Spécification Technique

### 1. Technologies Utilisées

#### **Backend :**

* **Python 3** :
  + Langage de programmation utilisé pour gérer la logique serveur et les traitements de données de l'application.
* **Flask** :
  + Framework web léger qui permet de gérer les routes et la logique métier. Flask est un choix adapté pour des applications web simples à moyenne échelle, en offrant flexibilité et simplicité de développement.
* **SQLAlchemy** :
  + ORM (Object Relational Mapping) utilisé pour interagir avec la base de données de manière abstraite, en manipulant les données sous forme d'objets Python. Cela permet de sécuriser les transactions et simplifie les requêtes complexes.

#### **Frontend :**

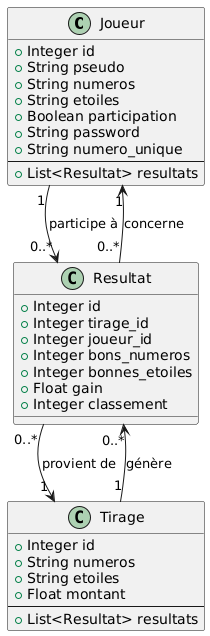
* **HTML5 et CSS3** :
  + Technologies utilisées pour structurer et styliser les pages web. HTML5 fournit la structure des éléments de la page, tandis que CSS3 gère l'apparence visuelle, comme les couleurs et la mise en forme.
* **JavaScript** :
  + Langage de programmation côté client qui permet d’ajouter des interactions dynamiques et des validations côté utilisateur pour améliorer l'expérience de navigation.

#### **Base de Données :**

* **SQLite** :
  + Base de données légère, facile à configurer et intégrée à l’application. Elle est idéale pour des applications de petite à moyenne taille où une base de données serveur complète n’est pas nécessaire.

### 2. Modèles de Données

Diagramme de Classes :



Tables SQL :

-- Table Joueur

CREATE TABLE Joueur (

id INTEGER PRIMARY KEY AUTOINCREMENT,

pseudo VARCHAR(50) NOT NULL UNIQUE,

numeros VARCHAR(50),

etoiles VARCHAR(50),

participation BOOLEAN DEFAULT TRUE,

password VARCHAR(100) NOT NULL,

numero\_unique VARCHAR(50) NOT NULL UNIQUE

);

-- Table Tirage

CREATE TABLE Tirage (

id INTEGER PRIMARY KEY AUTOINCREMENT,

numeros VARCHAR(50),

etoiles VARCHAR(50),

montant FLOAT NOT NULL

);

-- Table Resultat

CREATE TABLE Resultat (

id INTEGER PRIMARY KEY AUTOINCREMENT,

tirage\_id INTEGER NOT NULL,

joueur\_id INTEGER NOT NULL,

bons\_numeros INTEGER,

bonnes\_etoiles INTEGER,

gain FLOAT,

classement INTEGER,

FOREIGN KEY (tirage\_id) REFERENCES Tirage(id) ON DELETE CASCADE,

FOREIGN KEY (joueur\_id) REFERENCES Joueur(id) ON DELETE CASCADE

);

-- Activer les contraintes de clé étrangère (pour SQLite)

PRAGMA foreign\_keys = ON;

### 3. Routes Flask

- / (GET, POST) : Affiche la page d'accueil avec la liste des joueurs et permet l'inscription des joueurs avec validation du mot de passe. Les joueurs peuvent choisir leurs numéros et étoiles manuellement ou automatiquement.

- /creer\_joueur (supprimé, la fonctionnalité est incluse dans /) : La création de joueurs se fait désormais directement sur la page d'accueil.

- /simuler\_joueurs (POST) : Permet de générer des joueurs simulés avec des numéros et étoiles aléatoires, limités à 100 joueurs maximum.

- /tirage (GET, POST) : Lance un tirage de loterie, génère les résultats pour tous les joueurs participants, et calcule les gains en fonction des numéros et étoiles correspondants.

- /resultats (GET) : Affiche les résultats du dernier tirage avec le classement des joueurs et leurs gains respectifs.

- /supprimer\_joueur/<int:id> (GET, POST) : Permet de supprimer un joueur après confirmation.

- /modifier\_numeros/<int:id> (GET, POST) : Permet de modifier les numéros et étoiles d’un joueur.

- /supprimer\_tous\_les\_joueurs (POST) : Supprime tous les joueurs de la base de données.

- /reclamer/<int:joueur\_id> (GET, POST) : Permet à un joueur de réclamer son gain en saisissant son mot de passe.

### - /modifier\_participation/<int:id> (POST) : Permet de modifier l'état de participation d'un joueur (activer ou désactiver la participation).

