FERROLI TOUITI Evan

Documentation de l’application des Parcs d’attraction

Schéma de la base de donnée 1
Fonctionnalités ajoutés sur le site 2
Modifications possible 7
Versions utilisé pour le site 8

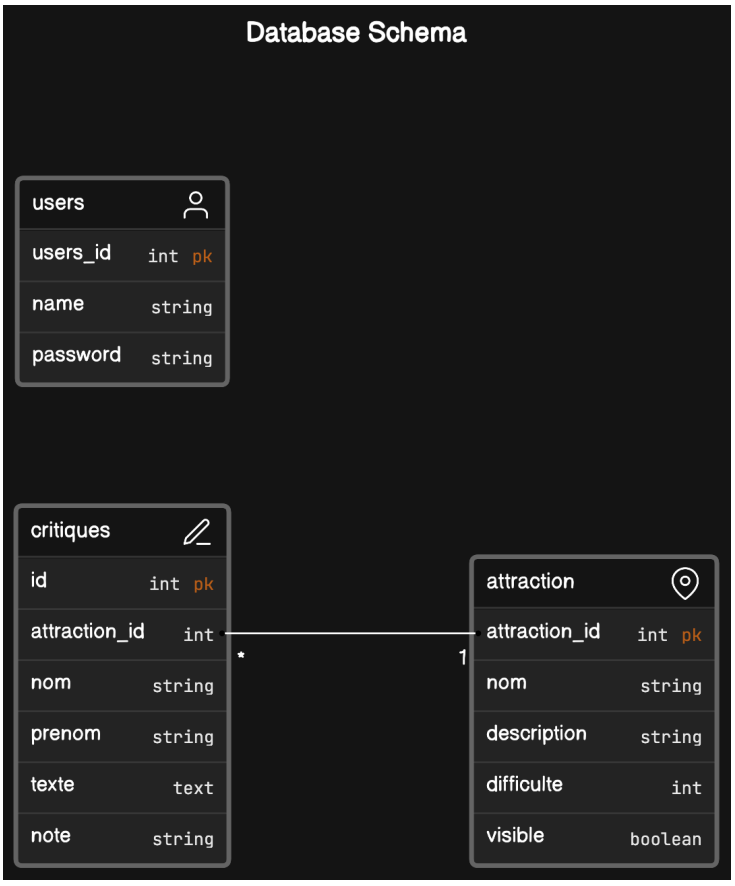
Exécuté sous MACOS, il y a donc les identifiants de la BD à modifier dans init.py et request/request.py

Ou alors directement exécuter les commandes ./copy.sh dans le dossier parc et la commande python dans le dossier python.

Sinon, il faut modifier les fichiers request/request.py et init.py et remettre les informations de la base de données de docker.

Schéma de la base de donnée:

Voici un schéma de la base de donnée du projet:



