Document de Conception PPII 2
Solveur du Flopple

Norman ASSING, Anna BIAUSQUE
Clément COUCHEVELLOU, Caroline WANG

13 avril 2022

Table des matières

Chapitre 1

Cahier des charges

Contents

1.1 Contexte

L'objectif est la conception d'un solveur de WORDLE, WORDLE que nous devons nous-même réaliser de A à Z, avec de multiples contraintes techniques.

1.2 Cadre technique : Jeu de WORDLE

La première étape est la conception du jeu de WORDLE, ici prénommé FLOPPLE, basée sur une architecture Web/Python/Base de données. L'application doit être finalisée pour le 31 avril, pour cela nous avons donc accès à Python et Flask, pour la partie intéractive, SQL pour le stockage et la gestion des données ainsi que HTML et CSS pour la partie statique. Le jeu devra être paramètrable, en effet le joueur doit pouvoir choisir le nombre d'essais qu'il aura à sa disposition pour trouver le mot, mais il doit également pouvoir choisir la longueur du mot en question, qui sera compris entre 2 et 25, car limité par le dictionnaire français. Afin qu'un joueur ne tombe pas plusieurs fois sur le même mot nous devons également sauvegarder dans une base de données les mots déjà trouvés par un joueur.

Chapitre 2

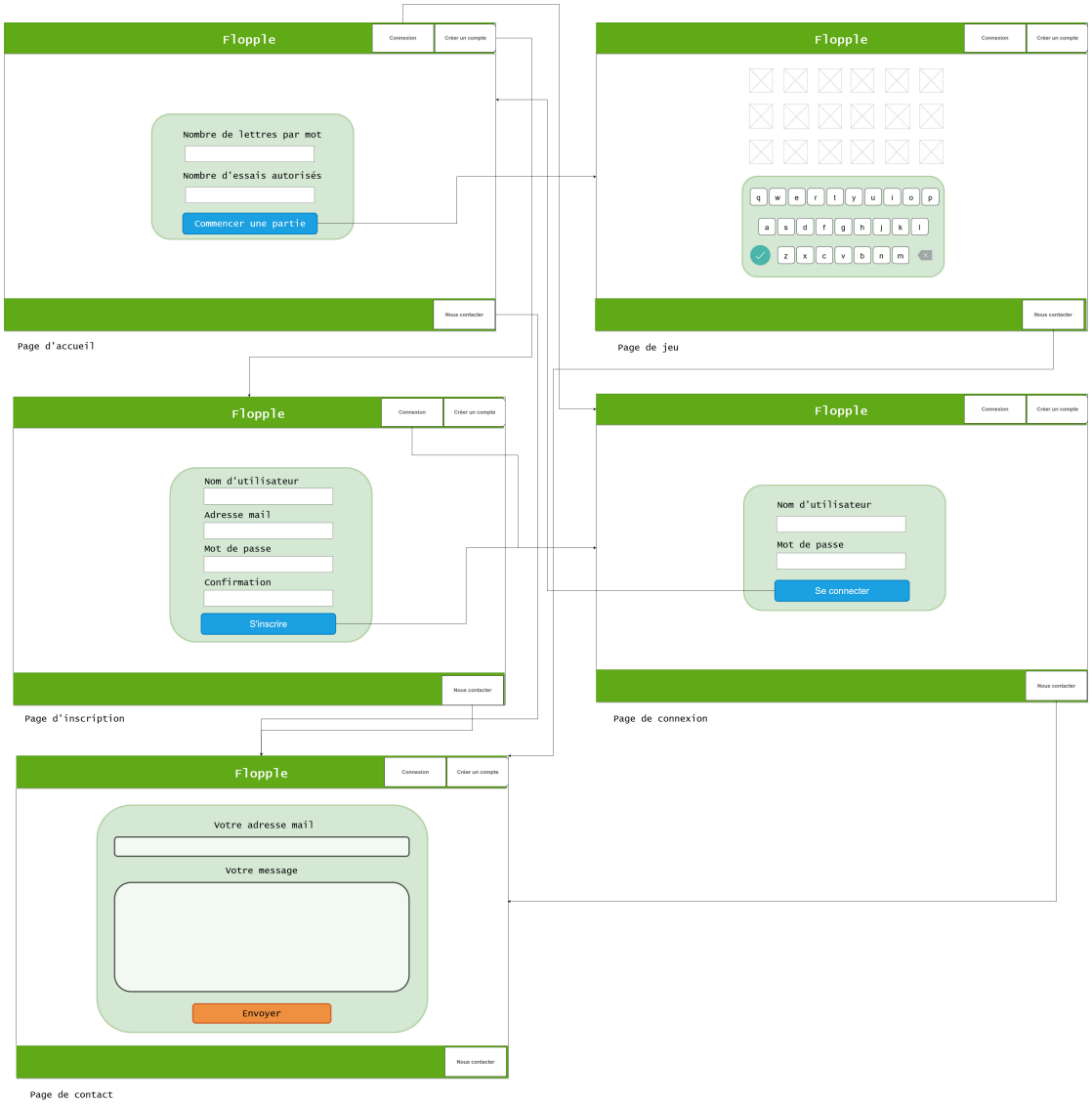
Conception de notre Application web

Contents

2.1 Utilisateur

Chaque utilisateur peut créer un compte sur l'application, ce qui lui permet notament de sauvergarder les mots qu'il a déjà réussi à trouver afin de ne pas retomber dessus en jouant. Les informations relatives aux utilisateurs sont stockées dans notre base de données qui sera détaillée plus loin.

2.2 Mockups



2.3 WORDLE en Python

Le fichier Python séléctionne de manière aléatoire un mot de la bonne longueur parmi les mots stockés dans la base de données. On fait enuite intéragir grâce à Flask la page HTML et le fichier Python afin de récupérer puis d'analyser les mots entrés par l'utilisateur, en les comparants avec le mot cible. A chaque nouveau mot entré, le fichier Python renvoie les similarités sous forme de couleurs associées à chaque lettre, ainsi que les mots précédemment entrés et leurs similarités avec le mot cible. Est enfin renvoyé le nombre d'essais restant.

2.4 Intégration web

La base du site est réalisé via HTML/CSS, couplé à Flask via un fichier Python pour rendre le site intéractif. Afin de rendre le site plus agréable, dynamique et ergonomique possible, nous utiliseront également JavaScript pour l'affichage dynamique et la gestion des cookies.

2.5 Partie Base de données

