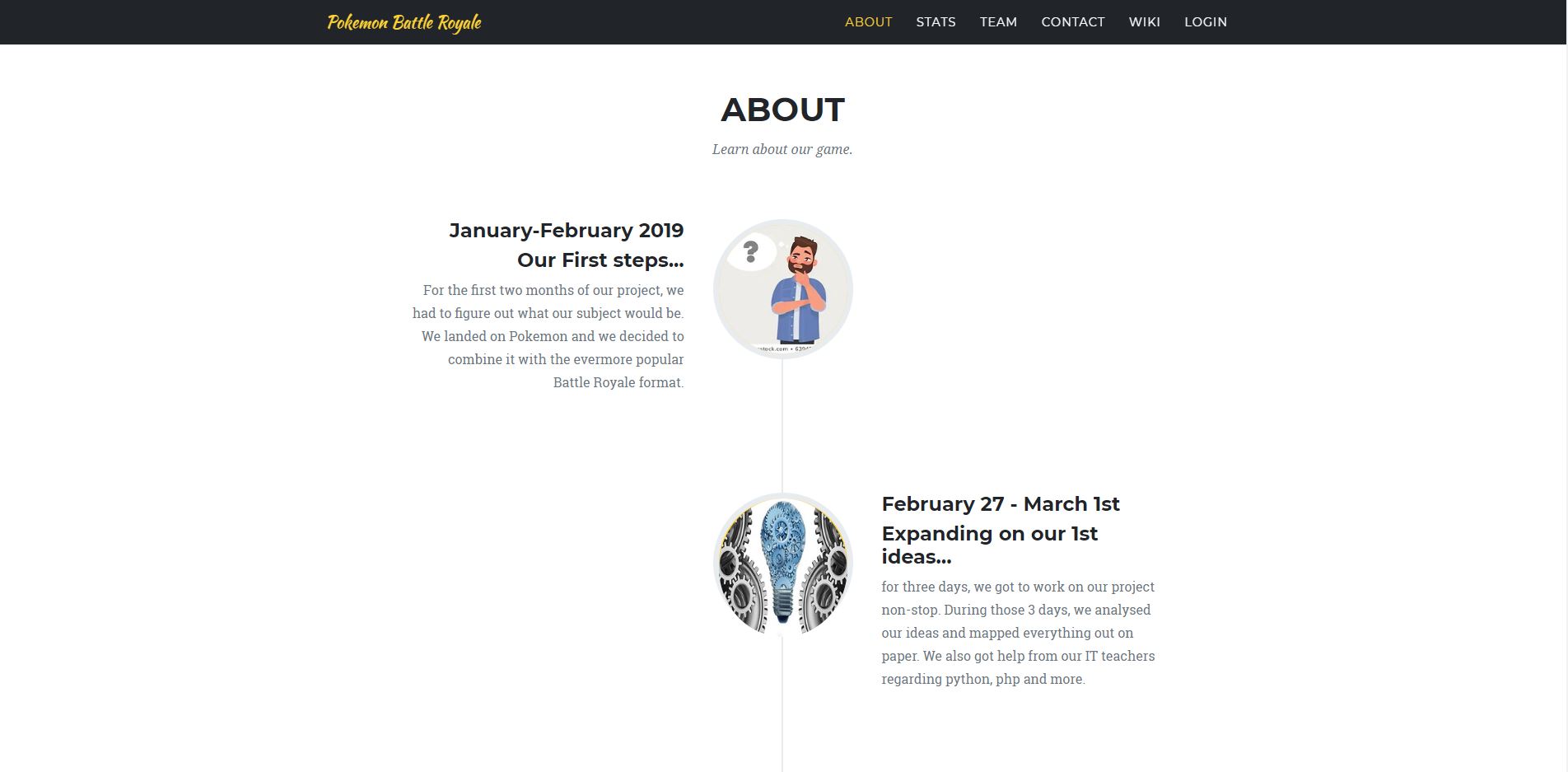
## Phase3 : Tests

La phase de tests a évolué en parallèle avec la phase de code. Vu que nous avons codé par module, nous n’avons donc pas vraiment une phase de tests à proprement parler. Nous avons cependant rencontré plusieurs problèmes mais rien qui nous a bloqués pendant trop de temps.