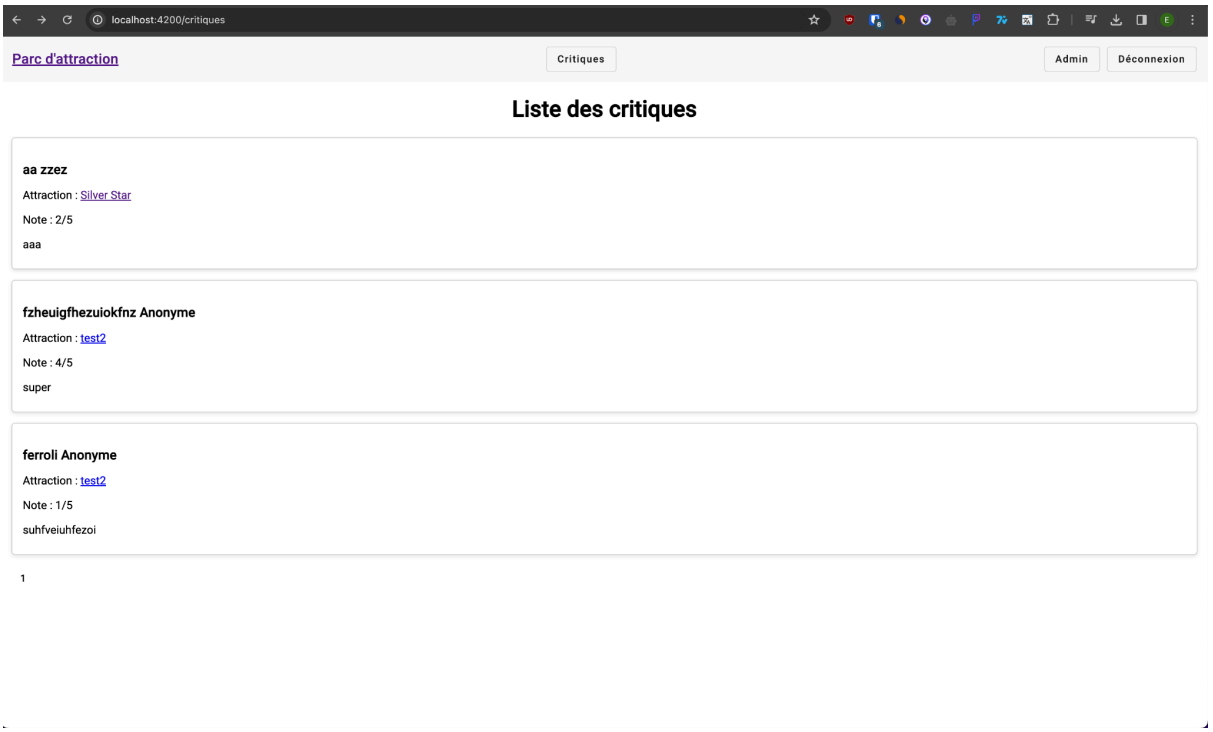
Nous avons donc une table users qui stockera les données des administrateurs tels que le nom et le mot de passe

La table attraction stockera les données des attractions comme le nom de l'attraction, la description, la difficulté et l'état, si elle est visible ou non sur le site.

La table critique stockera les critiques que des personnes mettront s'il le veulent. Ils n'ont pas besoin d'être connectés pour poster une critique et le nom n'est pas obligatoire. Il y a une clé étrangère qui fait référence à l'id de l'attraction, comme ça, si l'attraction est supprimée, les critiques liées à cette attraction seront automatiquement supprimées. Il y a aussi un nom, un prénom (non obligatoire), un texte de maximum 500 caractères et une note sur 5 donnée par l'auteur de la critique

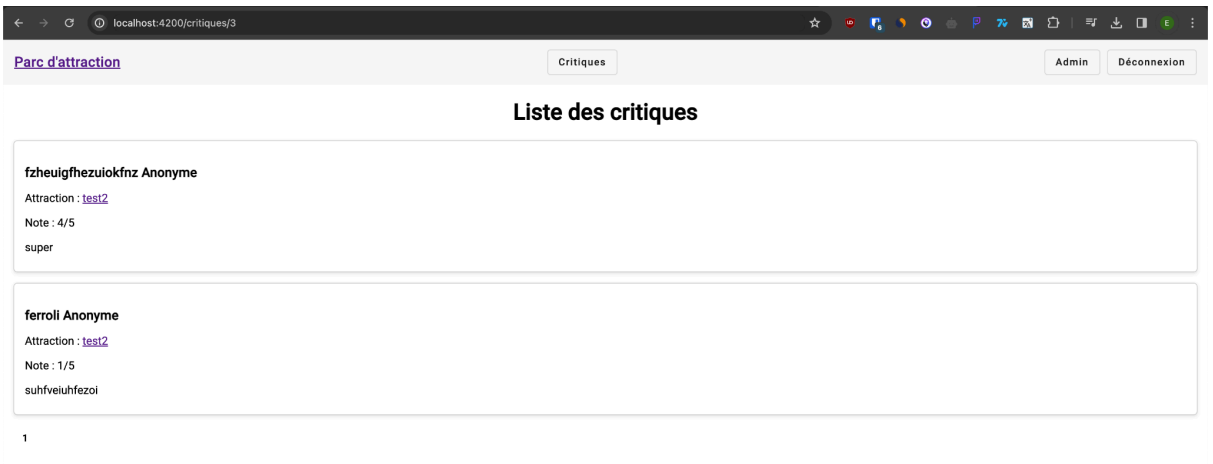
Fonctionnalités ajoutés sur le site

Pour commencer, voici le système des critique :