Pour notre base de données, nous utilisons SQL, plus précisemment SQLite3 sur Python. Le choix de SQLite3 pour gérer nos données sous Python a été motivé majoritairement pour la performance du SQL pour gérer de grandes quantités de données, mais également par le fait que SQLite3 permette de gérer très intuitivement des bases de données directement dans notre fichier Python.

Notre base de données est constituée de plusieurs Tables, la table userInfo qui regroupe toutes les informations concernant les utilisateurs.

Les informations d'un compte utilisateur sont conservées dans la table userInfo. Un compte utilisateur est identifié par un identifiant *id* (de type *integer*) et est obligatoirement associé à un unique nom d'utilisateur *username* (de type *text*), une unique adresse mail *email* (de type *text*), un mot de passe *password* (de type *text*), la série de victoires en cours *currentStreak* (de type *int*) et la série maximum de victoires du joueur *maxStreak* (de type *int*).

Ainsi que la table words qui contient toutes les informations relatives aux mots que le jeu reconnait.

Chaque mot de la langue est conservé dans la table words. Un élément de cette table est identifié par un identifiant *id word* (de type *integer*) et représente obligatoirement et sans redondance un mot *word* (de type *text*) d'une longueur *length* (de type *int*) donnée.

La gestion de l'historique des parties du joueur requiert également une table games.

Les données de chaque partie sont conservées dans la table games. Une partie est identifiée par un identifiant *id game* (de type *integer*) et les informations conservées sont : le nombre d'essais maximum *tries* (de type *int*) de la partie, les tentatives *guesses* (de type *text*) du joueur (chaque nouvelle tentative est concaténée aux mots précédents, eux-mêmes concaténés dans l'ordre chronologique d'entrée), les résultats *results* (de type *text*) correspondant à chaque tentative (cette chaine de caractère est obtenue de la même manière que *guesses*), le statut *status* (de type *int*) de la partie (perdue, gagnée ou en cours), le nombre d'essais *leftT* (de type *int*) restant à la fin de la partie. Une partie terminée est associée à un mot par son identifiant *#id_ word* (de type *integer*) et à un compte utilisateur par son identifiant *#id* (de type *integer*).

Les tables utilisées sont donc les suivantes :

userInfo(id, username, email, password, currentStreak, maxStreak)
words(id word, word, length)
games(id game, tries, guesses, results, status, leftT, #id_word, #id)

Les contraintes attachées au schéma de la base de données sont énumérées ci-dessous :

— Les clés primaires sont soulignées.
— Les clés étrangères sont identifiées par #.