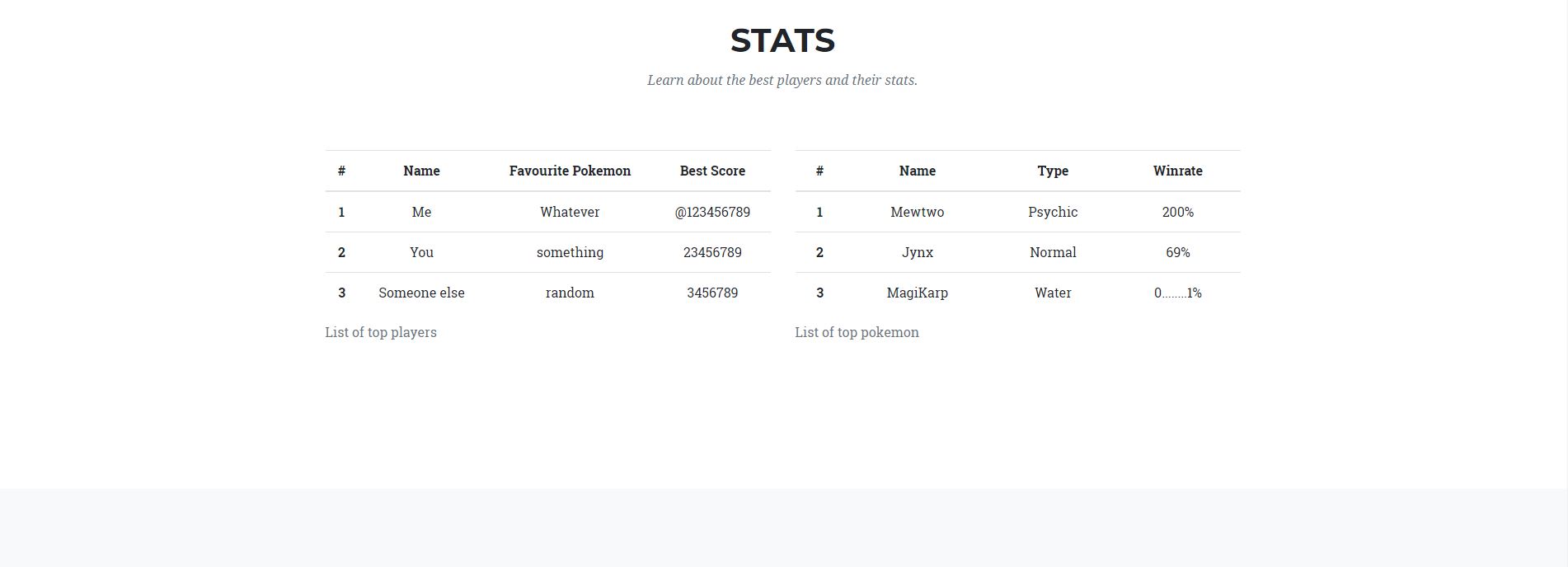
# 15.Vues Finales

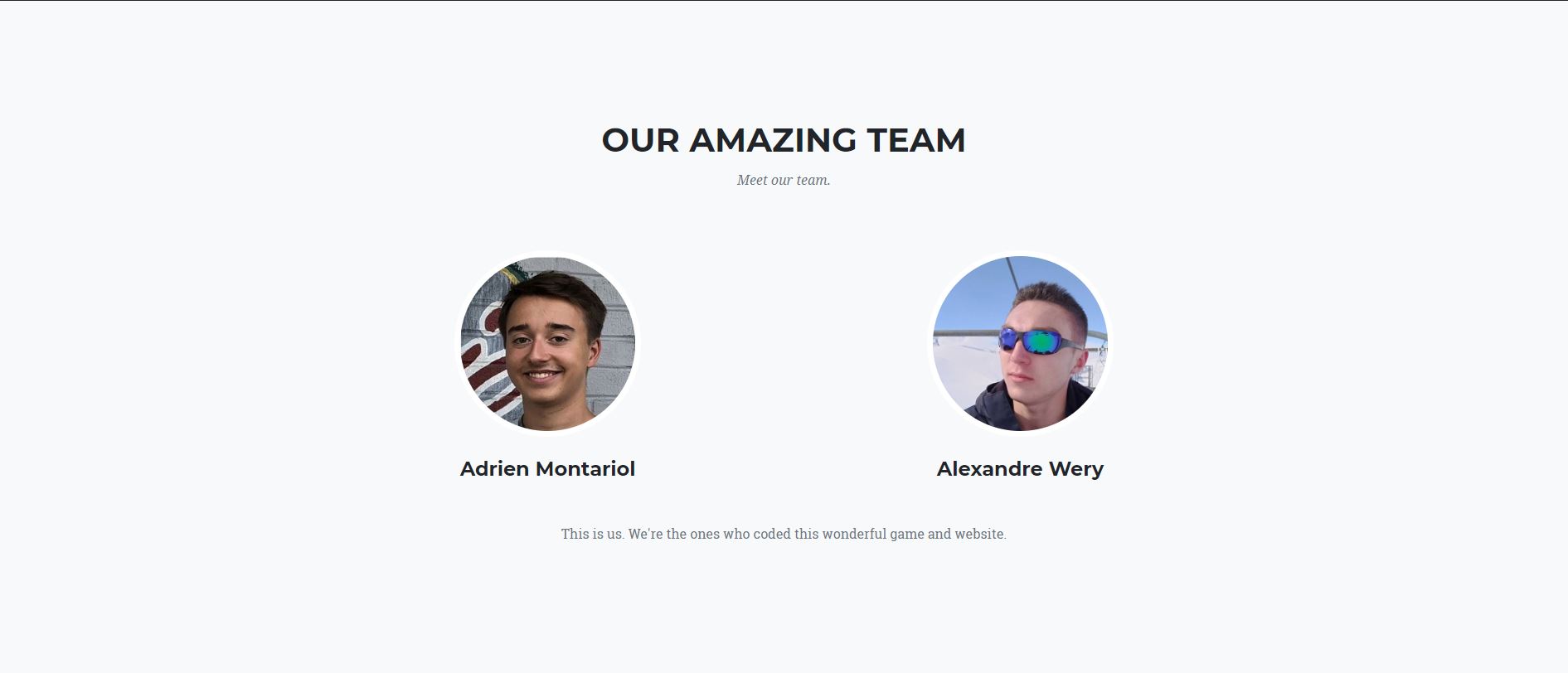