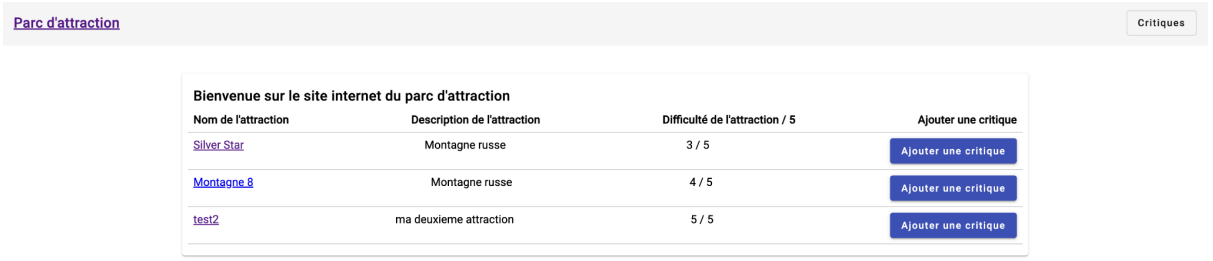
Nous avons un bouton dans la barre en haut afin d'accéder facilement à toutes les critiques. Vous pouvez cliquer sur le nom d'une attraction afin d'accéder aux critiques d'une attraction en particulier comme on peut le voir sur ce screen.

FERROLI TOUITI Evan



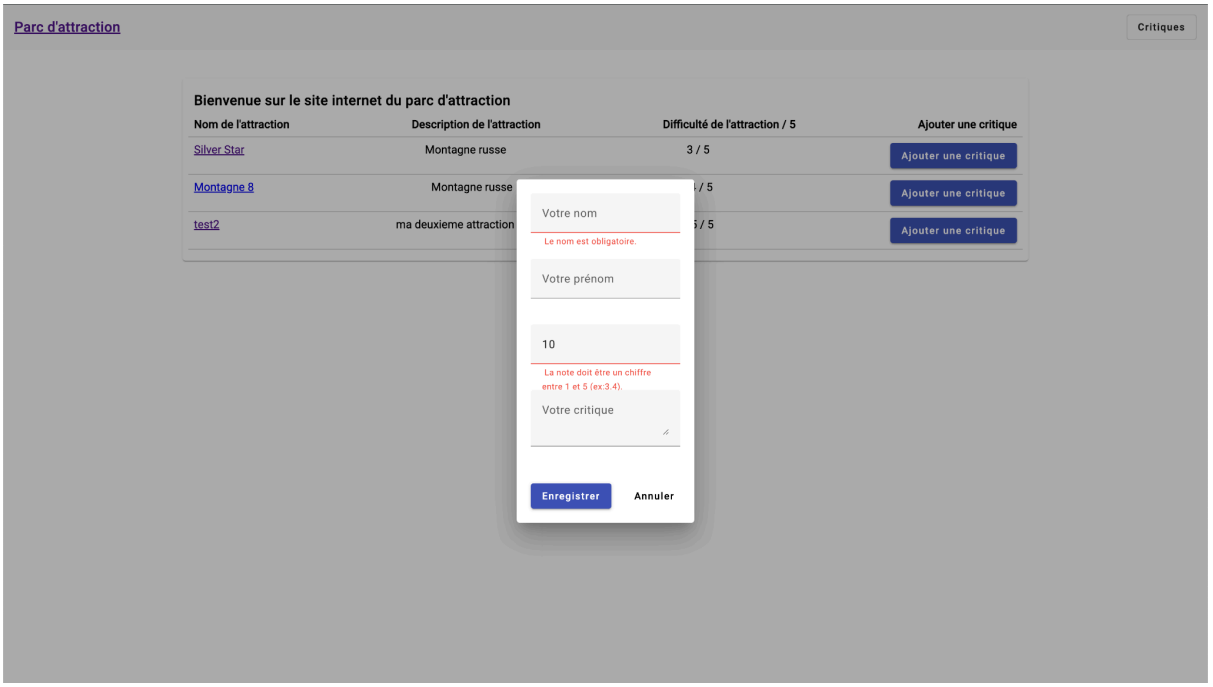
Vous pouvez aussi accéder aux critiques d'une attraction en cliquant sur le nom d'une attraction depuis la page d'accueil.