### 4. Sécurité et Intégrité des Données

#### Validation des Données :

* **Vérification des pseudos uniques** : Dans le processus d'inscription, le pseudo de chaque joueur est vérifié pour garantir qu'il soit unique. Si un pseudo est déjà utilisé, un message d'erreur clair est affiché à l'utilisateur via flash("Pseudo déjà pris, veuillez choisir un autre.", "error"), et le formulaire d'inscription est rechargé.
* **Contrôle des entrées utilisateur pour éviter les injections SQL** : SQLAlchemy, utilisé dans le projet, gère automatiquement la prévention des injections SQL en utilisant des requêtes préparées. De plus, les entrées utilisateur telles que les pseudos, numéros, et étoiles sont soigneusement récupérées et validées. Par exemple, la longueur du pseudo est vérifiée (len(pseudo) < 2 or len(pseudo) > 15), et les numéros et étoiles sont validés pour être correctement formatés et dans les plages autorisées.

#### Gestion des Erreurs :

* **Messages d'erreur clairs en cas de problème** : Le système affiche des messages explicites à l'utilisateur en cas de problème grâce à la fonction flash(). Par exemple, si le pseudo est déjà pris, l'utilisateur reçoit un message lui demandant de choisir un autre pseudo. De même, lors d'une tentative de tirage sans participants, un message est affiché pour indiquer qu'aucun joueur n'est inscrit pour ce tirage : "Aucun joueur participant n'est inscrit pour ce tirage.". Ces messages permettent à l'utilisateur de comprendre rapidement l'origine du problème et de corriger ses actions.

## Interface Utilisateur

### 1. Design et Ergonomie

- Interface Claire : Utilisation de couleurs et de typographies lisibles pour faciliter la lecture.

- Navigation Intuitive : Menu principal accessible sur toutes les pages pour naviguer entre les différentes fonctionnalités.

- Responsive Design : Adaptation de l'affichage pour les plus petits écrans

### 2. Pages Principales

#### 2.1. **Page d'Accueil** (/)

* **Liste des Joueurs** :
  + La page d'accueil affiche un tableau des joueurs inscrits, leurs pseudos, leurs numéros, leurs étoiles et leur statut de participation.
  + Si l'utilisateur veut créer un joueur, il peut le faire directement sur cette page via un formulaire d'inscription.
* **Actions Disponibles** :
  + **Créer un nouveau joueur** : Le formulaire d'inscription permet de créer un joueur en renseignant un pseudo, un mot de passe, et en sélectionnant des numéros et des étoiles manuellement ou automatiquement.
  + **Simuler des joueurs** : Un bouton renvoie à la route /simuler\_joueurs pour générer des joueurs fictifs avec des numéros et étoiles aléatoires.
  + **Lancer un tirage** : Un lien ou bouton permet de lancer un tirage via la route /tirage où le montant du tirage peut être renseigné.
  + **Voir les résultats du dernier tirage** : Un lien permet de consulter les résultats via la route /resultats pour afficher le dernier tirage effectué.

#### 2.2. **Formulaires**

* **Création de Joueur** (sur / via le formulaire d'inscription) :
  + Ce formulaire contient des champs pour :
    - Le pseudo du joueur.
    - Un mot de passe.
    - Des numéros et étoiles choisis manuellement, ou générés automatiquement.
    - Un bouton pour valider et soumettre le formulaire.
* **Simulation de Joueurs** (/simuler\_joueurs) :
  + Un formulaire sur cette page demande le nombre de joueurs à simuler.
  + Les joueurs générés automatiquement participent par défaut au prochain tirage avec des numéros et étoiles aléatoires.
* **Lancer un Tirage** (/tirage) :
  + Ce formulaire demande à l'utilisateur de renseigner le montant total du tirage, puis génère des numéros et étoiles gagnants de façon aléatoire.
  + Il calcule également les résultats pour chaque joueur participant en comparant les numéros gagnants avec ceux des joueurs inscrits.

#### 2.3. **Pages de Résultats**

* **Affichage des Numéros Gagnants** (/tirage) :
  + Après avoir lancé un tirage, la page affiche les numéros et étoiles gagnants sous une forme visuelle simple et les résultats de chaque joueur (leurs bons numéros, étoiles et les gains).
* **Classement** (/resultats) :
  + Cette page présente un tableau avec le classement des joueurs ayant participé au dernier tirage. Elle affiche :
    - Le pseudo du joueur.
    - Les numéros et étoiles qu'il a choisis.
    - Le nombre de bons numéros et étoiles trouvés.
    - Le gain du joueur

#### 2.4. **Pages de Gestion des Joueurs**

* **Modifier les numéros d’un joueur** (/modifier\_numeros/<int:id>) :
  + Cette page permet de modifier les numéros et étoiles d'un joueur déjà inscrit en les sélectionnant à nouveau.
* **Supprimer un joueur** (/supprimer\_joueur/<int:id>) :
  + Cette page demande une confirmation avant de supprimer un joueur spécifique.
* **Réclamer un gain** (/reclamer/<int:joueur\_id>) :
  + Cette page permet à un joueur de réclamer son gain en entrant son mot de passe. Si le mot de passe est correct et que le joueur a un gain à réclamer, il reçoit les informations nécessaires pour récupérer son gain.
* **Modifier la participation d’un joueur** (/modifier\_participation/<int:id>) :
  + Cette route permet d'activer ou de désactiver la participation d'un joueur au prochain tirage.