## Site





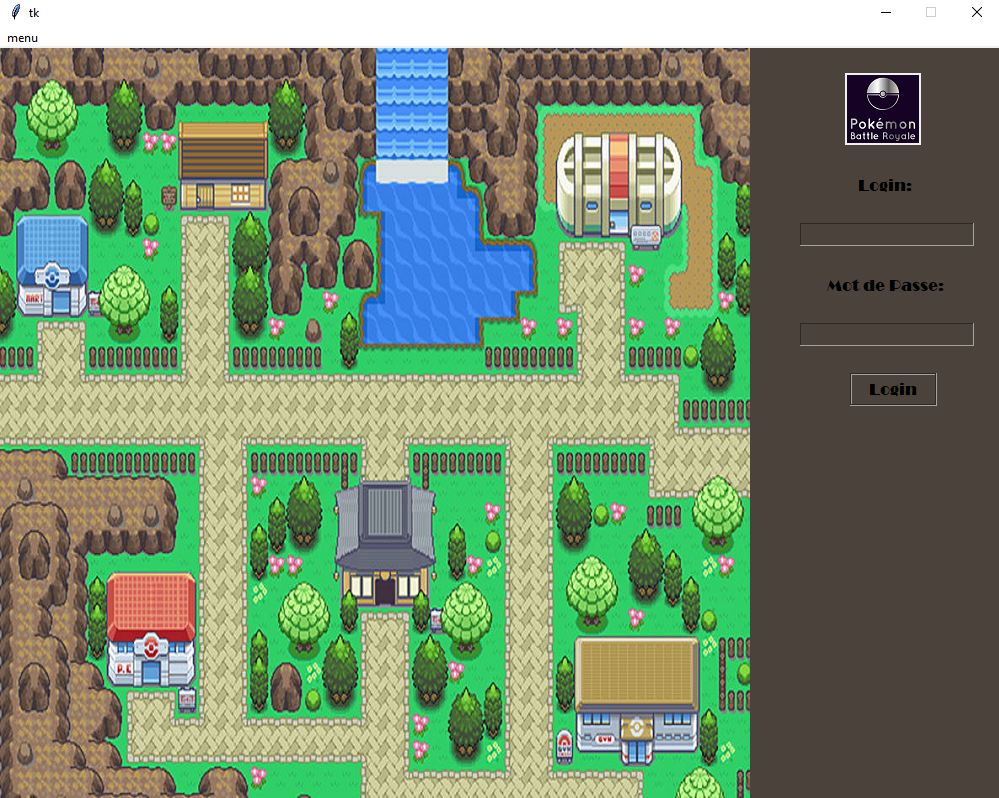


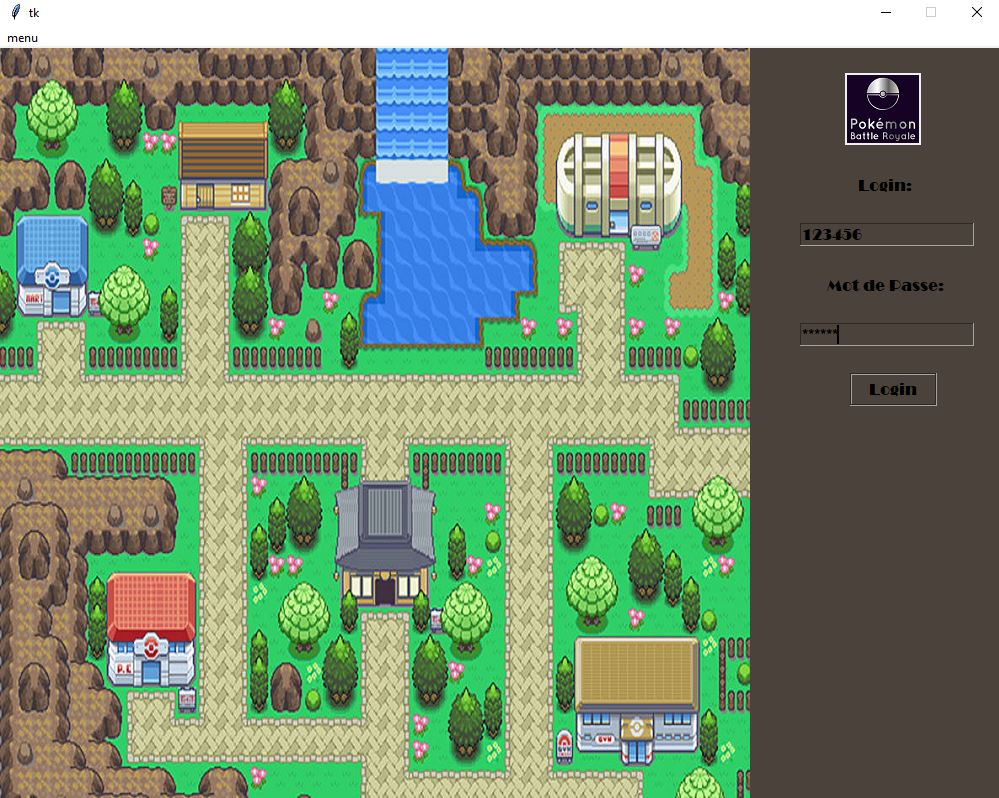