Comme on peut le voir, j'ai redessiné un peu la page d'accueil pour rendre le tableau plus jolie et j'ai ajouter la possibilité d'écrire une critique sur une attraction, que l'on soit connecté ou non :



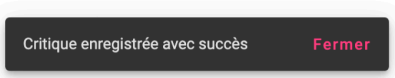
En cliquant sur ajouter une critique, vous aurez accès à un formulaire afin d'écrire votre critique:

FERROLI TOUITI Evan



Des requis on été ajouté, par exemple, si on ne rempli pas son nom, la note ou le texte de la critique, cela nous mettra une remarque en rouge en bas. Seul le prénom n'est pas obligatoire comme indiqué sur le Cahier des Charges. Celui-ci sera remplacé par la mention "Anonyme" dans la base de données et sur le site.

Une fois enregistré, une pop-up s'affichera en bas afin de nous confirmer que la critique à bien été publiée.

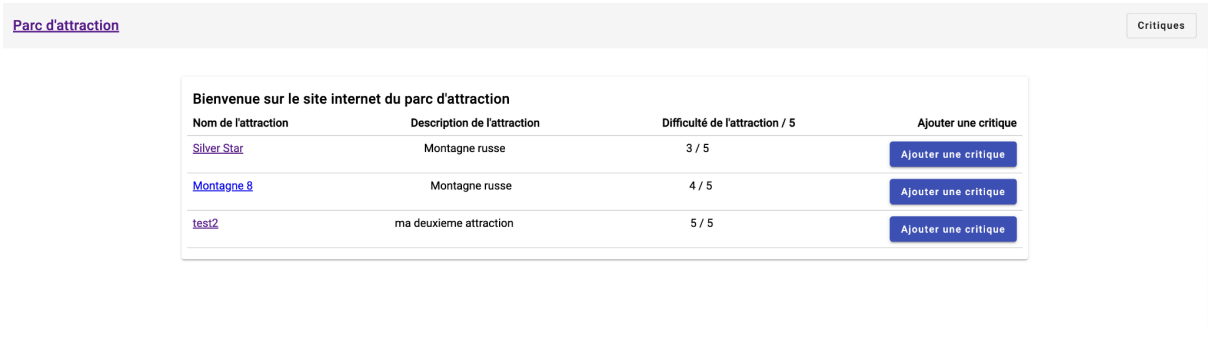


La langue anglais à également été ajoutée grâce à l'outil i18n de Angular.
Pour l'instant, le site n'est pas disponible en Anglais et Francais simultanément, il faut lancer les 2 instances séparément grâce aux commandes suivantes :