— Le nom d'utilisateur *username* et l'adresse mail *email* d'un joueur ne peuvent avoir la valeur *null* et doivent être uniques dans la table.
— Le mot de passe *password* ne peut être *null*.
— La série de victoires courante et la série de victoires maximale d'un joueur sont des entiers naturels.

— Le mot *word* n'est pas *null*
— La longueur d'un mot *length* est un entier naturel non nul.

— Le nombre maximum d'essais possibles *tries* est un entier naturel non nul.

— Les attributs *guesses* et *results* ont pour valeur par défaut " lorsqu'un joueur lance une nouvelle partie.
— Le statut *status* d'une partie est restreint aux valeurs 0 si la partie a été perdue, 1 si elle a été gagnée, 2 si elle est encore en cours.
— Le nombre d'essais restants d'un joueur pour une partie *leftT* est un entier compris entre 0 et *tries*.
— L'*#id_word* de la table **games** fait référence à l'identifiant *id word* de la table **words**.
— L'*#id* de la table **games** fait référence à l'identifiant *id* de la table **userInfo**.

Ces tables sont illustrées par la figure 2.1.

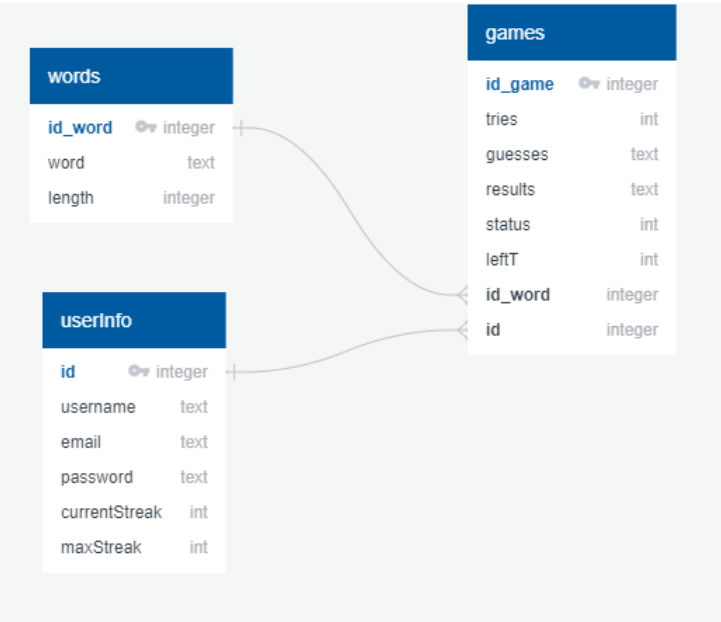


FIGURE 2.1 – Schéma de la base de données

La mise sous troisième forme normale (3NF) des tables décrites ci-dessus donne les tables suivantes :

userInfo(id, #username, currentStreak, maxStreak)
userInfo_username(username, #email)
userInfo_email(email, password)

words(id word, #word)
words_word(word, length)

games(id game, #tries, #guesses, #id_word, #id)
games_triesANDguesses(tries, #guesses, status, leftT)
games_guesses(guesses, results)

Le schéma de la base de données sous troisième forme normale est alors celui de la figure 2.2 :