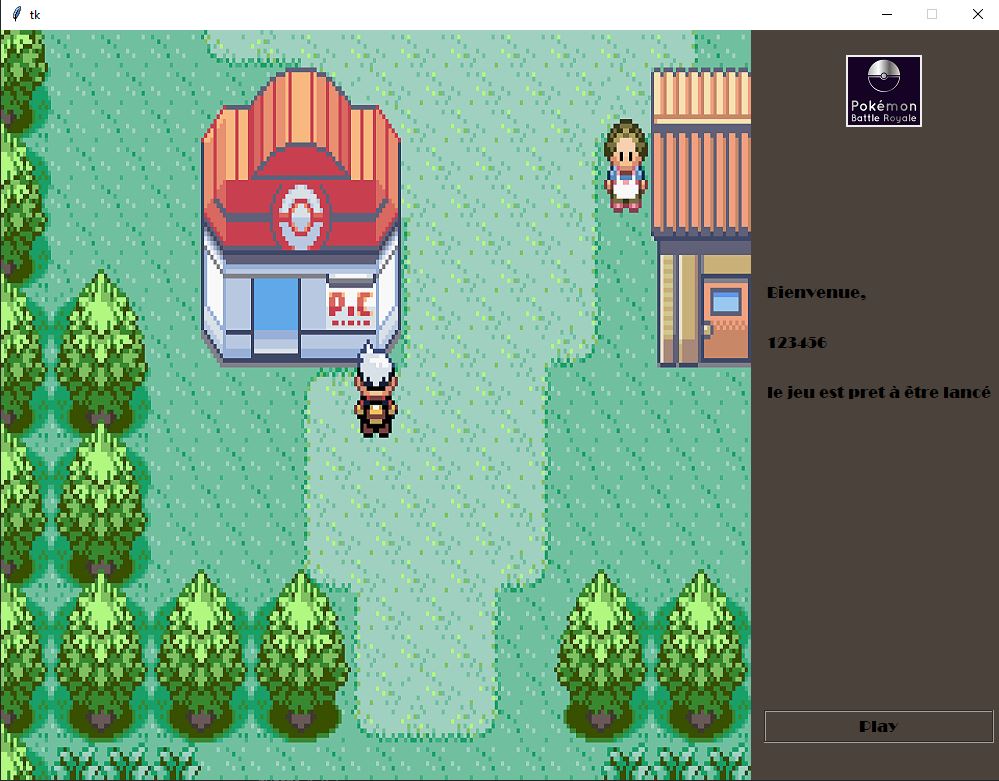


## Jeu

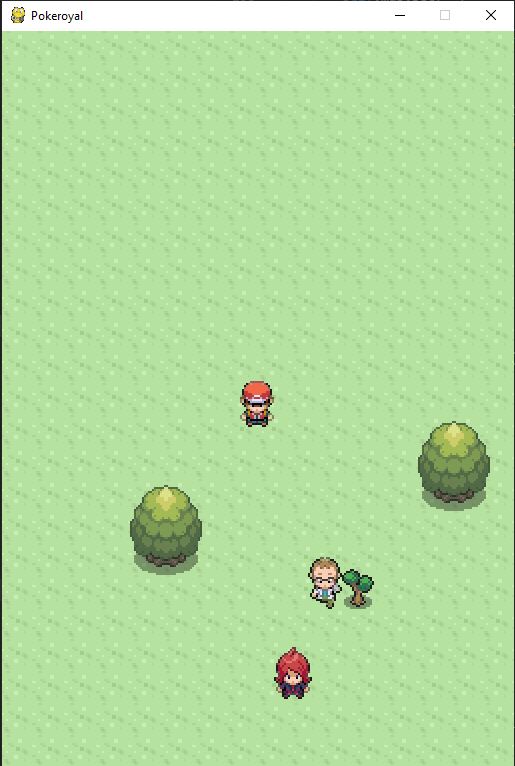
