COMPTE RENDU TP1 AVENEL THEOPHILE

DEVELOPPEMENT WEB

Au cours de ce tp j'ai pu mettre au fur et à mesure mon code sur Githib et le code ce trouve ici : https://github.com/thekester/tp1devwebnodejs . Le code valide se trouve dans la branche main. Vous pouvez voir les différents commits et les différent smerge request permettant de voir ma progression dans le code. Le README.md continent les différentes commandes pour lancer l'application node et contient ausis des gifs illustrant les différentes étapes réalisés au fur et à mesure du TP1.

Les difficultés rencontrés:

Coder la fonctionnalité pour acheter des produits car il fallait mettre en place un système de requête post j'ai donc utilisé Ajax pour pouvoir aussi mettre à jour la quantité disponible en stock en temps réel sans avoir besoin de refresh la page.

```
$(document).ready(function() {
  $('#acheter-btn').click(function() {
    // Vérifier si le bouton est désactivé
    if ($(this).is(':disabled')) {
      return;
    }

    // Confirmation avant l'achat
    if (confirm('Êtes-vous sûr de vouloir acheter ce produit ?')) {
      $.ajax({
        url: '/buy/' + #{produit.id},
        type: 'POST',       You, 9 hours ago • Add the buy button and modify the code implemen…
        success: function(response) {
          $('#message').text(response.message);
          if (response.success) {
            // Mettre à jour le stock affiché
            var nouveauStock = response.nouveauStock;
            $('.product-details p:nth-child(3)').html('<strong>Stock Disponible : </strong>' + nouveauStock + ' articles');
            // Si le stock est 0, désactiver le bouton et changer le texte
            if (nouveauStock === 0) {
              $('#acheter-btn').prop('disabled', true).text('Épuisé');
            }
          }
        },
        error: function(xhr, status, error) {
          $('#message').text('Erreur lors de l\'achat.');
        }
```

La création d'un volume persistant pour la base de données m'a aussi pris pas mal de temps

```
You, 5 hours ago | 1 author (You)
services:       You, 5 hours ago • Modify the buying function
  app:
    build: .
    ports:
      - "${PORT}:${PORT}"  # Assurez-vous que votre application écoute sur process.env.PORT
    volumes:
      - ./bdd/:/app/bdd/   # Monte le fichier de base de données
      - .:/app  # Monte le répertoire actuel dans /app dans le conteneur
    env_file:
      - .env  # Charge les variables d'environnement depuis .env
    restart: always

volumes:
  tavenel_db:

```

dans les volumes, il ne faut pas mettre les fichiers mais uniquement les dossiers où l'on veut que nos fichiers aillent car si on met par exemple /bdd/tavenel.bdd cela créera le dossier bdd et le dossier tavenele.bdd à l'intérieur du dossier bdd ce qui n'est pas le résultat souhaité.

```
const dbPath = path.join('./bdd', 'tavenel.db');
console.log(`Chemin de la base de données: ${dbPath}`);

const db = new sqlite3.Database(dbPath, (err) => {
  if (err) {
    console.error('Erreur lors de la connexion à la base de données:', err.message);
  } else {
    console.log('Connecté à la base de données SQLite.');
  }
});
```

.

Comme j'ai mis mon code dans Github j'ai souhaité mettre en place un workflow pour tester l'app node.js. J'ai suivi le tutoriel : https://medium.com/@realshamshod01/set-up-auto-deploy-for-node-js-app-just-in-4-steps-using-github-actions-and-pm2-7d192fbd2c37 et après avoir rencontrer de nombreuses difficultés, voici un exemple de workflow qui a fonctionné: https://github.com/thekester/tp1devwebnodejs/actions/runs/11710430349

Si vous le souhaitez, vous pouvez consulter cette vidéo pour voir uen exécution entière du pipeline CI/CD pour l'application node qui fait tourner pendant 40 secondes l'application sur une instance pm2 de Github (self host en local sur mon PC).

https://youtu.be/FfZZgWvjeGA