Anglais :
ng serve --configuration=en --open

FERROLI TOUITI Evan

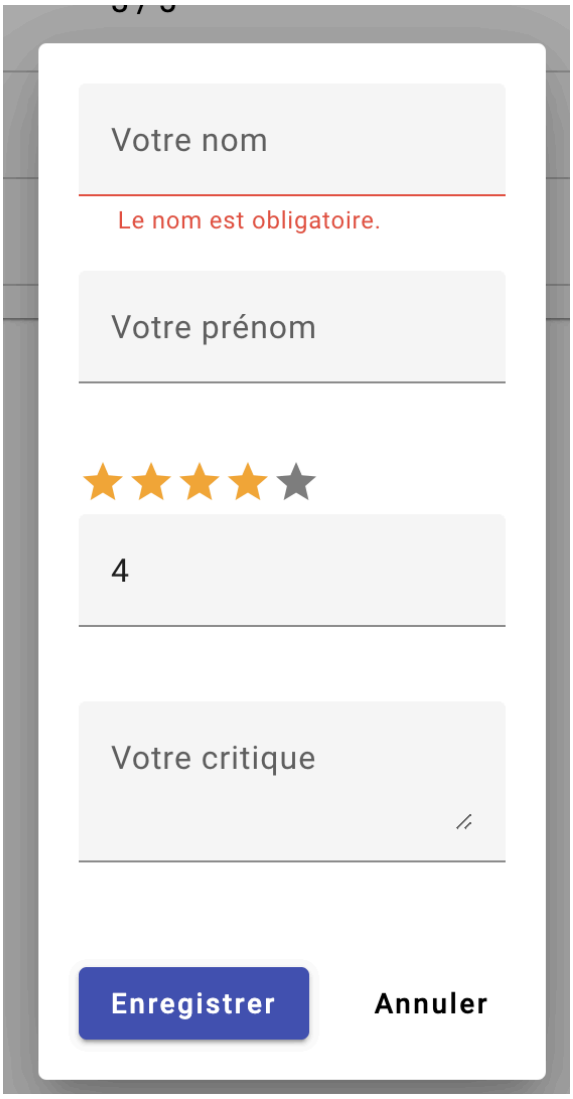
Français :
ng serve



Pour cela, il y a dans le dossier local, il y a un fichier messages.xlf, qui sont les messages originaux avec un ID unique pour les différencier.
Il y a un autre fichier appelé messages.en.xlf qui contient les phrases traduite avec toujours l'id.
Chaque instance va utiliser le fichier dont elle a besoin. C'est défini dans le fichier angular.json.

FERROLI TOUITI Evan

Bonus:



Système d’étoile pour ajouter une note de manière intuitive.

Système de moyenne:

Il y a les moyenne affichés en fonction des différentes notes des critiques.

FERROLI TOUITI Evan

Voici ce qui à été modifié sur le backend Python:
l'ajout dans le init.SQL de la structure de la nouvelle tables pour les critiques:

```sql
CREATE TABLE `critiques` (
  `id` int NOT NULL AUTO_INCREMENT,
  `attraction_id` int NOT NULL,
  `nom` varchar(255) DEFAULT NULL,
  `prenom` varchar(255) DEFAULT NULL,
  `texte` text NOT NULL,
  `note` varchar(255) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `attraction_id` (`attraction_id`),
  CONSTRAINT `critiques_ibfk_1` FOREIGN KEY (`attraction_id`) REFERENCES `attraction` (`attraction_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Il y a l'ajout des routes pour les critiques:

```python
# Critique
@app.post('/critiques')
def addCritique():
    json = request.get_json()
    retour = critiques.add_critiques(json)
    if (retour):
        return jsonify({"message": "Element ajouté.", "result": retour}), 200
    return jsonify({"message": "Erreur lors de l'ajout.", "result": retour}), 500

@app.get('/critiques')
def getAllCritique():
    result = critiques.get_all_critiques()
    return result, 200

@app.get('/critiques/<int:index>')
def getCritique(index):
    result = critiques.get_critiques(index)
    return result, 200

@app.delete('/critiques/<int:index>')
def deleteCritique(index):
    json = request.get_json()

    if (critiques.delete_critiques(index)):
        return "Element supprimé.", 200
    return jsonify({"message": "Erreur lors de la suppression."}), 500
```

Modifications possible:

- Comme modification et amélioration possible, il y a l'affichage d'un message d'erreur lorsque l'on saisit un login/mot de passe incorrect qui pourrait être correct.
- Sur ce que j'ai rajouté au niveau de la liste générale des critiques, ça serait pas mal d'ajouter un menu en haut pour choisir l'attraction, au lieu de revenir sur la page d'accueil et cliquer sur l'attraction.
- Ajouter un champs de recherche d'attraction dans la navbar ou dans la page d'accueil pour chercher l'attraction plus facilement/rapidement

FERROLI TOUITI Evan

- Ajouter un bouton de login en haut dans la navbar

Versions utilisé pour le site:

Angular : 17.1.0

Les librairies Python utilisés :

click v 8.1.3
colorama v 0.4.6
Flask v 2.2.2
Flask-Cors v 3.0.10
itsdangerous v 2.1.2
Jinja2 v 3.1.2
MarkupSafe v 2.1.1
PyJWT v 2.6.0
six v 1.16.0
Werkzeug v 2.2.2
mariadb v 1.1.10