### Launcher

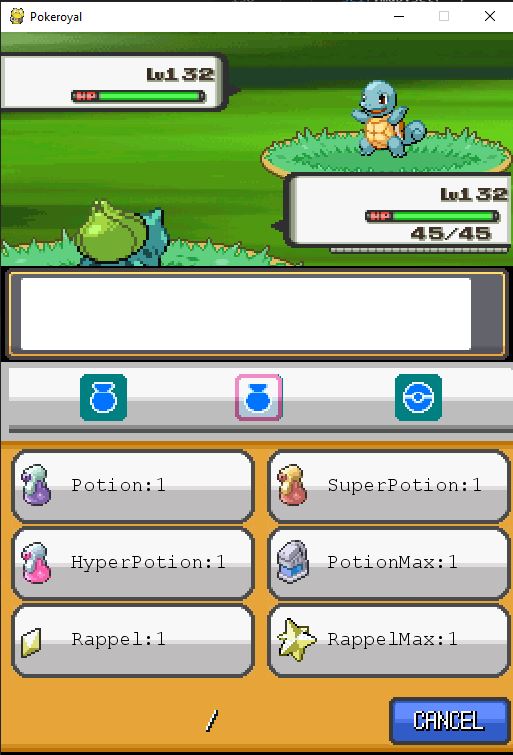
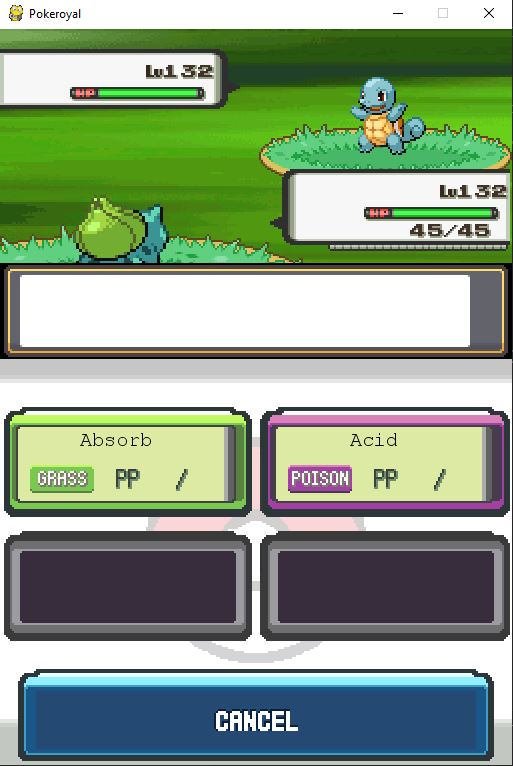