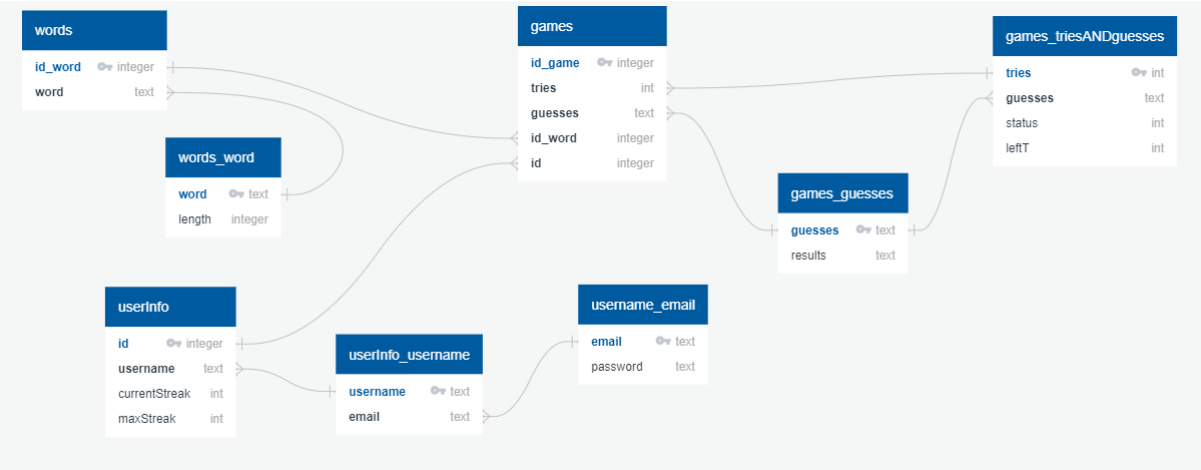


FIGURE 2.2 – Schéma de la base de données mise sous forme 3NF

2.6 Particularité de l'application

Plutôt que de créer un simple clone en français de Wordle, nous avons décidé d'ajouter des fonctionnalités ludiques dans le but de donner un réel intérêt à notre application. Ainsi, Flopple serait une version de Wordle où l'on collecterait des images autour du thème du mème de Big Floppa.

FIGURE 2.3 – L'image la plus célèbre de Big Floppa

Big Floppa est le nom donné à plusieurs caracals, un genre de la famille des félins. Le plus célèbre est Gosha, dont le compte Instagram dépasse les 150 000 abonnés.

2.7 Détail des fonctionnalités supplémentaires

Lorsqu'un joueur trouve un mot, il obtient un Floppa Coin, qui lui permet de faire un tirage dans un distributeur afin d'obtenir des images de Big Floppa. Un système de rareté fera que certaines images seront plus difficiles à obtenir que d'autres.

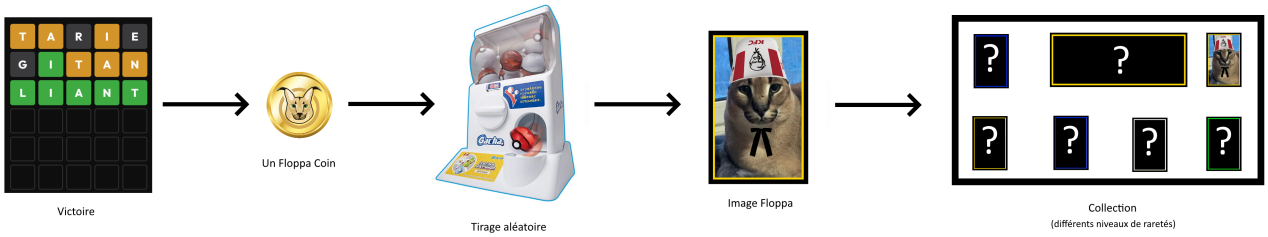


FIGURE 2.4 – Schéma expliquant le système de collection d'images

Cependant, les parties ne seront pas sans risque. S'il perd trop souvent, le joueur verra son compte se faire supprimer, et il n'aura plus d'autre choix que de tout recommencer depuis le début. Cette mécanique de jeu est introduite via le plus grand ennemi de Floppa, un chat de la race des Sphynx nommé Bingus.

FIGURE 2.5 – Bingus

Chaque partie perdue donne au joueur un fragment de Bingus, et au bout de trois fragments, le compte du joueur est définitivement supprimé. En revanche, si lors d'un tirage au distributeur, le joueur reçoit une image étant déjà dans sa collection, tous ses fragments seront retirés.

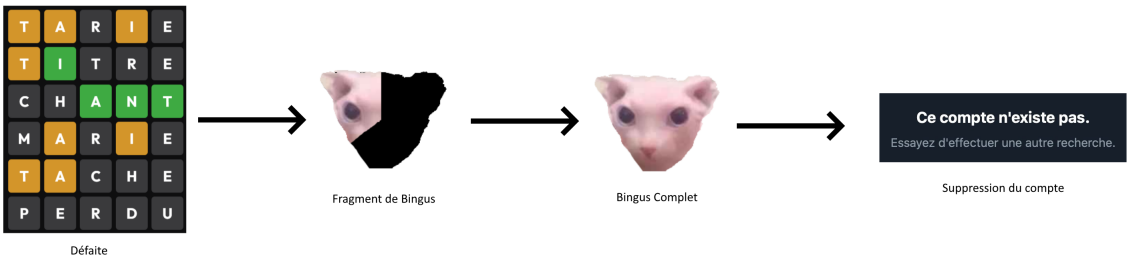


FIGURE 2.6 – Schéma expliquant le processus de suppression de compte