### Jeu

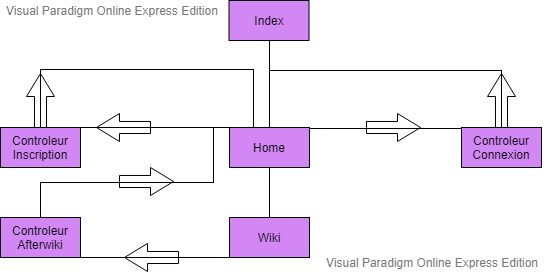




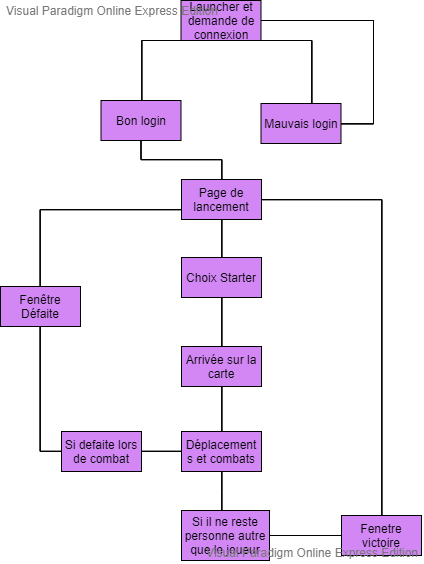


# 16.Arborescences Finales

## Site Web

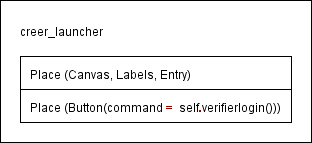


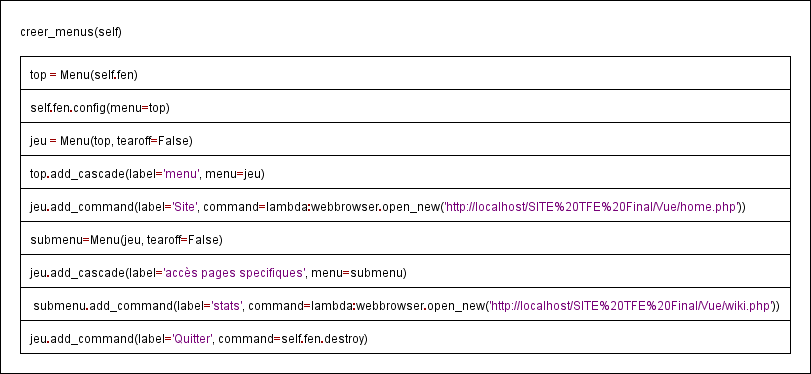
## Programme Python

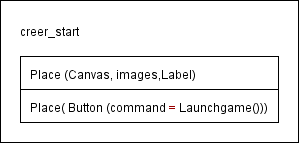


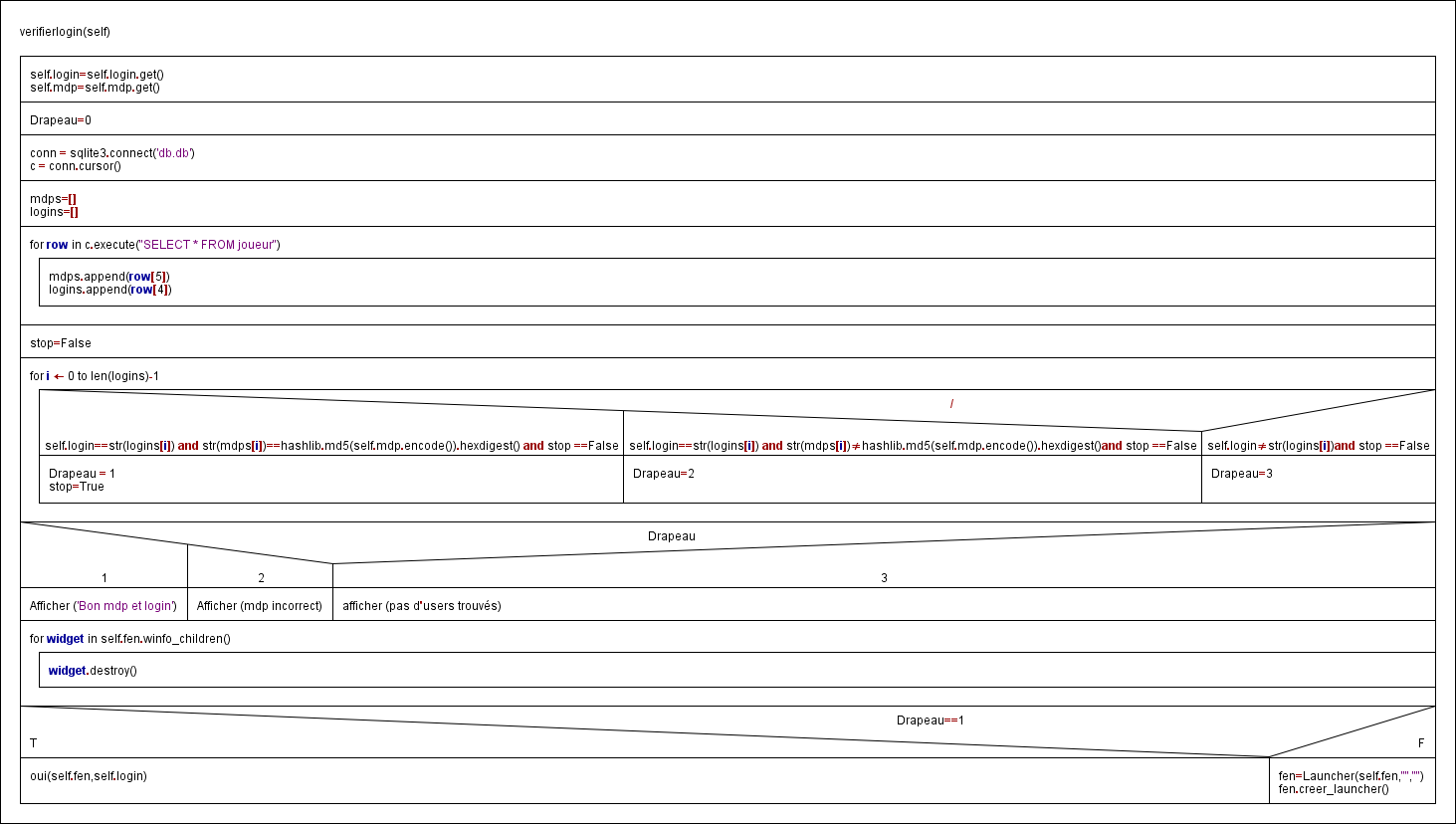
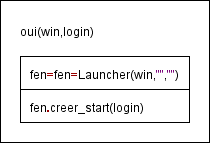
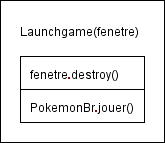
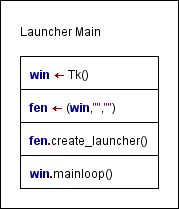
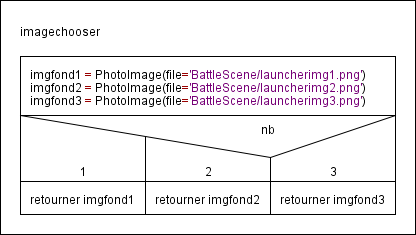
# 17.GNS

## Programme

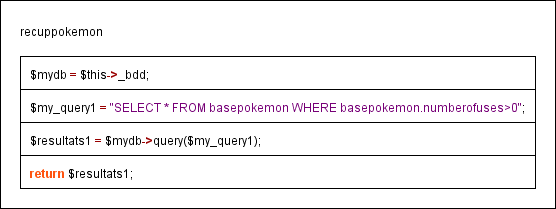
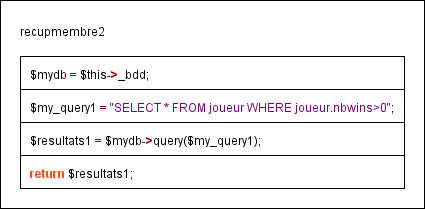
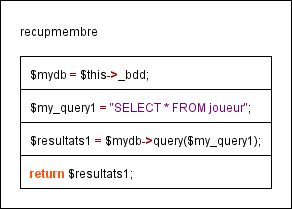
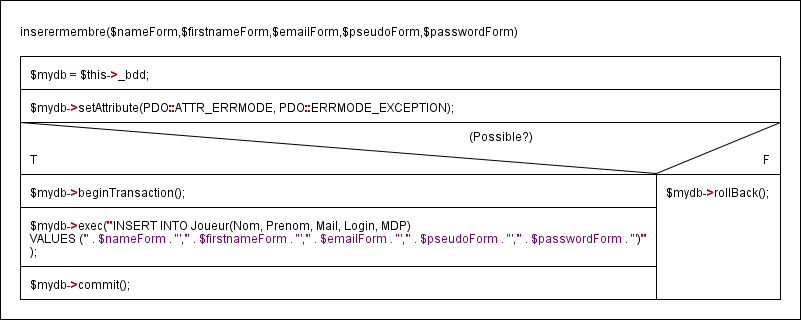








## Site



# 18.Schémas textuels finaux

Classe

Nom : Play

role:

Attributs:

Starters : attribut pour stocker les différents pokémons de départ

pokeball : attribut pour stocker une image

conn : attribut pour stocker la connexion à la db

self.x = position x du joueur sur la map en pixels

self.y =position y du joueur sur la map en pixels

self.vel = vitesse de déplacement

self.tileset = image contenant toutes les parties de la map

self.level=contient la disposition de la map

self.xs=largeur de la map en tuiles

self.ys=hauteur de la map en tuiles

self.xtj=position x du joueur sur la map en tuiles

self.ytj=position y du joueur sur la map en tuiles

self.lt=[0,0]

self.collision=contient un liste qui détermine si une tuile est traversable ou non

self.trainers = charge l'image contenant les personnages du jeu

self.message=Vrai si il y a un message à afficher

self.eact=dictionnaire contenant les interaction avec la tuile choisie

self.way=initialisation de la direction du personnage du joueur à nulle

methodes:

move:methode utilisée pour gérer les déplacements du joueur

hocollision : methode utilisée verifier les collisions

pickpokeball : methode pour gerer l'interactiona vec les pokeballs au sol

-------------------------------------------------------------------------------------------------------------------

Classe:

nom:createpnj

role : instancier des personnages non joueur

Attributs

self.number=num#numéro de l'équipe du PNJ dans la DB

self.teamnumber=numero de l'equipe du pnj

self.fen=fenetre

self.pos=coordonnées du PNJ

self.trainer=détermine si je PNJ est un dresseur

self.range=détermine la taille du tour que le PNJ décrit

self.direction=détermine la direction dans laquelle le pnj est tourné

self.ranstate=0

self.vit=vitesse de déplacement

self.tclass=type du PNJ

self.xtp=coordonnée x du PNJ

self.ytp=coordonnée y du PNJ

self.active=si le PNJ peut ineragir

methodes:

move : methode utilisée pour gérer le mouvement des pnj

collisionp : methode pour gérer les collisions du pnj

duel : methode utilisée pour gérer le fait de dueler

createteam : methode utilisée pour créer l'equipe du pnj

---------------------------------------------------------------------------------------------------------------------

Classe :

nom : initfight

role : initier les combats

attributs:

self.Pteam=équipe du joueur

self.Bteam=pokemon sauvage/équipe du pnj

self.wmenu=indique le menu choisi par le joueur

self.select=variable tampon contenant le dernier menu survolé par l'utilisateur

self.sel=indique la position du sac, s’il est en mode soins ou balls

self.bsel=indique la position temporaire du sac, si il est en mode soins ou balls

self.cc=Vrai si l'utilisateur peut cliquer

self.tsoin=Vrai si l'utilisateur veux utiliser un objet de soin

methodes:

fight : méthode utilisée pour gérer les combats(le menu)

bmenu : méthode utilisée pour gerer le fait de cliquer sur des elements du menu

bfight: méthode utilisée pour gérer le fait d'attaquer dans le menu bataille

bgattacks : méthode pour gérer l'affichege des attaques

bpokemon : méthode pour gérér l'interaction avec l'equipe

bbag : méthode pour gérer l'interaction avec l'equipe

# 19.Mode d’Emploi :

Lire ReadMe.txt dans le dossier Jeu

# 20.Cas de Tests

|  |  |  |
| --- | --- | --- |
| N° | Prévision | Résultat |
| 1.1 | Déplacement du personnage dans une image | Tout a bien fonctionné |
| 1.2 | Collision | Problèmes avec les collisions selon les directions mais facilement résolu |
| 1.3 | Collision | Tout marche parfaitement |
| 1.4 | Interaction avec un élément | Pas de problèmes, fonctionne comme souhaité |
| 1.5 | Déclenchement de combat | Facilement résolvable |
| 1.6 | Interaction avec clic gauche | Fonctionne facilement |
| 1.7 | Interaction avec db | Fonctionne facilement |
| 1.8 | Requetes SQL | Quelques difficultés mais pas de problèmes sur le long terme |
| 1.9 | Mise à jour des bases de données entre elles | Problèmes non résolus dans un sens. |
| 2.1 | Interaction avec la db via php | Problèmes avec la synthaxe mais rien d’insurmontable |
| 2.2 | Blindage de l’encodage dans les différents formulaires | Pas de problèmes |
| 2.3 | Tests de l’interaction avec la connexion à la page web | Pas de problèmes |
| 2.4 | Tests de blindage JS | Non résolus |
| 2.5 | Création du wiki et de son interaction avec l’aspect connectivité du site web | Pas de problèmes |
|  |  |  |

# 21.Remerciements

Nous tenons à remercier nos professeurs et nos camarades de classe pour ces deux super années passées en leurs compagnie.

# 22.Conclusion

En conclusion, ce projet nous a apporté de la perspective sur ce que c’est de faire un gros travail. En effet, il y a beaucoup de choses que nous aurions pu faire mieux mais nous sommes tout de même fiers de notre travail et sommes impressionnés par le résultat.

Nous avions pour but de faire un jeu adapté à tout public qui est facile à prendre en main. Nous avons réussi à réaliser certains de ces éléments au détriment des autres mais c’est comme ça qu’on apprend à mettre les choses en perspective.

Si on devait refaire ce projet, on changera la manière dont on s’y est pris avec le temps. En effet, on s’y est pris trop tard. On voyait la deadline a des mois de nous et on se disait qu’on avait encore le temps. Ce n’est que lorsqu’on l’a vue se rapprocher qu’on a sérieusement commencé à se bouger.

Même si ce projet nous a apporté beaucoup de stress, nous sommes tout de même content de nous et nous espérons que vous le serez aussi.